An Algebraic Theory of Componentised Interaction

Christopher James Chilton

Balliol College, Oxford

A thesis submitted to the University of Oxford for the Degree of Doctor of Philosophy

Trinity Term 2013

Department of Computer Science, University of Oxford
Wolfson Building, Parks Road, Oxford, OX1 3QD, UK

To my mother, father, sister and grandparents

Abstract

This thesis provides a specification theory with strong algebraic and compositionality properties, allowing for the systematic construction of new components out of existing ones, while ensuring that given properties continue to hold at each stage of system development. The theory shares similarities with the interface automata of de Alfaro and Henzinger, but is linear-time in the style of Dill's trace theory, and is endowed with a richer collection of operators.

Components are assumed to communicate with one another by synchronisation of input and output actions, with the component specifying the allowed sequences of interactions between itself and the environment. When the environment produces an interaction that the component is unwilling to receive, a communication mismatch occurs, which can correspond to run-time error or underspecification. These are modelled uniformly as inconsistencies. A linear-time refinement preorder corresponding to substitutivity preserves the absence of inconsistency under all environments, allowing for the safe replacement of components at run-time.

To build complex systems, a range of compositional operators are introduced, including parallel composition, logical conjunction and disjunction, hiding, and quotient. These can be used to examine the structural behaviour of a system, combine independently developed requirements, abstract behaviour, and incrementally synthesise missing components, respectively. It is shown that parallel composition is monotonic under refinement, conjunction and disjunction correspond to the meet and join operations on the refinement preorder, and quotient is the adjoint of parallel composition. Full abstraction results are presented for the equivalence defined as mutual refinement, a consequence of the refinement being the weakest preorder capturing substitutivity.

Extensions of the specification theory with progress-sensitivity (ensuring that refinement cannot introduce quiescence) and real-time constraints on when interactions may and may not occur are also presented. These theories are further complemented by assume-guarantee frameworks for supporting component-based reasoning, where contracts (characterising sets of components) separate the assumptions placed on the environment from the guarantees provided by the components. By defining the compositional operators directly on contracts, sound and complete assume-guarantee rules are formulated that preserve both safety and progress. Examples drawn from distributed systems are used to demonstrate how these rules can be used for mechanically deriving component-based designs.

Acknowledgments

Having reached the end of my doctoral research, it remains only to acknowledge the assistance and support I have received over the course of my studies. First and foremost I am grateful to Professor Marta Kwiatkowska for her guidance and enthusiasm in undertaking this work. Marta's reassurance and suggestions in regard to the direction of my research ultimately brought the aforementioned to fruition when I was unsure of its relevance and benefit. I am also grateful for the financial assistance I received from Marta that allowed me to undertake this doctoral study, in addition to attending numerous conferences, summer schools and meetings, as well as a stint in Uppsala. The opportunity to work with Marta has provided a stimulating environment for fostering my own learning and development.

Further thanks should be offered to Professor Bengt Jonsson and Dr Xu Wang with whom I collaborated extensively over the course of my work. A number of insightful discussions with Bengt and Wang essentially shaped various topics contained within this thesis. Thanks also to Bengt for hosting me in Uppsala on two occasions, which provided the opportunity to test new ideas and relate my work to practical application domains. Wang's extensive knowledge of timed systems made the final stages of thesis writing a less daunting task, and for that I am indebted.

This thesis was invariably influenced by numerous others, and I would like to acknowledge the engagement I have had with colleagues on the CONNECT project and those closer to home in Oxford, as well as my examiners: Professors Gerald Lüttgen and Joël Ouaknine. I am also grateful for the support I have received from friends and family over the course of my studies, which has been touching. In particular, I would like to thank my mother, father, sister and grandparents for their love and perseverance. To them I dedicate this thesis.

Finally, this work was made possible by the generous funding provided from the CONNECT project [ISJ+09], part of the Future and Emerging Technologies programme within the ICT theme of the Seventh Framework Programme for Research of the European Commission. Additional partial sponsorship was provided by ERC Advanced Grant VERIWARE, in addition to EPSRC projects UbiVal (EP/D076625/2) and LSCITS (EP/F001096/1).

Chris Chilton

Oxford, Michaelmas 2013

Contents

List of Figures

CHAPTER

ONE

Introduction

The complexity of modern computerised systems is growing at a phenomenal rate, owing to the availability of cheap and ever more efficient hardware, along with our willingness to embrace technology within all aspects of our lives. Physical devices and software entities are no longer anticipated to run in isolation, but are expected to interact seamlessly with one another so as to provide rich and cohesive services. Toleration of failure is increasingly unacceptable, whether in safety-critical domains, or in high-dependence environments, with the latter rapidly permeating an expanding number of application areas. While hardware failure and hardware-induced software anomalies are inevitable, software and interaction errors are often a consequence of incorrect design borne out of system complexity. One need only look at a handful of examples (e.g., the Ariane 5 rocket, which exploded after lift-off due to a software error in the inertial reference system) to see that the smallest of errors can lead to systemic malfunction on a colossal scale.

To avoid such undesirable behaviour, a sound methodology is required for the specification of error-free software systems (along with a procedure for generating code from the specification), and also adequate verification techniques are needed for identifying system errors. For all but the most trivial of systems, both of these approaches become infeasible due to issues of scalability, whether in the size of the specification, or the amount of time required for verification, unless appropriate abstraction techniques are adopted.

One approach to circumvent this issue is to take a componentised view of software design, whereby large systems are built compositionally out of smaller components, each of which performs a distinct and clear task that can be understood in isolation from the remainder of the system [McI68, Sha96, Sif05]. It is desirable that global properties of the composed system can be inferred, by proof rules, from the local properties of its constituent components. Without needing to compute the composition, a potentially exponential reduction can be obtained in the size of the state space.

Component-based design can be supported by a specification theory along with a reasoning framework. The specification theory provides an abstract formalism for modelling components, together with a wide range of operations for building new components incrementally and independently, and a refinement relation for comparing components. The reasoning framework allows for the specification of properties, provides a satisfaction relation indicating whether a property is ful-

filled by a component, and contains a number of compositional proof rules for inferring properties satisfied by the composition of components based on the properties satisfied individually.

A number of specification theories and reasoning frameworks have been developed previously, for a range of concurrent models of communication. This thesis provides formalisms for modelling and reasoning about the interactions arising in component-based systems, when communication is asynchronous and non-blocking. An overarching aim has been to provide support for modelling protocols, distributed systems and asynchronous hardware designs, where an environment supporting handshaking is either an unrealistic expectation, or must be achieved in a domain that is inherently asynchronous. The developed frameworks provide modelling notations capable of capturing the essential behaviour of components, in order to determine and reason about the communication mismatches that can arise through asynchrony. Existing formalisms for this communication model (such as, e.g., interface automata [dAH01], prefix-closed trace structures [Dil88], and the receptive process theory [Jos92]) lack cohesion, in the sense that they do not cover all of the aforementioned features, for building complex systems effectively.

1.1 Overview and Contribution of the Thesis

The topic of this thesis is firmly rooted at the heart of interface theory, where it is primarily concerned with the specification and verification of component-based systems. Components are assumed to communicate with one another by synchronisation of input and output (I/O) actions, with the behaviour of a component specifying the allowed sequences of interactions between itself and the environment. Unlike traditional I/O models of communication, such as CCS [Mil80] and I/O automata [LT89], outputs in the framework are assumed to be non-blocking. Taking this in combination with the fact that components are not required to be receptive on inputs means that composition of components can introduce incompatibilities in the form of communication mismatches, e.g., when one component issues an output that cannot be received by another component at that particular time. Communication mismatches and inconsistencies can be thought of as safety violations.

An overarching aim of the thesis is to formulate a specification theory of components, for modelling system interactions, which enjoys strong algebraic and compositional properties. The specification theory comes equipped with an appropriate notion of component (as discussed previously), but, unlike for process calculi (such as CCS [Mil80], CSP [BHR84] and the receptive process theory [Jos92]), components are not required to be built out of primitive terms; it is therefore assumed that the space of components from which new components can be built is already sufficiently rich.

A linear-time refinement preorder is adopted, which indicates whether a component can be safely substituted with another. Component $\mathcal{P}$ can be safely replaced by component $\mathcal{Q}$ if, for

any environment that $\mathcal{P}$ can interact with that does not introduce communication mismatches, it holds that $\mathcal{Q}$ will work in that environment without introducing communication mismatches. This is the weakest preorder preserving absence of communication mismatches (or equivalently preserving safety), which allows for the seamless replacement of any component by a refinement. To complete the specification theory, five operators are introduced for building new components compositionally:

- **Parallel composition.** This operator is used for examining the structural composition of a system of components, meaning that it shows the combined effect of the components interacting with one another. This is useful for examining any induced communication mismatches.
- **Conjunction.** This operator supports independent development of specifications, which loosely means that the conjunction of two components will work in any environment that is compatible for at least one of the components. Consequently, the conjunction yields a specification that is a refinement of both of the specifications to be conjoined.
- **Disjunction.** Being the dual of conjunction, the disjunction of two components yields a component whose compatible environments are safe for both operands. Therefore, the disjunction is refined by both of its operands.
- **Hiding.** This operator supports hierarchical development by contracting the interface (collection of observable I/O actions) of a component, in order to allow behaviour abstraction.
- **Quotient.** As the adjoint of parallel composition under the substitutive refinement preorder, given a system specification $\mathcal{R}$, together with a sub-specification $\mathcal{P}$, the quotient yields the least refined component $\mathcal{Q}$ such that the composition of $\mathcal{P}$ and $\mathcal{Q}$ refines $\mathcal{R}$. Quotient thus supports incremental development by means of synthesising missing components.

Strong algebraic and compositional properties are demonstrated for the operators, such as proving that parallel composition is monotonic under refinement and conjunction corresponds to the meet operator on the refinement preorder, which are crucial for supporting component-based design. This framework is useful for the subsequent body of work, outlined below, which increases the power of the base theory by developing a number of extensions for capturing more complicated behaviours.

- **Progress sensitivity.** This is an extension of the specification theory that instils a notion of finitary liveness in the form of quiescence. A component is said to be quiescent if it is unable to produce output without further stimulation from the environment. The refinement preorder is strengthened so that substitutivity continues to hold (corresponding to preservation of safety), in addition to the competing requirement that a non-quiescent behaviour

cannot be refined by a quiescent behaviour. This has a profound effect on the definitions of conjunction and quotient, since the former is the meet operator on refinement, while the latter is the adjoint of parallel composition with respect to refinement. Consequently, the compositionality properties need to be re-established.

- **Operational representations.** The substitutive and progress-sensitive specification theories are in the style of denotational semantics, in that components are defined in terms of trace sets, the compositional operators correspond to operations on traces, and refinement is formulated as trace containment. To support operational development, which is more akin to actual programs, an operational theory of components is presented for both the substitutive and progress-sensitive frameworks that enjoys the same compositional properties as in the trace-based equivalents. Under the assumption of determinism, it is demonstrated that refinement can be defined using the branching-time alternating simulation relation of de Alfaro and Henzinger [dAH01] as defined for interface automata.

- **Reasoning frameworks.** Assume-guarantee frameworks are developed for reasoning compositionally about safety properties satisfied by components in the substitutive theory, and progress properties satisfied by components in the progress-sensitive theory (see [JT96, CT12] for frameworks based on temporal logic). Properties are specified by contracts (see e.g. [Mey92, BHGQ10]), which consist of prefix closed sets of traces (corresponding to safety properties) representing assumptions made on the environment and guarantees provided by the component. For progress properties, a set of non-quiescent traces is also included in the contract. A satisfaction relation relates components with contracts whose properties are satisfied, and a collection of sound and complete assume-guarantee rules are presented for inferring the properties satisfied by compositions of components based on the properties satisfied by their constituent parts. To obtain these rules, a compositional theory of contracts is developed, whereby equivalents of the operators on the specification theory are defined directly on contracts.

- **Real-time modelling.** While the aforementioned specification theories capture the temporal ordering of interactions between components, they do no specify the actual time when these interactions occur. A trace-based framework is therefore developed for modelling the critical timing constraints imposed by componentised systems, capable of capturing safety and bounded liveness requirements. This framework comes in two flavours, in order to suit two different kinds of timed systems. The first allows for the global clock to be stopped, which enforces system termination, while the second insists that components must run indefinitely without the ability to terminate.

1.2 Dissertation Structure

Outlining the remainder of this dissertation, Chapter 2 highlights related work and makes explicit how those works differ from the theories presented in the subsequent chapters. Chapter 3 introduces the specification theory for modelling components, on which all of the extensions of this dissertation are based. Specifically, Section 3.1 presents a trace-based theory of components with respect to the substitutive refinement preorder, while Section 3.2 extends the previous section to the setting where the refinement preorder ensures both substitutivity as well as progress. Chapter 4 gives an alternate presentation of the previous chapter in terms of operational models for both the substitutive and progress-sensitive refinement preorders. The assume-guarantee frameworks are introduced in Chapter 5 for both the safety setting (Section 5.1, pertaining to the substitutive theory of components) and the progress-sensitive setting (Section 5.2, for the progress-sensitive theory of components). In Chapter 6, a real-time extension of the trace-based specification theory is presented. First, in Section 6.2, a theory is presented whereby systems are permitted to terminate, whereas Section 6.3 insists that components must run indefinitely. Based on Section 6.3, Chapter 7 introduces a real-time assume-guarantee framework for reasoning about the non-terminating theory of components. Finally, Chapter 8 concludes and discusses related work. Proofs for all of the key results are included throughout the dissertation at the appropriate points.

1.3 Published Material

Substantial portions of this dissertation have appeared in the proceedings of international conferences and journals. The substitutive compositional specification theory for components first appeared at the European Symposium on Programming [CCJK12] for the operations of parallel, conjunction and quotient, both in the trace-based and operational frameworks. An enhanced version including disjunction and hiding, in addition to progress-sensitivity, is available as [CJK13a] and has been recommended for publication, subject to minor revision, in Theoretical Computer Science. The technical results contained in the papers are my own, although I am grateful to Taolue Chen, Bengt Jonsson and Marta Kwiatkowska for their valuable guidance and fruitful discussions on the subject matter, and for assisting in the writing.

Similarly, the assume-guarantee framework appeared at the Formal Aspects of Component Software conference [CJK13b] for the operations of parallel, conjunction and quotient in the safety setting. An extension of the work with progress-sensitivity was published in a special issue of the Science of Computer Programming journal [CJK14]. Again, the technical results contained in both of the articles are my own, while the suggestion to look at the topic should be attributed to Bengt Jonsson and Marta Kwiatkowska, both of whom provided support throughout the work, and assisted in the writing of the papers. Special thanks should be offered to Bengt for the invitation

to undertake some of this work in Uppsala, and to Marta for making the visit logistically possible.

The high level ideas behind the timed specification theory can be attributed to discussions with Xu Wang, which culminated in the FORMATS paper [CKW12] and technical report [CKW13] (both additionally co-authored with Marta Kwiatkowska). While the contribution of those papers should largely be credited to Xu Wang, the content of this dissertation shares only some of the overarching principles, since it adopts an approach built on the untimed formalism of Chapter 3, rather than games.

The terminating theory of timed components in this dissertation was conceived out of the suggestion that components should be allowed to prevent the passage of time. This convention, also adopted in [CKW12], permits the use of terminating environments in the formulation of refinement, which ensures that the theory is a straightforward extension of the untimed substitutive framework, since (in the terminating theory) inconsistencies are propagated backwards only over output actions, rather than outputs and delays.

On the other hand, the non-terminating theory requires a more complex treatment of 'inconsistency inevitability', with the techniques described in this dissertation sharing similarity with the $\bot$-backpropagation method introduced in [CKW13]. Furthermore, the conjunction and quotient operators have a pruning operation applied to them that is closely related to the $\top$-backpropagation technique developed in [CKW13]; however, this also matches the error backpropagation technique developed for the untimed theory of progress-sensitive components.

A number of papers exhibiting practical applicability of the work in this dissertation also deserve mention. The paper [BCIJ13], which appeared in the proceedings of the Software Engineering and Formal Methods conference, shows how the progress-sensitive specification theory can be applied to the automatic synthesis of mediators, allowing functionally compatible yet not directly interoperable components to communicate with one another. The high-level methodology behind the theory (phrasing the task as a quotienting problem with an error-free and non-quiescent specification, and encoding ontology constraints as observer components) was developed in conjunction with Bengt Jonsson during my visit to Uppsala. The technical content related to ontological representations and reasoning was provided by Amel Bennaceur, while the prototype implementation was developed by Malte Isberner. The paper can be seen as a substantially refined and fleshed out version of [ACI+10], which appeared at the International Symposium on Leveraging Applications. This earlier paper attempted to build mediators in a compositional manner by composing atomic components, each of which can resolve basic communication incompatibilities. Essentially, however, the approach was not as fruitful as the automatable methodology appearing in [BCIJ13], although Inverardi and Tivoli (co-authors on the original paper) considered an enhanced version looking at protocol mediation patterns, culminating in [IT13].

CHAPTER

TWO

Related Work

This chapter provides an exposition of work related to the topic of the thesis, by covering the emergence of components, interface theories, process calculi, specification formalisms and contract-based reasoning frameworks. While most of the formulations are qualitative in nature, some broach the quantitative domain by encompassing real-time and stochastic behaviours. Where appropriate, comparisons are made with the work presented in this dissertation, although some contrasts are deferred until the technical framework has been established.

2.1 Emergence of Components

One of the earliest references to components from a computer science perspective can be traced back to a keynote speech of Douglas McIlroy in 1968, addressed to the NATO Software Engineering conference [McI68]. By analogy of a production line with sub-assemblies, McIlroy envisaged software engineers building systems out of componentised routines produced by third parties. The routines would be obtained from a component market, where they would be classified along a number of dimensions including precision, robustness, generality and time-space requirements. For reasons of scalability, McIlroy opined that each routine should be parameterised, so that a single implementation can satisfy all (or at least most) valuations of the aforementioned dimensions by a suitable selection of parameters.

It can be argued that this view of a component is diametrically opposed to our intended meaning nowadays. McIlroy focused on having a single component for different variants of the same function; however, this does not allow for the prospect of different manufacturers supplying variants of the same function. Thus, what is required is the concept of an interface to a component. This permits a user to switch between different implementations of a componentised routine, without having to recode their application (assuming the components share a common interface).

It is now standard for components to come part and parcel with an interface; it is what defines a component, other than its behaviour. There is no immediate point in the literature where this definition first arises, although it probably stems from the work on adding modularisation to programming languages. Parnas [Par72] discusses issues of modularity in such a context, and in particular raises the concept of independent development, a notion strongly connected with this

thesis, whereby distinct requirements can be combined.

Naturally, this work on modularity was a precursor to the inception of object-oriented programming (OOP) in the form of the Smalltalk language [GR83]. At the heart of the language lie the principles of encapsulation and separation of implementation from interface, key properties of component-based design. In terms of popularity, it can be argued that OOP is the epiphany of component-based design. However, that would serve a great injustice on a number of alternate theories that can be considered far more component-oriented than OOP, for example, WRIGHT [AG97], BIP [Sif05], and Reo [Arb04, BSAR06], to name but a few. OOP, despite its many advantages, does not support true component-based design in the sense of this thesis, in that it is not implementation agnostic. The content of this thesis is geared towards abstract modelling and verification, with the potential of being implemented, rather than focusing on practical application and instantiation as a means in itself. Therefore, from hereon, the emphasis of this chapter is directed towards mathematically abstract formalisms for modelling component-based systems.

2.2 Component Models and Specification Theories

Traditionally, components and their interactions were modelled by means of process calculi, which are formalisms for modelling and reasoning about concurrent systems. Communication in the process algebra CSP [BHR84] is based on handshaking, meaning that all concerned components must synchronise on common actions, thus differing from the form of communication adopted in this dissertation. WRIGHT [AG97] is a theory based on CSP that makes explicit the glue (a process) handling communication between components. Given the lack of high-level operations on CSP, such as quotient, the desired communication in the glue process must be crafted by hand, which is not amenable to large scale system development. CCS [Mil80] is a process calculus that distinguishes inputs from outputs and has a branching-time equivalence defined in terms of bisimulation, which differs from the linear-time refinements defined for CSP. However, the I/O distinction in CCS is purely syntactic, since outputs are blocking just like inputs. Consequently, the formalism does not capture the asynchrony required for our specification theory, as discussed in Chapter 1.

Process calculi build components out of primitive terms, such as the deadlocked process STOP in CSP or the empty process $\emptyset$ in CCS, with a range of basic operations such as prefixing, choice, hiding, renaming, parallel composition and recursion. This contrasts with the formalisms presented in this dissertation, where it is assumed that the space of components from which new components can be built is already sufficiently rich. Consequently, we focus our attention towards such formalisms from hereon, which tend to be automata-based (although there are exceptions).

Bliudze and Sifakis presented a theory of structured interaction, called BIP [Sif05] (Behaviour, Interaction, Priority), based on automata, for describing complex patterns of system interaction.

However, the theory does not support reasoning about communication mismatches arising through asynchrony, and does not provide compositional design operations, even though an algebraic theory of interaction is devised, for reasoning about common forms of system interaction. Similarly, the exogenous channel-based coordination model Reo [Arb04] does not support reasoning about asynchrony and does not consider high-level operations such as conjunction and quotient. Refinement is based on bisimulation and simulation (for the constraint automata semantics [BSAR06]), which does not correspond with (or is too strong for) substitutivity.

The theory of I/O automata, due to Lynch and Tuttle [LT89] (and independently introduced by Jonsson in [Jon87], an extended version of which is [Jon94]), is based on labelled transition systems, whose communication vocabularies, annotating the transitions, are partitioned into input, output and hidden actions. Since a component cannot prevent the environment from issuing an input, the automata are required to be input-receptive, meaning that each input must be enabled in every state. Consequently, substitutive refinement can be cast in terms of trace containment [Jon94] or simulation [LV95] (both under the assumption that the components to be compared have the same set of inputs). The former of these is similar to the refinement used by the formalism in this dissertation, except that it is more straightforward for I/O automata, since inconsistencies cannot arise due to input-receptivity. The operation of parallel composition is defined in the usual asynchronous way by synchronising on common actions and interleaving on the independent ones, while conjunction can be defined as a synchronous product, meaning that its set of traces is the intersection of its operands' traces (again when the input action sets are the same). Disjunction can be defined simply as the union of the behaviours. Hiding is already defined on outputs [Jon94] (we also define it on inputs) and quotient can be defined in a straightforward manner [DvB99], since input-receptivity ensures that communication mismatches cannot occur.

Just over a decade later, de Alfaro and Henzinger formulated an optimistic game-based version of I/O automata, called interface automata [dAH01], that drops the requirement of input-receptivity on components. Therefore, an interface automaton provides assumptions on what the environment can and cannot issue in each state. Refinement is defined in terms of an alternating simulation [AHKV98], which is substitutive, but is stronger than the linear-time characterisation used in this dissertation (a simplified version of alternating refinement for input-deterministic interface automata is presented in [dAH05]). Alternating refinement is too strong for determining substitutivity due to the necessity of selecting a matching transition to complete the simulation under non-determinism. The operational component representation in this dissertation differs from interface automata, in that we provide an explicit representation for inconsistency, which must be inferred through non-input enabledness in interface automata. This has repercussions for parallel composition, defined as a cross-product for interface automata, by synchronising on common actions and interleaving on the independent ones, followed by a pruning process, which removes any input transition from which there is a sequence of outputs and hidden actions leading to a state

whereby one of the components can produce an output not accepted by the other component. This differs from the framework in this dissertation, since the explicit representation of inconsistencies ensures that communication mismatches automatically arise as inconsistencies in the product, and the pruning technique is included as part of refinement, rather than the parallel operator. Conjunction and disjunction have been defined on input-deterministic interface automata when the components to be composed have the same interface (see [LV13]), while [DHJP08] defines conjunction (called shared refinement) on a synchronous component model. A definition of quotient has been provided for deterministic interface automata by Bhaduri and Ramesh [BR08], which mirrors the method developed by Verhoeff [Ver94].

Bujtor and Vogler have detailed three characterisations of error-pruning for interface automata [BV12], with an optimistic approach based on local errors (i.e., errors reachable through output and hidden transitions) matching that developed by de Alfaro and Henzinger, and also sharing similarities with the component formalism described in this dissertation and presented in [CCJK12]. The full abstraction result of [BV12] is essentially the same as the one we provide for our framework. The other characterisations are based on error propagation through hidden transitions alone, and through all types of transition. However, neither of these error-pruning operations corresponds to substitutivity, and therefore they are not relevant to this dissertation.

As a precursor to interface automata, de Nicola and Segala developed a process-algebraic characterisation of I/O automata [dNS95] that is actually applicable to interface automata, due to the introduction of non-input enabledness through the prefix and choice operators. Each process is associated with a sort, which corresponds to the interface through which the process can interact. Two semantics are provided for dealing with non-enabled inputs. The angelic approach assumes that a process remains unchanged when a non-enabled input is received, while the demonic approach forces a process to become chaotic on receiving a non-enabled input. The demonic semantics allows for the modelling of interface automata, since receiving a non-enabled input in an interface automaton allows erroneous unconstrained behaviour to ensue. Refinement is defined by trace inclusion, but does not extend to inconsistent trace containment as in this dissertation. Consequently, the theory is not able to distinguish a non-enabled input from one that is enabled and can subsequently behave chaotically. The process algebra supports the operations of prefixing, parallel composition, internal and external choice, hiding, renaming and recursion, but does not support the high-level operations of conjunction and quotient. Compositionality results and algebraic laws are established for the operators of the theory, with respect to the refinement relation. A shortcoming of the framework, like for all process calculi, is that a component must be built out of primitive terms, which can be quite tedious and impractical. It is also difficult to examine the behaviour of such a process, without considering a semantic representation.

Logic LTSs, devised by Lüttgen and Vogler [LV07], 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 [LV10], but no quotient), and refinement is given by ready-simulation, a branching time relation that requires the refining component not to introduce any new logical 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 the Logic LTS operations of, for example, conjunction, are incomparable to those in this dissertation. Taking inspiration from [LV07], the operational models of components introduced in this dissertation are essentially I/O automata augmented by an inconsistency predicate on states for indicating communication mismatches (and, consequently, non-enabled inputs, in addition to other faults), making our formalism achieve goals similar to those for interface automata, but with notation and semantics derived from I/O automata and Logic LTSs.

By considering the traces of our operational models, it is possible to extract a semantic model of components that records essential information related to the structure of a component for determining substitutivity. This trace-based model, which captures the observable and inconsistent behaviour of a component, essentially matches the prefix-closed trace structures devised by Dill [Dil88] for modelling the I/O interactions in asynchronous circuits. Dill provides a conformance relation, which is similar to our substitutive refinement, except that we generalise this to allow non-identical (static) alphabets. A liveness extension is also provided, based on infinite traces, while we consider the weaker notion of progress, based on quiescence, which utilises finite-length traces. A richer collection of operators are considered in this dissertation, such as conjunction and quotient, whereas Dill largely focuses on parallel composition and hiding.

In a style similar to Dill, Josephs *et al.* [JHJ89] formulate an I/O extension of CSP [BHR84] for modelling asynchronous circuits. The work differs from this dissertation in that processes must communicate through unbounded buffers, which eliminates the possibility of communication errors arising through non-enabledness of inputs. Avoiding this, Josephs [Jos92] formulates a theory of receptive processes, where components must communicate directly with one another. This has connections to our progress-sensitive 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. A further difference is that the receptive process theory is incapable of distinguishing communication mismatches from behaviour that repeatedly performs any output, due to insufficient information being recorded about the process. Josephs' work does not consider conjunction and quotient (the latter is defined on the restricted class of delay-insensitive networks [JK07], where it is referred to as factorisation; however, this does not match our setting). A similar framework, due to Jonsson, is documented in [Jon91].

Substitutivity of asynchronous systems can be determined using model-based testing, such as the ioco (input-output conformance) theory [Tre11], which checks that the outputs emitted by a

component after any trace conform to those permitted by the test specification. Only basic operations are defined on the ioco theory, such as parallel composition and hiding, but even for this small subset it is demonstrated that the conformance relation is non-compositional in general. Aarts and Vaandrager highlight the similarities between interface automata and the ioco theory [AV10], a key result of that paper equating quiescence-extended alternating simulation refinement on interface automata with the ioco relation, under determinism of models. The quiescence-extended alternating refinement is obtained by making quiescence observable: a self-loop annotated with a unique symbol δ, treated as an output, is appended to each quiescent state in the interface automata. Having performed this decoration, the standard alternating refinement check can be performed. In comparison with our framework, this implies that the ioco relation coincides with our progress-sensitive refinement for deterministic components free of divergence, since under such a restriction, our substitutive refinement relation is identical to alternating refinement (see Section 4.8 for the details) and a simple correspondence can be drawn between the progress-sensitive versions.

Increasingly, automata-theoretic models for component-based systems have taken the form of a specification theory, where a specification captures the requirements for a component to function in the 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. The justification for such frameworks is aptly described in [RBB$^+$09b, BCN$^+$12], whereby the authors highlight the importance of substitutive refinement and the necessity of being able to combine independently developed system requirements.

Larsen introduced modal specifications as a formalism for representing loose specifications, that is, specifications whose obligations and requirements can be refined over time towards a more concrete system description [Lar90]. A modal specification can be thought of as a (non I/O) LTS whose transitions are labelled by interactions and modalities, the latter indicating whether an interaction may or must be able to happen. Refinement is given by a branching-time relation approximating something between an alternating simulation and bisimulation, and the logical operations of conjunction and disjunction are provided, together with the standard operations of CCS. However, as highlighted in [LV13], the conjunctive and disjunctive operators are not closed in the sense that they yield well-defined modal transition systems. A number of attempts have been made to address this (see, e.g., [LX90, BCK11]), while [LV13] present the most satisfactory solution, where conjunction and disjunction are defined on a subclass of disjunctive modal transition systems that permit τ transitions. In [RBB$^+$09a, RBB$^+$09b, RBB$^+$11], a specification theory for modal specifications is introduced that considers a substitutive refinement relation, along with the operations of parallel composition, conjunction and quotient [Rac08]. The notion of liveness and progress is based on must-modalities, and thus differs from the trace-based formulation is this dissertation.

Extensions of modal specifications to the I/O setting are also provided in [RBB$^+$09b, RBB$^+$11],

where a mapping is given from deterministic interface automata without hidden actions to modal interfaces. These theories of modal interfaces (modal specifications with I/O distinction) are similar to the theory of [LNW07], 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 [RBB+09b, RBB+11] 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.

Lüttgen and Vogler provide a complete reformulation of the specification theory for modal interface automata [LV13], which addresses a number of shortcomings in the existing works. Principally, models are not required to be deterministic, and, moreover, a mapping from input-determistic interface automata to modal interface automata is supplied. Definitions are provided for the operations of parallel, conjunction and disjunction, which are all shown to be pre-congruences, while the latter two are shown to be the meet and join operators on the refinement preorder, respectively. However, a definition of quotient has not been formulated.

The frameworks developed in this dissertation have similarities with the modal interfaces and modal interface automata just mentioned, but we have different component representations and linear-time refinement preorders. However, we partially overlap in the range of operators considered, together with the compositionality properties that are demonstrated, which are essential for supporting component-based design.

2.3 Contract-based Reasoning

Contracts within computer science have been around for decades, an early example being the specification of a program in Floyd-Hoare logic [Hoa69] by means of pre- and post-conditions, although it is difficult to find widespread use of the term prior to Meyer's discursive appraisal [Mey92]. Since then, contracts have naturally been associated with assume-guarantee (AG) reasoning [MC81], whereby a component provides guarantees about its behaviour, under the assumption that its environment behaves in a particular way. Such frameworks should be compositional and equipped with sound proof rules, so that properties can be inferred about composite systems by examining the constituents from which it is composed. By not performing the composition,

issues of scalabilty can be circumvented or reduced when performing model checking [AENT03].

Historically, AG reasoning was concerned with compositional reasoning for processes, components and properties expressed in temporal logics, such as LTL and CTL [Pnu85, CLM89, GL94, JT96], where the overarching aim was to specify and verify concurrent systems. A variety of rule formats have been proposed, including symmetric, asymmetric and circular types, which are nearly always shown to be sound, and sometimes complete. It is often the case that the premises of such rules require the learning of auxiliary assumptions [CGP03], so that systems can be decomposed in an effective manner.

Maier [Mai01] provides an abstract framework for reasoning about parallel composition in a circular manner, as do Amla *et al.* in [AENT03]. The framework that we present allows for more types of decomposition, by also supporting the operations of conjunction, disjunction and quotient on contracts. Moreover, it is not obvious to see that the frameworks of [Mai01, AENT03] support the component formulation we use based on non-blocking outputs. Similarly, Ben-Hafaiedh *et al.* [BHGQ10] provide a contract-based framework for reasoning about safety and progress properties of components, in a framework where communication is encoded by a subset of the BIP [Sif05] interaction primitives, which do not support reasoning about asynchrony. Again, composition is restricted to parallel, and the notion of progress differs from the type we introduce based on quiescence.

Abadi and Lamport [AL93] considered compositional reasoning for contracts in the generic setting of state-based processes, a revised version of which is [AL95], based on the temporal logic of actions. They formulated a Compositionality Principle for parallel, which is shown to be sound for safety properties. A logical formulation of specifications is discussed by [AP93], where intuitionistic and linear logic approaches are adopted. In contrast, our work considers an action-based component model and has a richer set of composition operators, including conjunction, disjunction and quotient.

Maier [Mai01] demonstrates through a set-theoretic setting that compositional circular AG rules for parallel composition (corresponding to intersection) cannot be both sound and complete. This seems to contradict the work of Namjoshi and Trefler [NT10], although the discrepancy can be attributed to the fact that their sound and complete circular rule is non-compositional, since they need to include auxiliary assertions. The compositional AG rule that we provide for parallel composition is sound and complete, since we rely on the convention that an output is controlled by at most one component, which breaks circularity.

More recent proposals focus on compositional verification for component theories such as interface and I/O automata. Emmi *et al.* [EGP08] extend a learning-based compositional AG method to interface automata. Sound and complete rules are presented for the original operators defined by [dAH01], namely compatibility, parallel and refinement based on alternating simulation, but

conjunction, disjunction and quotient are absent. Moreover, the rules are limited to being asymmetric in nature, and furthermore involve the learning of assumptions, using an approach such as that documented in [PGB+08].

By imposing input-receptivity on interface automata, Larsen *et al.* [LNW06] define an AG framework, where assumptions and guarantees are specified as I/O automata. A parallel operator is defined on contracts that is the weakest specification respecting independent implementability, for which a sound and complete rule is presented. Our work differs by not requiring input-enabledness of components or guarantees, and allowing for specifications to have non-identical alphabets to their implementations. We also define conjunction, disjunction and quotient, and support progress properties, thus providing a significantly richer reasoning framework.

The theory of modal interfaces [RBB+11], mentioned in the previous section, does not consider a contract-based framework, but does separate the notion of implementation from specification. Rules specifying the relationship between implementations and specifications under the operations of parallel, conjunction and quotient are provided, which share similarities with the sound and complete rules for our AG framework, although we define the compositional operators directly on specifications represented by contracts.

A contract-based presentation of modal specifications is introduced in [BDH+12], where a generic construction is provided for obtaining a contract-based framework from a component-based specification theory equipped with the operations of parallel, conjunction and quotient. However, the contract-based framework only comes equipped with parallel composition; the operations of conjunction, disjunction and quotient are not defined directly on contracts, unlike in our framework. We briefly remark that [DCL11] defines conjunction on contracts, but this is for a simplified contract framework, as witnessed by the definition of parallel composition.

An abstract mathematical framework for contract-based design is presented in [BCF+08], based on set-theoretic operations on sets of behaviours. Although parallel composition, conjunction and disjunction are defined, no attempt is made at the arguably more difficult operation of quotient, which we also provide. The framework does not give consideration to the specifics of the execution model, hence it is unclear whether the rules can be instantiated for any particular communication model. Furthermore, AG rules are not provided for the framework, which prevents it from being used for reasoning about component-based systems.

Finally, we remark that [BCN+12] includes a detailed exposition on system design within a contract-based framework. An extensive comparison is made with related work, and large scale case studies are presented.

2.4 Quantitative Extensions

While the formalisms outlined previously are capable of capturing and reasoning about the temporal ordering of interactions, they are not geared towards analysing the non-functional properties of components, such as performance. Therefore, we consider a number of formalisms that are quantitative in nature, whether that is in terms of real-time or stochastic behaviours.

Starting with real-time models, the timed automata due to Alur and Dill [AD94] are a formalism for modelling timed component interactions, but without I/O distinction. Historically the models induced timed languages consisting solely of infinite-length words, meaning that an infinite-number of interactions must occur in an execution, but subsequent reformulations have also considered finite length traces [OW07], as in this dissertation (note that we permit components to terminate, and we also let components avoid interacting with the environment as long as time can pass unimpeded). The presentation in [AD94] is foundational, being concerned with notions of equivalence and modelling, rather than compositionality. Parallel composition has been defined, and is supported by the timed automata modelling tool UPPAAL [BDL04], but compositional operators in the form of a specification theory are scarce, unless I/O distinction is introduced.

A number of I/O sensitive timed automata formalisms have been presented, including [KLSV11, dAHS02, DLL$^+$10b]. The theory in [KLSV11], a natural timed extension of I/O automata [LT89], insists on input-receptivity, meaning that communication mismatches cannot arise through components being unwilling to accept an input at certain times. Components are not permitted to terminate, as in our timed theory of Section 6.3, which means that every state must be able to let time pass indefinitely, or there must be an enabled output action to another state, thus providing urgency semantics. Refinement is defined both in terms of trace containment and forward simulation [LV96], and definitions are provided for the operations of parallel composition and hiding.

The theory of timed interfaces [dAHS02], based on interface autoamata [dAH01], represents timed components by a timed game played by the Input and Output controllers. Invariants belonging to Output correspond to the invariants we introduce in our formalism, while invariants belonging to Input correspond to our co-invariants. The game consists of two transition systems, one for each player, whereby each player proposes a move in each state. A component is compatible with its environment if Input has a strategy whereby it can avoid communication mismatches arising, and does not require time to be blocked. Confusion over the obligations of who stops time need to be resolved by apportioning blame to the players. The timed game provides a semantics for the timed interface automaton, which is conceptually similar to the operational model we introduce. Our semantics are defined in terms of traces, and we provide a rich collection of operators and a notion of refinement, whereas only parallel composition is defined for timed interface automata.

A timed specification theory based on I/O automata is presented in [DLL$^+$10b], which sepa-

rates the notion of specification from implementation. Specifications and implementations must both be input-enabled, while an implementation is a specification that satisfies output urgency, meaning that if an output can be produced in a state then that state cannot delay, along with independent progress, which ensures a state must either delay indefinitely or it must produce an output at some point. Refinement is based on a timed variant of alternating simulation, differing from our trace-based formulation, and the operations of parallel composition, conjunction and quotient are defined. Tool support is provided by means of ECDAR [DLL+10a], with applications being shown in [DLL+12].

Zhou *et al.* [ZYM01] propose a timed trace version of Dill's theory of asynchronous circuits [Dil88], and explain that conformance can be checked in terms of mirroring. The definition of mirroring is more complicated in the timed setting, plus the framework does not target component-based design, as the operations considered are limited to parallel composition.

The paper [ČGL93] introduces timed modal specifications, but using a syntax derived from timed CCS [Wan90]. The range of operators considered is limited, consisting of delay, must and may prefixes, choice, parallel and hiding. As for modal specifications, the refinement is branching-time in nature, which differs from the weakest preorders preserving substitutivity that we formulate. Similarly, Bertrand *et al.* [BLPR09] present a compositional specification theory for timed modal specifications, using an automaton representation, incorporating the operations of parallel, conjunction and quotient. Refinement is given by a branching-time relation, differing from the linear-time formulation we provide, and there is no explicit separation of inputs from outputs.

Finally, Thiele *et al.* [TWS06] consider real-time interfaces, but based on a logical formulation different from our setting. A notion of refinement is provided, although the relationship to our theory is unclear. Parallel composition is defined for connecting interfaces, but the operations of conjunction, disjunction and quotient are not considered.

We now mention a number of formalisms that are probabilistic in nature. Given that a probabilistic extension is not considered in this dissertation, we mainly focus on those frameworks that take the form of a specification theory, rather than providing a detailed survey of the foundational models.

First, Xu *et al.* [XGG10] present a probabilistic contract-based framework, where implementations are represented by interactive Markov chains (an automaton whose states are partitioned into action and probabilistic states). An action state can engage in interactions with the environment, while the probabilistic state has a distribution over successor states. This model is generalised to a contract, characterising a set of implementations, which is an interactive Markov chain that has intervals of probability associated with its probabilistic successors. Refinement is given by probabilistic simulation, and the operations of parallel, conjunction and hiding are defined on contracts.

In the probabilistic setting, conjunction is only defined when the contracts to be conjoined (which must be normalised with respect to probabilistic bisimulation) are probabilistically similar.

Caillaud *et al.* [CDL$^+$10] adopt a specification theory similar to [XGG10], except that all states are probabilistic (since interactions are encoded in terms of sets of atomic propositions on states), while the allowable probabilities associated with successor states are encoded using linear constraints, rather than just intervals. The theory is endowed with the operations of parallel composition and conjunction, and refinement is akin to probabilistic simulation. In passing, we remark that Delahaye *et al.* [DCL11] also introduce a probabilistic contract framework including the operations of parallel and conjunction.

Before concluding this section, we comment that abstract probabilistic automata [DKL$^+$11] (and a probabilistic timed variant [HKKG13]) are formalisms suitable for representing specifications of components. A theory of probabilistic I/O automata has also been documented (including discrete probabilistic and continuous-time behaviour) [WSS97], including their reformulation in terms of switched probabilistic I/O automata [CLSV05] based on a scheduling mechanism.

CHAPTER

THREE

Trace-Based Theory of Components

This chapter formulates a compositional specification theory for component-based systems, where a component model specifies the allowed sequences of input and output interactions with the environment. Models are characterised by an interface, which specifies the interaction primitives that the component can engage in, along with sets of traces for representing the interactive behaviours. While outputs are controlled and issued by the component itself, inputs are under the control of, and are issued by, the environment. Since input receptiveness is not a requirement of the theory (unlike, for example, I/O automata [LT89]), a communication mismatch occurs when the environment issues an input that is observable, yet not accepted, by a component. Thus, a component model places assumptions on when observable inputs can be issued by the environment.

The trace-based representation of models is in the style of denotational semantics, in that it is deplete of structure relating to non-determinism and execution. In fact, the models capture precisely enough information for determining whether a component is substitutable with another. Informally, component $\mathcal{Q}$ is substitutable for $\mathcal{P}$ if, for each environment that $\mathcal{P}$ can interact with without introducing communication mismatches, it follows that $\mathcal{Q}$ can interact with that environment, also without introducing communication mismatches. Based on this principle, a linear-time refinement is provided, which is the weakest preorder preserving substitutivity of components. Such a relation is essential for component-based design, since it allows for the dynamic evolution of a system at run-time by replacing its subcomponents, whilst providing a guarantee that communication mismatches will not be introduced.

Section 3.1 introduces this substitutive specification theory, and 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. The component formulation highlights the algebraic properties of the specification theory, and is shown to be fully abstract with respect to observation of communication mismatches. These are indispensable properties for being able to reason compositionally about component-based systems. Through simple examples, it is demonstrated that the theory naturally supports a component-based design process that starts from some initial design considerations and applies the operations of the theory compositionally and in a stepwise fashion to obtain complex systems.

While substitutive refinement prevents the introduction of communication mismatches, it also promotes behaviour suppression, since a component that refuses to produce any output is an automatic refinement of any other component. To avoid such trivial refinements, Section 3.2 extends the specification theory by formulating a refinement preorder that guarantees substitutivity along with the preservation of progress. A behaviour of a component model is said to make progress if it can be extended by an output without having to receive further interaction from the environment. The enhanced refinement preorder thus stipulates that a refining component must make progress whenever the original can, in addition to satisfying the conditions for substitutivity. Such a strengthening of the refinement preorder requires reformulation of the compositional operators of the theory so that the algebraic properties are restored.

The substitutive specification theory contains similarities with Dill's prefix closed trace structures [Dil88], while the progress-sensitive theory has connections with Josephs' receptive process theory [Jos92]. A detailed comparison of those related works with the theory of this chapter is deferred until Section 3.3, after the technical details have been presented. It should be remarked that a preliminary version of the specification theory appeared in [CCJK12], although restricted to the substitutive setting for the operations of parallel, conjunction and quotient. Applicability of the framework was demonstrated in [IT13, BCIJ13], where the quotient operation was used to synthesise mediator components.

3.1 A Theory of Substitutable Components

In this section, we introduce a compositional specification theory for a trace-based representation of components. 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 the weakest refinement relation preserving safe substitutivity of components and provide definitions for the compositional operators of the theory.

Definition 3.1 (Component). A *component* $\mathcal{P}$ is a tuple $\langle \mathcal{A}^I_\mathcal{P}, \mathcal{A}^O_\mathcal{P}, T_\mathcal{P}, F_\mathcal{P} \rangle$ in which $\mathcal{A}^I_\mathcal{P}$ and $\mathcal{A}^O_\mathcal{P}$ 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 $t' \in (\mathcal{A}^I_\mathcal{P})^*$, 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. $\diamond$

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, for example, run-time errors and communication mismatches. Thus, $F_{\mathcal{P}}$ can be used to record the traces in $T_{\mathcal{P}}$ that involve non-enabled inputs. Based on this convention of distinguishing enabled from non-enabled inputs, we say that our *theory* is not input receptive, 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.

Example 3.2. Throughout this chapter, we use the following running example to demonstrate the suitability of our framework for component-based design. A multi-function device capable of printing and scanning is modelled as a component Device in Figure 3.1. 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 to 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 dissertation, whether a node is a circle, square or contains $\bullet$ is irrelevant; the distinction will become apparent later on. $\diamond$

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

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

3.1.1 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 $\mathcal{Q}$ to exist safely in *every* environment that is safe for $\mathcal{P}$. Whether

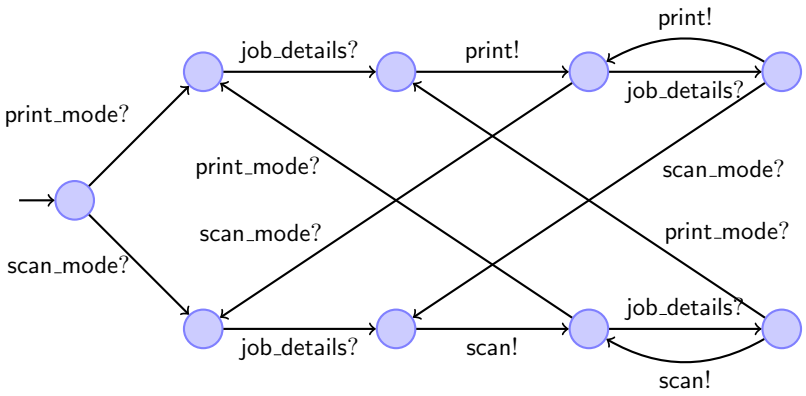

Figure 3.1: Printing/scanning device (Device)

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 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 3.3. 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 3.4 (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. $T_{\mathcal{E}(\mathcal{Q})} \subseteq T_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})$

I5. $F_{\mathcal{E}(\mathcal{Q})} \subseteq F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})$. ◇

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 [CCJK12] we did not impose this constraint, as it is not necessary to guarantee substitutivity. However, in this dissertation 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. The lifting $T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q}$ represents the extension of $\mathcal{P}$'s interface to include all inputs in $\mathcal{A}^I_\mathcal{Q} \setminus \mathcal{A}^I_\mathcal{P}$. As these inputs are not accepted by $\mathcal{P}$, they are treated as bad inputs, hence the suffix closure with arbitrary (chaotic) behaviour. Finally, condition I5 ensures that $\mathcal{Q}$ cannot introduce any new errors that are not in $\mathcal{P}$'s behaviour. Note that checking $F_\mathcal{Q} \subseteq F_\mathcal{P} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$ 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.

The motivation for treating inputs in $\mathcal{A}^I_\mathcal{Q} \setminus \mathcal{A}^I_\mathcal{P}$ as bad in $\mathcal{P}$ is a consequence of the behaviour arising through encountering such an input in $\mathcal{P}$ being unspecified, meaning that the behaviour of $\mathcal{Q}$ should be unconstrained. This convention shares similarity with the weak refinement defined by Raclet *et al.* [RBB$^+$09b], whereby the refining component $\mathcal{Q}$ may provide, although is not obliged to provide, safe behaviour after encountering an input in $\mathcal{A}^I_\mathcal{Q} \setminus \mathcal{A}^I_\mathcal{P}$. However, a key difference between our refinement and that of Raclet *et al.* is that for the latter, a refining component is required to respect the behaviour of the original component, after having encountered a non-specified input. This is due to their framework projecting out inputs in $\mathcal{A}^I_\mathcal{Q} \setminus \mathcal{A}^I_\mathcal{P}$, while we adopt a lifting that disregards subsequent behaviour. As an aside, Raclet *et al.* also provide a strong refinement, which ensures that inputs in $\mathcal{A}^I_\mathcal{Q} \setminus \mathcal{A}^I_\mathcal{P}$ must never result in unsafe behaviour. Our theory could have adopted one of these alternative conventions for handling unspecified inputs, which would in turn have affected the definition of refinement, and consequently the operations of conjunction, disjunction and quotient, which are all defined with respect to the choice of refinement.

Definition 3.5 (Equivalence). Components $\mathcal{P}$ and $\mathcal{Q}$ are said to be *equivalent*, written $\mathcal{P} \equiv_{imp} \mathcal{Q}$, iff $\mathcal{P} \sqsubseteq_{imp} \mathcal{Q}$ and $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$. $\diamond$

The refinement relation is reflexive, and is transitive subject to the compatibility constraint I3 holding. Unfortunately, this compatibility constraint cannot be inferred since inequality is not transitive. However, equivalence as defined above does form an equivalence relation without any further conditions.

Lemma 3.6. 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. Transitivity follows by transitivity of subset inclusion, given $T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}} \subseteq T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{R}}$, and $T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{R}} \subseteq T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{R}}$. □

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.

3.1.2 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 3.7. 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 $\mathcal{P} \mid\mid \mathcal{Q}$ 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.

In the theory of interface automata [dAH01] (which also does not insist on input receptivity, although component models are operational) a backward propagation of inconsistencies is performed over output actions, followed by a pruning operation on inconsistencies. This backward propagation is akin to the $\mathcal{E}$ operator of our theory. We do not need to use this operator to define the parallel composition, since backward propagation of inconsistencies is implicitly performed as part of our definition for refinement. Further detail can be consulted in Section 4.8.

As a further remark, we comment that the parallel composition operator for interface automata combines input and output actions into a hidden action. Our theory eschews this decision for two reasons. First, it is no longer possible to support broadcast communication, whereby one output can be received by multiple components. Secondly, parallel composition is not an associative operator when inputs and outputs combine to become hidden, which constrains the algebraic properties satisfied by the theory.

Lemma 3.8. The definition of parallel composition yields a component.

Proof. First note that $F_{\mathcal{P}||\mathcal{Q}}$ is closed under extensions, $F_{\mathcal{P}||\mathcal{Q}} \subseteq T_{\mathcal{P}||\mathcal{Q}}$ and $T_{\mathcal{P}||\mathcal{Q}}$ is prefix closed. To show receptivity, let $t \in T_{\mathcal{P}||\mathcal{Q}}$ and $a \in \mathcal{A}^I_{\mathcal{P}||\mathcal{Q}}$. If $t \in F_{\mathcal{P}||\mathcal{Q}}$, then certainly $ta \in F_{\mathcal{P}||\mathcal{Q}}$ and so $ta \in T_{\mathcal{P}||\mathcal{Q}}$. Instead, if $t \in T_{\mathcal{P}||\mathcal{Q}} \setminus F_{\mathcal{P}||\mathcal{Q}}$, then $t \upharpoonright \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}}$ and $t \upharpoonright \mathcal{A}_{\mathcal{Q}} \in T_{\mathcal{Q}}$. By input receptivity of $T_{\mathcal{P}}$ and $T_{\mathcal{Q}}$ it follows that $ta \upharpoonright \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}}$ and $ta \upharpoonright \mathcal{A}_{\mathcal{Q}} \in T_{\mathcal{Q}}$. Hence $ta \in T_{\mathcal{P}||\mathcal{Q}}$, and so $T_{\mathcal{P}||\mathcal{Q}}$ is closed under finitary input receptiveness. □

Lemma 3.9. 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 $\mathcal{A}_{(\mathcal{P}||\mathcal{Q})||\mathcal{R}}$.

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

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

□

Parallel is not idempotent in general because of composability, which requires disjointness of output actions. 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 3.10. 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 $t \in F_{\mathcal{E}(\mathcal{P}'||\mathcal{Q}')}$ implies $t \in F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P}||\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'})$ (and respectively for the T-sets), assume that the result holds for all strict prefixes of t.

First suppose that $t \notin \mathcal{A}^*_{\mathcal{P}||\mathcal{Q}}$. Then there exists a prefix $t'a$ of t such that $t' \in \mathcal{A}^*_{\mathcal{P}||\mathcal{Q}}$ and $a \in \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'} \setminus \mathcal{A}_{\mathcal{P}||\mathcal{Q}}$. As t' is a strict prefix of t, it follows that $t' \in T_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$ and so $t'a \in T_{\mathcal{E}(\mathcal{P}||\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'}$. Hence $t \in T_{\mathcal{E}(\mathcal{P}||\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'}$ as required.

Instead, if $t \in \mathcal{A}^*_{\mathcal{P}||\mathcal{Q}}$, then, in the case of showing F-containment, it follows that 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. □

Note that, in [RBB+11], parallel composition is claimed to be monotonic for modal interfaces without any conditions on the interfaces (except for composability). This is due to the authors using strong refinement (as remarked in Section 3.1.1), which is more restrictive than $\sqsubseteq_{imp}$, since it requires that all actions in $\mathcal{A}^I_{\mathcal{S}'_\mathcal{P}} \setminus \mathcal{A}^I_{\mathcal{S}_\mathcal{P}}$ and $\mathcal{A}^I_{\mathcal{S}'_\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$ never produce unsafe behaviour.

Example 3.11. The most liberal User that can interact with the Device (shown in Figure 3.1) 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 component equal to the pictorial representations of the Device and 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 3.1. Consequently, the trace would also be inconsistent in the parallel composition. ◇

3.1.3 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 [DHJP08, RBB+09b].

In a number of frameworks, including [LV07], 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 3.12. 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 there is an input along the trace that does not belong to 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 3.13. 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 3.14. 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). Let $t \in F_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})}$, then there exists t', a prefix of t and $t'' \in (\mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}})^*$, such that $t't'' \in F_{\mathcal{P}\wedge\mathcal{Q}}$. 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 $t't'' \in F_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}})$. By the properties of lifting, we see that $t't'' \in F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow (\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}))$ and $t't'' \in F_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow (\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}))$. The result then follows from noting that $\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}} = \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}$, $t'' \in (\mathcal{A}^O_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{Q}})^*$, and extension closure of $F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})$ and $F_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})$.

For the second claim, we again show the containment on inconsistent traces, as the proof for the observable traces is near identical. Let $t \in F_{\mathcal{E}(\mathcal{R})}$. Then from $\mathcal{R} \sqsubseteq_{imp} \mathcal{P}$ and $\mathcal{R} \sqsubseteq_{imp} \mathcal{Q}$ we obtain $t \in F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{R}})$ and $t \in F_{\mathcal{E}(\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{R}})$. Thus $t \in F_{\mathcal{E}(\mathcal{P})} \cap F_{\mathcal{E}(\mathcal{Q})}$ or $t \in F_{\mathcal{E}(\mathcal{P})} \cap (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{R}})$ or $t \in F_{\mathcal{E}(\mathcal{Q})} \cap (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{R}})$ or $t \in (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{R}}) \cap (T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{R}})$. The first three imply $t \in F_{\mathcal{P}\wedge\mathcal{Q}}$ (since the liftings in the second and third choices can be replaced with $\uparrow \mathcal{A}^I_{\mathcal{P}}$ and $\uparrow \mathcal{A}^I_{\mathcal{Q}}$ respectively), while the fourth possibility implies $t \in F_{\mathcal{P}\wedge\mathcal{Q}} \cup (T_{\mathcal{P}\wedge\mathcal{Q}} \uparrow \mathcal{A}^I_{\mathcal{R}})$. Hence $t \in F_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P}\wedge\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{R}})$ 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 3.15. To demonstrate conjunction, we consider a device that is capable of printing and faxing documents. The behaviour of this device is shown in Figure 3.2. Note how this device is capable of printing multiple documents after having received job_details (indicated by the self-loop labelled with print). At this stage, square nodes, and circular nodes containing •, have no significance.

The conjunction of the original multi-function device (capable of printing and scanning, shown in Figure 3.1) and this new printing/faxing device is shown in Figure 3.3. 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 3.3 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 3.1 with fax_mode and fax, and extending the set of actions of the device in Figure 3.2 with scan_mode and scan, so that the components to be conjoined have identical interfaces, then the conjunction is a component as shown in Figure 3.4. This device is capable of scanning and faxing 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 e.g. [LV07]), 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. ◇

3.1.4 Disjunction

Disjunction is the dual of conjunction, so corresponds to the join operator on the refinement pre-order. Therefore, the disjunction of a collection of components is the finest component that they

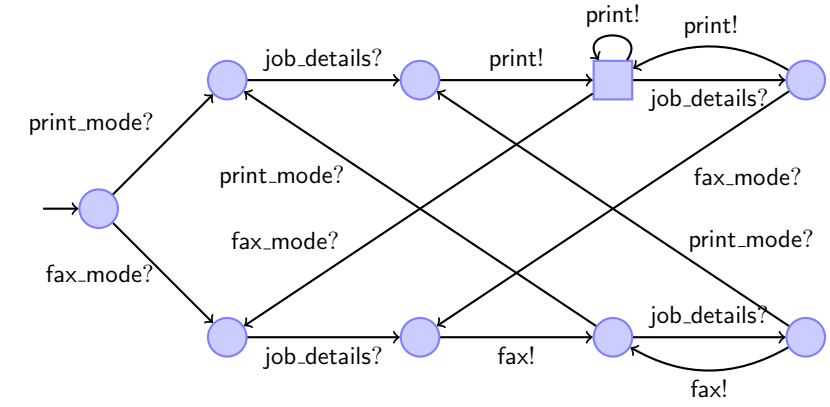

Figure 3.2: Printing/faxing device

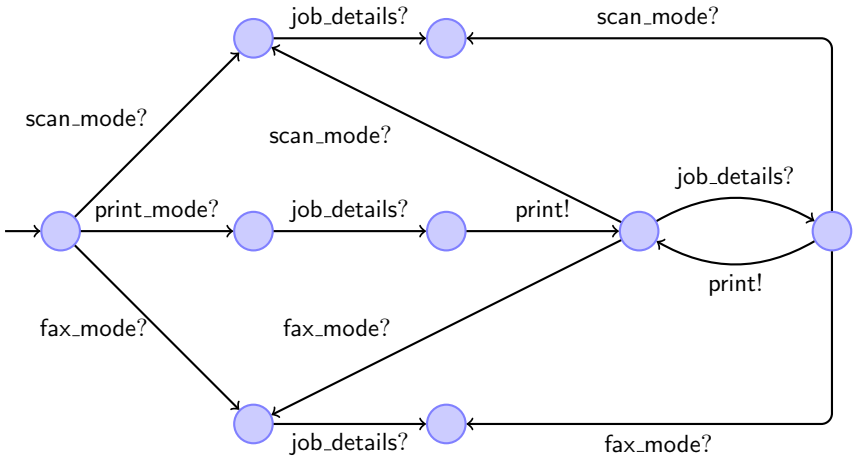

Figure 3.3: Conjunction of the printing/scanning and printing/faxing devices

each refine, meaning that an environment safe for the disjunction is an environment safe for both of its arguments. Composability of components under disjunction is the same as for conjunction.

Definition 3.16. 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:

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

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.

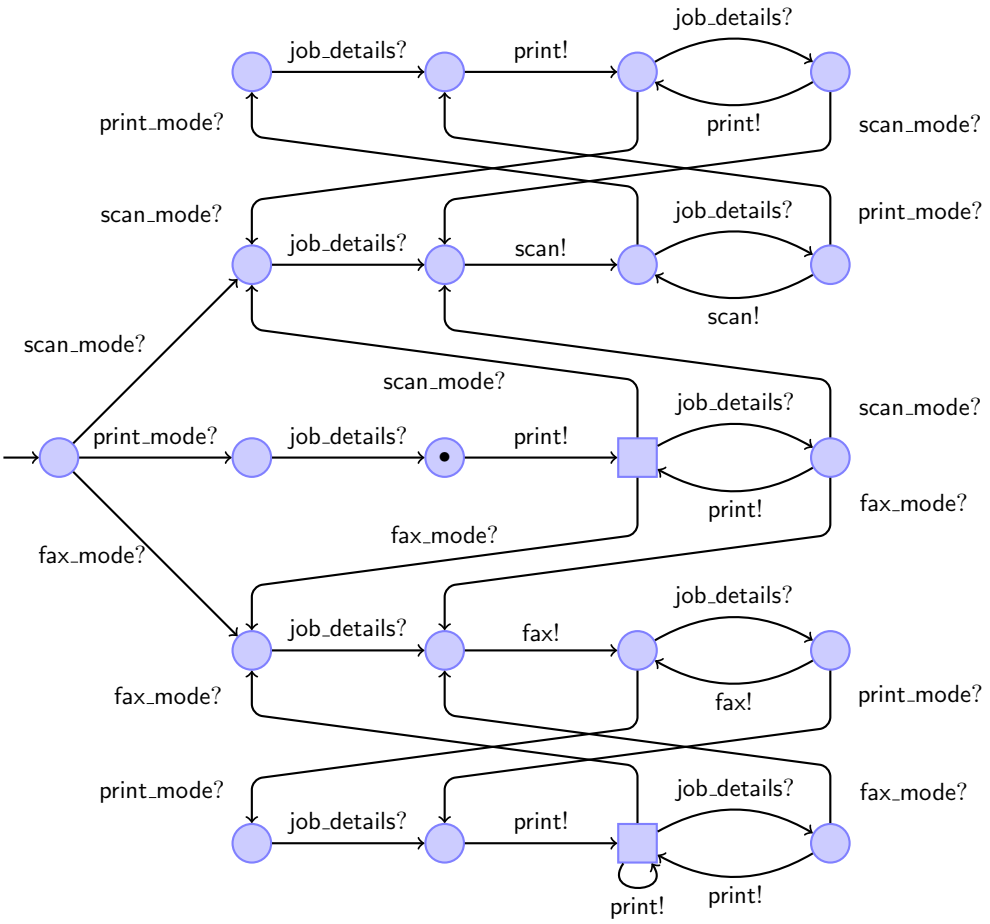

Figure 3.4: Conjunction of the printing/scanning and printing/faxing devices when the components have identical interfaces incorporating all actions

Lemma 3.17. 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.

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

□

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

Theorem 3.18. 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 $t \in F_{\mathcal{E}(\mathcal{P})}$. Then there exists a prefix t' of t and a trace $t'' \in (\mathcal{A}^O_{\mathcal{P}})^*$ such that $t't'' \in F_{\mathcal{P}}$. Now either $t't'' \in \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$, implying $t't'' \in F_{\mathcal{P}\vee\mathcal{Q}}$ and so $t \in F_{\mathcal{E}(\mathcal{P}\vee\mathcal{Q})}$, or there exists a prefix $t_1 i$ of t' with $t_1 \in \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$ and $i \in \mathcal{A}^I_{\mathcal{P}} \setminus \mathcal{A}^I_{\mathcal{Q}}$. Consequently, $t_1 \in T_{\mathcal{P}\vee\mathcal{Q}}$ and $t_1 i \in T_{\mathcal{P}\vee\mathcal{Q}} \uparrow \mathcal{A}^I_{\mathcal{P}}$. Hence $t \in T_{\mathcal{P}\vee\mathcal{Q}} \uparrow \mathcal{A}^I_{\mathcal{P}}$ as required. For observable trace containment, suppose $t \in T_{\mathcal{P}}$. Then either $t \in T_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$ or $t \in (T_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}) \uparrow \mathcal{A}^I_{\mathcal{P}}$. This means that $t \in T_{\mathcal{P}\vee\mathcal{Q}} \cup (T_{\mathcal{P}\vee\mathcal{Q}} \uparrow \mathcal{A}^I_{\mathcal{P}})$ as required. Hence $\mathcal{P} \sqsubseteq_{imp} \mathcal{P}\vee\mathcal{Q}$. Showing $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}\vee\mathcal{Q}$ is similar.

For the second claim, suppose $t \in F_{\mathcal{E}(\mathcal{P}\vee\mathcal{Q})}$. Then there exists t', a prefix of t and $t'' \in (\mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}})^*$, such that $t't'' \in F_{\mathcal{P}\vee\mathcal{Q}}$ 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 $t_p \in F_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$. From $\mathcal{P} \sqsubseteq_{imp} \mathcal{R}$, it follows that $t_p \in (F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}})) \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$ and so $t_p \in F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}})$. As $t't'' \in \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$, it follows that $t't'' \in (\mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}} \cup \mathcal{A}_{\mathcal{R}})^*$. Hence $t't'' \in F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}})$, from which we can deduce $t \in F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}})$. For the observable trace $t \in T_{\mathcal{P}\vee\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}\vee\mathcal{Q})}$, it holds without loss of generality that $t \in T_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$. From $\mathcal{P} \sqsubseteq_{imp} \mathcal{R}$ it follows that $t \in (T_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}})) \cap \mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$, and so $t \in T_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}})$ as required.

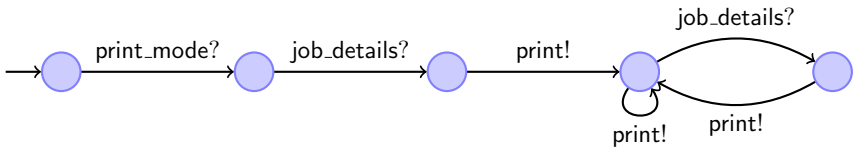

Figure 3.5: Disjunction of the printing/scanning and printing/faxing devices

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 3.19. A user wishing to use a multi-function device is non-deterministically allocated the printing/scanning device (Figure 3.1) or the printing/faxing device (Figure 3.2). 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 3.5. ◇

3.1.5 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 3.20. 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 3.21. 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 $t \in F_{\mathcal{E}(\mathcal{Q}/b)}$. Then $t \in F_{\mathcal{E}(\mathcal{Q})/b}$ and so $t \in F_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{A}^*_{\mathcal{Q}/b}$. By $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ we have $t \in (F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \upharpoonright \mathcal{A}^I_{\mathcal{Q}})) \cap \mathcal{A}^*_{\mathcal{Q}/b}$. This means that $t \in (F_{\mathcal{E}(\mathcal{P})} \cap \mathcal{A}^*_{\mathcal{P}/b}) \cup (T_{\mathcal{E}(\mathcal{P})} \cap \mathcal{A}^*_{\mathcal{P}/b})(\mathcal{A}^I_{\mathcal{Q}/b} \setminus \mathcal{A}_{\mathcal{P}/b})(\mathcal{A}_{\mathcal{P}/b} \cup \mathcal{A}^I_{\mathcal{Q}/b})^*$, implying $t \in F_{\mathcal{E}(\mathcal{P})/b} \cup (T_{\mathcal{E}(\mathcal{P})/b} \upharpoonright \mathcal{A}^I_{\mathcal{Q}/b})$. Hence $t \in F_{\mathcal{E}(\mathcal{P}/b)} \cup (T_{\mathcal{E}(\mathcal{P}/b)} \upharpoonright \mathcal{A}^I_{\mathcal{Q}/b})$ 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 $t \in F_{\mathcal{E}(\mathcal{Q}/b)}$, 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 from $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ it follows that $t' \in F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \upharpoonright \mathcal{A}^I_{\mathcal{Q}})$. If $t' \in F_{\mathcal{E}(\mathcal{P})}$, then $t' \in \mathcal{A}^*_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{Q}}$, which implies $t' \upharpoonright \mathcal{A}_{\mathcal{P}/b} = t$. Hence $t \in F_{\mathcal{E}(\mathcal{P})/b}$, which implies $t \in F_{\mathcal{E}(\mathcal{P}/b)}$. If $t' \notin F_{\mathcal{E}(\mathcal{P})}$, then there is $t''at''' \equiv t'$ such that $t'' \in T_{\mathcal{E}(\mathcal{P})}$, $a \in \mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}_{\mathcal{P}}$ and $t''' \in (\mathcal{A}_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}})^*$. From this it follows that $t'' \upharpoonright \mathcal{A}_{\mathcal{P}/b} \in T_{\mathcal{E}(\mathcal{P})/b}$ and $a \in \mathcal{A}^I_{\mathcal{Q}/b} \setminus \mathcal{A}_{\mathcal{P}/b}$, which in conjunction with the fact that $t''a \upharpoonright \mathcal{A}_{\mathcal{P}/b}$ is a prefix of t implies $t''a \upharpoonright \mathcal{A}_{\mathcal{P}/b} \in T_{\mathcal{E}(\mathcal{P})/b} \upharpoonright \mathcal{A}^I_{\mathcal{Q}/b}$. The remaining extension of $t''a \upharpoonright \mathcal{A}_{\mathcal{P}/b}$ to $t \in \mathcal{A}^*_{\mathcal{Q}/b}$ (not necessarily $t''' \upharpoonright \mathcal{A}_{\mathcal{P}/b}$) is certainly contained in $(\mathcal{A}_{\mathcal{P}/b} \cup \mathcal{A}^I_{\mathcal{Q}/b})^*$, which implies $t \in T_{\mathcal{E}(\mathcal{P})/b} \upharpoonright \mathcal{A}^I_{\mathcal{Q}/b}$ and so $t \in T_{\mathcal{E}(\mathcal{P}/b)} \upharpoonright \mathcal{A}^I_{\mathcal{Q}/b}$ as required. The T-set containment is similar. □

Example 3.22. Disaster strikes and the Device becomes broken such that it will no longer scan documents (depicted as BrokenDevice in Figure 3.6). As a result, the BrokenDevice should not be placed in scan_mode. The updated behaviour of the device is given by BrokenDevice/scan_mode,

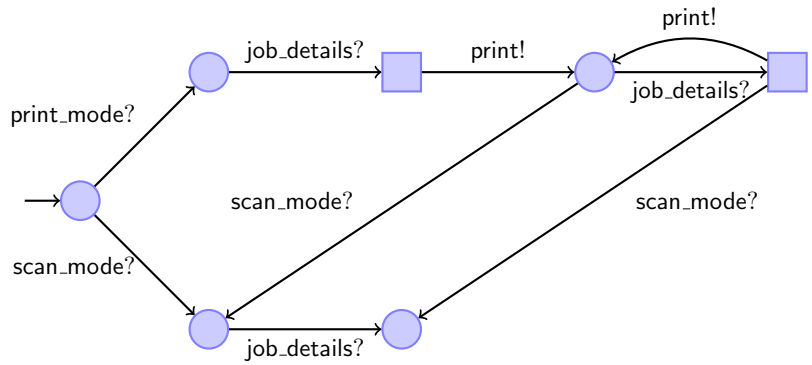

Figure 3.6: Broken device unable to scan (BrokenDevice)

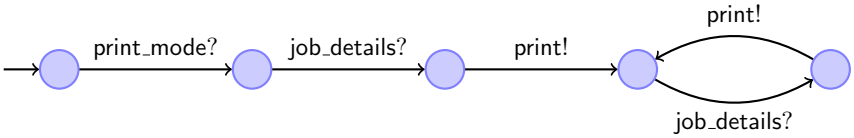

Figure 3.7: Hiding scan_mode in the broken device

as shown in Figure 3.7. The resulting component model contracts the interface of the BrokenDevice by being indifferent to scan_mode requests. ◇

3.1.6 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 upper/right adjoint of parallel composition in an appropriate Galois connection, as made explicit later on.

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

Definition 3.23. 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
  $\{t \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_\mathcal{R} \cdot t' \upharpoonright \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P} \implies t' \in T_{\mathcal{E}(\mathcal{R})}\} \cap$

$\{t \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \restriction \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \restriction \mathcal{A}_{\mathcal{P}} \in F_{\mathcal{P}} \implies t' \in F_{\mathcal{E}(\mathcal{R})}\}$

- $F_{\mathcal{R}/\mathcal{P}} = \{t \in T_{\mathcal{R}/\mathcal{P}} : \forall t' \in \mathcal{A}^*_{\mathcal{R}} \cdot t' \restriction \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t \text{ and } t' \restriction \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}} \implies t' \in F_{\mathcal{E}(\mathcal{R})}\}$. $\diamond$

Before explaining the intuition behind the definition, we first show that $\mathcal{R}/\mathcal{P}$ really is a component, since this is not evident from the formulations of $T_{\mathcal{R}/\mathcal{P}}$ and $F_{\mathcal{R}/\mathcal{P}}$.

Lemma 3.24. The quotient operation yields a component.

Proof. Clearly $T_{\mathcal{R}/\mathcal{P}}$ is prefix-closed and input-receptive, and $F_{\mathcal{R}/\mathcal{P}} \subseteq T_{\mathcal{R}/\mathcal{P}}$ by definition. To show extension closure of $F_{\mathcal{R}/\mathcal{P}}$, suppose $t \in F_{\mathcal{R}/\mathcal{P}}$. Then for each $t' \in \mathcal{A}^*_{\mathcal{R}}$ such that $t' \restriction \mathcal{A}_{\mathcal{R}/\mathcal{P}} = t$, it follows that $t' \restriction \mathcal{A}_{\mathcal{P}} \notin T_{\mathcal{P}}$ or $t' \in F_{\mathcal{E}(\mathcal{R})}$. These conditions are satisfied by any extension of t' with words in $\mathcal{A}^*_{\mathcal{R}}$ (as are the conditions for $T_{\mathcal{R}/\mathcal{P}}$), thus any extension of t by words in $\mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$ is also in $F_{\mathcal{R}/\mathcal{P}}$. $\square$

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, and 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 $\mathcal{P} \parallel (\mathcal{R}/\mathcal{P})$, hence it should be inconsistent in the quotient (so that we obtain the least refined solution). These two conditions correlate with $t \restriction \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}} \implies t \in F_{\mathcal{E}(\mathcal{R})}$ 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 $t \restriction \mathcal{A}_{\mathcal{P}} \in F_{\mathcal{P}} \implies t \in F_{\mathcal{E}(\mathcal{R})}$ and $t \restriction \mathcal{A}_{\mathcal{P}} \in T_{\mathcal{P}} \implies t \in T_{\mathcal{E}(\mathcal{R})}$.

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 3.25. Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components. Then $\mathcal{P} \parallel \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$ iff:

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

Proof. For the first claim, if $\mathcal{P} \parallel \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$, then $\mathcal{A}^O_{\mathcal{P}\parallel\mathcal{Q}} \subseteq \mathcal{A}^O_{\mathcal{R}}$. As $\mathcal{A}^O_{\mathcal{P}\parallel\mathcal{Q}} = \mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}}$, 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 $\mathcal{P} \parallel \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$.

For the second claim, let $t \in T_{\mathcal{E}(\mathcal{P}\parallel(\mathcal{R}/\mathcal{P}))}$ such that $t \notin \mathcal{A}^*_{\mathcal{R}}$, and assume that all strict prefixes of t are in $T_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\parallel(\mathcal{R}/\mathcal{P})})$. Then there is a prefix $t'a$ of t such that $t' \in \mathcal{A}^*_{\mathcal{R}}$ and $a \in \mathcal{A}^I_{\mathcal{P}} \setminus \mathcal{A}_{\mathcal{R}}$. Therefore, $t'a \in T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\parallel(\mathcal{R}/\mathcal{P})}$, which implies $t \in T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\parallel(\mathcal{R}/\mathcal{P})}$ by extension closure. In the case that $t \in \mathcal{A}^*_{\mathcal{R}}$, first suppose $t \in F_{\mathcal{E}(\mathcal{P}\parallel(\mathcal{R}/\mathcal{P}))}$. Then there exists a prefix t' of t and $t'' \in (\mathcal{A}^O_{\mathcal{P}\parallel(\mathcal{R}/\mathcal{P})})^*$ such that $t't'' \in F_{\mathcal{P}\parallel(\mathcal{R}/\mathcal{P})}$. Without loss of generality, suppose there is no prefix of $t't''$ in $F_{\mathcal{P}\parallel(\mathcal{R}/\mathcal{P})}$. 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 $t \in T_{\mathcal{P}\parallel(\mathcal{R}/\mathcal{P})} \setminus F_{\mathcal{E}(\mathcal{P}\parallel(\mathcal{R}/\mathcal{P}))}$. 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, let $t \in T_{\mathcal{E}(\mathcal{Q})}$. Then certainly $t \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$, since $\mathcal{A}^I_{\mathcal{Q}} = \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$ and $\mathcal{R}/\mathcal{P}$ 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 $t't'' \in \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$. 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 $\mathcal{P} \parallel \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$. 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 $t \in T_{\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{Q})}$. 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 $\mathcal{P} \parallel \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$. By the arbitrariness of t''', it follows that $t \in T_{\mathcal{R}/\mathcal{P}}$ as required. □

This definition of quotient generalises that supplied in [CCJK12] and [BR08], both of which require that the interface of $\mathcal{R}/\mathcal{P}$ synchronises with all actions of $\mathcal{P}$. Although in this dissertation 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 3.25 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 3.23 (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 $\mathcal{P} \parallel \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$ can be inferred, since parallel composition has interface restrictions for monotonicity to hold (cf Theorem 3.10).

The following corollary formalises the quotient operator as the upper/right adjoint of parallel composition by establishing an appropriate Galois connection on component spaces.

Corollary 3.26. Let $\mathcal{P}$ be a component, and let $\mathfrak{R} = \langle \mathbf{R}, \sqsubseteq_{imp} \rangle$ and $\mathfrak{Q} = \langle \mathbf{Q}, \sqsubseteq_{imp} \rangle$ be partially ordered sets where $\mathbf{R}$ consists of all components with input set $\mathcal{A}^I_{\mathbf{R}}$ and output set $\mathcal{A}^O_{\mathbf{R}}$, and $\mathbf{Q}$ consists of all components with input set $\mathcal{A}^I_{\mathbf{R}} \setminus \mathcal{A}^I_{\mathcal{P}}$ and output set $\mathcal{A}^O_{\mathbf{R}} \setminus \mathcal{A}^O_{\mathcal{P}}$. Then for each $\mathcal{R} \in \mathbf{R}$ and each $\mathcal{Q} \in \mathbf{Q}$, it holds that $\mathcal{P} \parallel \mathcal{Q} \sqsubseteq_{imp} \mathcal{R}$ iff $\mathcal{Q} \sqsubseteq_{imp} \mathcal{R}/\mathcal{P}$. Hence $\langle F_{\mathcal{P}} : \mathbf{Q} \longrightarrow \mathbf{R}, G_{\mathcal{P}} : \mathbf{R} \longrightarrow \mathbf{Q} \rangle$ forms a Galois connection between $\mathfrak{R}$ and $\mathfrak{Q}$ for every component $\mathcal{P}$, where $F_{\mathcal{P}}(X) \triangleq \mathcal{P} \parallel X$ and $G_{\mathcal{P}}(X) \triangleq X/\mathcal{P}$. Therefore, quotient can be seen as the upper/right adjoint of parallel composition.

Proof. Follows from Theorem 3.25. □

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

Theorem 3.27. 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, $\mathcal{A}^I_{\mathcal{Q}/\mathcal{R}} = \mathcal{A}^I_{\mathcal{P}/\mathcal{R}}$ 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, $\mathcal{A}^I_{\mathcal{R}/\mathcal{Q}} = \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$ and $(\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{P}}) \cap \mathcal{A}_{\mathcal{R}} = \emptyset$, then $\mathcal{R}/\mathcal{P} \sqsubseteq_{imp} \mathcal{R}/\mathcal{Q}$.

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 3.25.

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 3.10 (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 3.25 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 3.6), 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 3.25, 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 3.28. 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 printed or scanned. This constraint is represented by the component Constraint in Figure 3.8. 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.

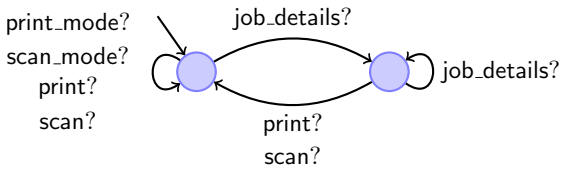

Figure 3.8: Constraint on job_details

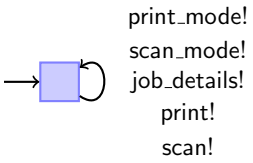

Figure 3.9: ErrorFree component

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 3.10). ErrorFree is the component represented in Figure 3.9 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, and hence User2 || Device conforms to the behaviour of Constraint. ◇

Applications of quotient to mediator synthesis were demonstrated in [IT13, BCIJ13], as remarked in Section 1.3.

3.1.7 Full Abstraction

In this section, we demonstrate that our refinement relation is the weakest preorder preserving 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 3.29. 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 $\epsilon \in F_{\mathcal{E}(\mathcal{P})}$. ◇

Theorem 3.30. 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 3.10, since the constraints for that Theorem are satisfied. Hence $\mathcal{Q} \,||\, \mathcal{R} \sqsubseteq^{F}_{imp} \mathcal{P} \,||\, \mathcal{R}$ as required.

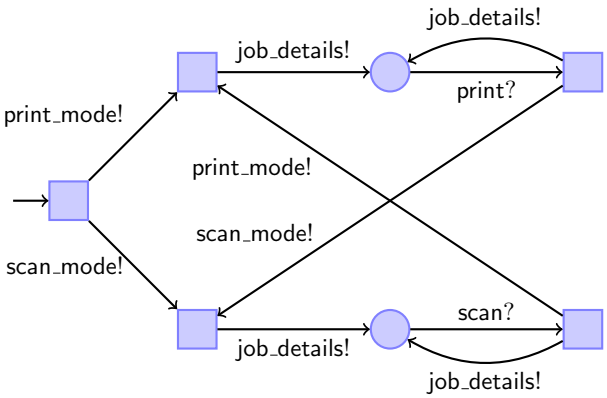

Figure 3.10: Component representing User2

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

In the case of the former, it follows (by the minimality of t) that $t \in F_{\mathcal{Q}}$. By the constraints on the alphabets, it follows that there is a maximal prefix t' of t such that $t' \in \mathcal{A}^*_{\mathcal{R}}$, and, moreover, this is the same maximal prefix such that $t' \in \mathcal{A}^*_{\mathcal{P}}$. If t' is a strict prefix of t, then by minimality of t we have $t' \in T_{\mathcal{E}(\mathcal{P})}$ and $t \in T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}}$, since the next action after t' must be in $\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}_{\mathcal{P}}$, but this is contradictory. Therefore, $t' = t$, which means we can construct an $\mathcal{R}$ such that $F_{\mathcal{R}} = \emptyset$ and $T_{\mathcal{R}}$ is the smallest set containing t that makes $\mathcal{R}$ a component. Now $t \in T_{\mathcal{R}}$ implies $t \in F_{\mathcal{Q}||\mathcal{R}}$, and so $\epsilon \in F_{\mathcal{E}(\mathcal{Q}||\mathcal{R})}$ given $t \in (\mathcal{A}^O_{\mathcal{Q}||\mathcal{R}})^*$. 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 $t \in T_{\mathcal{Q}}$. Again, there is a maximal prefix t' of t such that $t' \in \mathcal{A}^*_{\mathcal{R}}$, and, moreover, this is the same maximal prefix contained in $\mathcal{A}^*_{\mathcal{P}}$. If t' is a strict prefix, then the next symbol after t' is an element of $\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}_{\mathcal{P}}$. Hence, by minimality of t, it follows that $t \in T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}}$, but this is contradictory. Therefore, we know $t' = t$, 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 3.30 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 $\mathcal{A}^I_{\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$.

From this characterisation of $\sqsubseteq_{imp}$, 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 [vG94] (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 3.31. 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 3.10, 3.14, 3.18, 3.21 and 3.27. Consequently, $\equiv_{imp}$ is a congruence for all of the compositional operators. Taking this along with Theorem 3.30 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 $\equiv_{imp}$.

3.2 A Progress Sensitive Theory of Substitutable Components

A perceived shortcoming of interface automata (and hence our theory in Section 3.1) 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 [Hoa85], 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 3.1.1 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 3.1.1, 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 [Seg97, RV96], is motivated by the desire to utilise only finite-length traces, as in Section 3.1. 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 [Hoa85], 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 3.32. 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 3.4, 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 $\{t \in T_\mathcal{P} : \nexists o \in \mathcal{A}^O_\mathcal{P} \cdot to \in T_\mathcal{P}\} \cup D_\mathcal{P} \subseteq K_\mathcal{P} \subseteq T_\mathcal{P}$. ◇

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, e.g., the receptive process theory [Jos92] and the work of [Jon91].

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

3.2.1 Refinement

As in Section 3.1.1, refinement of component $\mathcal{Q}$ by component $\mathcal{P}$ needs to consider the 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 3.33. $\mathcal{Q}$ is said to be a *progress-sensitive refinement* of $\mathcal{P}$, written $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{P}$, iff:

- $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$
- $D_{\mathcal{E}(\mathcal{Q})} \subseteq D_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$
- $K_{\mathcal{E}(\mathcal{Q})} \subseteq K_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$. ◇

By $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ we mean refinement as in Definition 3.4 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 $D_{\mathcal{E}(\mathcal{Q})} \subseteq D_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$ and $K_{\mathcal{E}(\mathcal{Q})} \subseteq K_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$ 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 $\equiv^l_{imp}$, can easily be defined by means of mutual refinement, i.e., is equal to $\sqsubseteq^l_{imp} \cap (\sqsubseteq^l_{imp})^{-1}$.

Lemma 3.34. Progress-sensitive refinement is reflexive, and transitive subject to preservation of action types.

Proof. Follows by the exact same reasoning as in Lemma 3.6. □

3.2.2 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 3.35. 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 3.36. 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 3.10, 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 3.10) 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. □

3.2.3 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 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 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 3.37. 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
  $\{t \in T_{\mathcal{P}\wedge\mathcal{Q}} : \exists t' \in (\mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})^* \cdot tt' \notin K_{\mathcal{P}\wedge\mathcal{Q}} \text{ and } \forall o \in \mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}} \cdot tt'o \notin T_{\mathcal{P}\wedge\mathcal{Q}} \setminus Err\} \cdot \mathcal{A}^*_{\mathcal{P}\wedge\mathcal{Q}}$.

◇

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 3.38. 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 3.14. 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 3.14, 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 3.14 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 3.14, 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 3.39. 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 3.1 with the printing and faxing device of Figure 3.2, 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 3.4 (note that the D set is empty, since there are no τ transitions).

The upper non-quiescent state in Figure 3.4 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 state containing • does not need to be removed, since this state is permitted 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 3.11. ◇

3.2.4 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 3.40. 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:

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

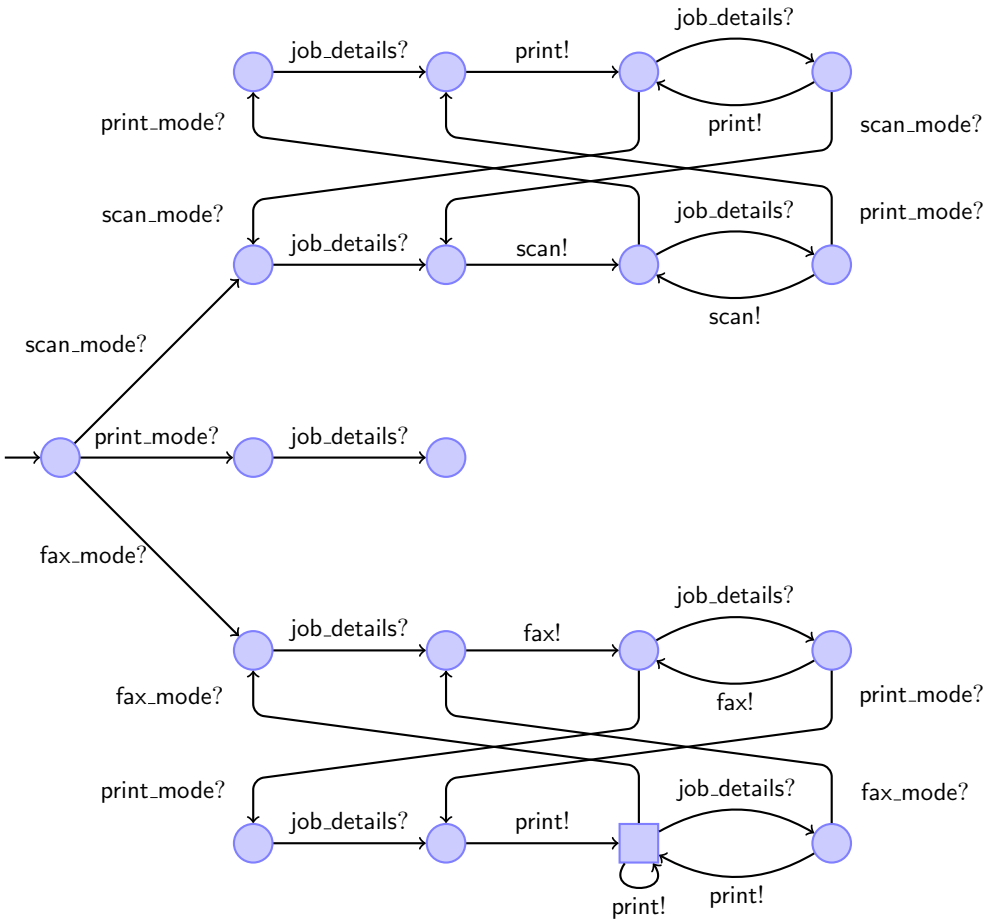

Figure 3.11: Progress-sensitive conjunction of the printing/scanning and printing/faxing devices when the components have identical interfaces incorporating all actions

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

Theorem 3.41. 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 3.18. The divergent and quiescent trace containment proofs are identical to showing containment of the observable traces. □

3.2.5 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 3.42. 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:

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

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 [Hoa85]).

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

Proof. The divergent and quiescent trace containments follow by the same reasoning as in Theorem 3.21 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 $t \in D_{\mathcal{Q}/b}$. Then there exists $t' \in T_{\mathcal{Q}}$ 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 $t' \in D_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})$. Hence $t \in D_{\mathcal{E}(\mathcal{P}/b)} \cup (T_{\mathcal{E}(\mathcal{P}/b)} \uparrow \mathcal{A}^I_{\mathcal{Q}})$ as required. In the case of the latter, $t' \in T_{\mathcal{Q}}$ implies $t' \in T_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})$. The difficult case is when $t' \in T_{\mathcal{E}(\mathcal{P})}$, from which we can deduce $t'b^i \in T_{\mathcal{E}(\mathcal{P})}$ for each $i \in \mathbb{N}$. Hence $t \in D_{\mathcal{E}(\mathcal{P}/b)}$. Quiescent trace containment is similar. □

3.2.6 Quotient

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

QC1. 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.

QC2. Traces of $\mathcal{R}/\mathcal{P}$ that introduce new quiescence conflicts, after having repeatedly removed traces satisfying this or the previous condition.

Definition 3.44. 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 $\mathcal{R} \mathbin{/_l} \mathcal{P}$ 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
  $\{t \in T^l_{\mathcal{R}/\mathcal{P}} : \exists t' \in (\mathcal{A}^I_{\mathcal{R}/\mathcal{P}})^* \cdot tt' \notin K^l_{\mathcal{R}/\mathcal{P}} \text{ and } \forall o \in \mathcal{A}^O_{\mathcal{R}/\mathcal{P}} \cdot tt'o \notin T^l_{\mathcal{R}/\mathcal{P}} \setminus Err\} \cdot \mathcal{A}^*_{\mathcal{R}/\mathcal{P}}$. $\diamond$

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 3.45. 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 3.25. For the second claim, inconsistent and observable trace containment follows by Theorem 3.25, 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 $t \in D_{\mathcal{P}||_l(\mathcal{R}/_l\mathcal{P})} \setminus F_{\mathcal{E}(\mathcal{P}||_l(\mathcal{R}/_l\mathcal{P}))}$ and for the difficult case $t \in \mathcal{A}^*_{\mathcal{R}}$. 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 $t \in D_{\mathcal{E}(\mathcal{R})}$. For

the quiescent containment, suppose $t \in K_{\mathcal{P}||_l(\mathcal{R}/_l\mathcal{P})}$ and $t \in \mathcal{A}^*_{\mathcal{R}}$. 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 3.25, 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 $T_{\mathcal{E}(\mathcal{Q})} \cap Err = \emptyset$. □

By the same reasoning as in Corollary 3.26, it can be shown that quotient is the upper/right adjoint of parallel composition, by formulating an appropriate Galois connection. The result follows effortlessly from Theorem 3.45.

Theorem 3.46. 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, $\mathcal{A}^I_{\mathcal{P}/\mathcal{R}} = \mathcal{A}^I_{\mathcal{Q}/\mathcal{R}}$ 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, $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^I_{\mathcal{R}/\mathcal{Q}}$ and $(\mathcal{A}^I_{\mathcal{Q}} \setminus \mathcal{A}^I_{\mathcal{P}}) \cap \mathcal{A}_{\mathcal{R}} = \emptyset$, then $\mathcal{R} /_l \mathcal{P} \sqsubseteq^l_{imp} \mathcal{R} /_l \mathcal{Q}$.

Proof. The proof is the same as in Theorem 3.27 when using Theorems 3.36 and 3.45 in place of Theorems 3.10 and 3.25, and Lemma 3.34 in place of Lemma 3.6. □

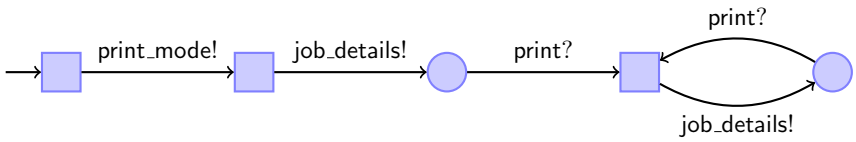

Figure 3.12: Component representing User3

Example 3.47. To demonstrate quotient in the quiescent framework, suppose that a user wishes to interact with BrokenDevice (Figure 3.6), 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 3.10) 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 3.12. ErrorFree is the component having chaotic behaviour over all actions, which we treat as outputs (Figure 3.9). 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 $\langle$scan_mode, job_details$\rangle$ is quiescent, but is required to make progress. We therefore remove the trace $\langle$scan_mode, job_details$\rangle$ from 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 3.12.

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 and subsequently removed. Consequently, all states would have to be pruned, meaning that no safe user, ensuring progress, can exist. $\diamond$

3.2.7 Full Abstraction

In Section 3.1.7 we provided a full abstraction result for substitutive equivalence that demonstrates $\mathcal{P} \equiv_{imp} \mathcal{Q}$ by checking that inconsistency is equi-reachable[1] in both $\mathcal{P}$ and $\mathcal{Q}$ under each environment. This is generalised to the progress-sensitive setting by additionally checking that quiescence is equi-reachable in both $\mathcal{P}$ and $\mathcal{Q}$, providing that $\mathcal{P}$ and $\mathcal{Q}$ are divergent-free components. To our knowledge, there is no *basic* preorder for which a full abstraction result can be provided when components include divergence, unless divergence is equated with inconsistency. For simplicity, as in other frameworks, we assume that $\mathcal{P}$ and $\mathcal{Q}$ have the same interface. We now define the basic preorder that checks for inconsistency and quiescence.

Definition 3.48. Let $\mathcal{P}$ and $\mathcal{Q}$ be components. Then $\mathcal{Q}$ is *inconsistency and progress substitutable* for $\mathcal{P}$, denoted by $\mathcal{Q} \sqsubseteq_{imp}^{F,l} \mathcal{P}$, iff:

- $\mathcal{Q}$ can become inconsistent implies $\mathcal{P}$ can become inconsistent (i.e., $\epsilon \in F_{\mathcal{E}(\mathcal{Q})}$ implies $\epsilon \in F_{\mathcal{E}(\mathcal{P})}$); and
- $\mathcal{Q}$ can become quiescent implies $\mathcal{P}$ can become quiescent. ◇

Based on this basic preorder, we now show that progress-sensitive refinement can be cast in terms of contextual checking of $\sqsubseteq_{imp}^{F,l}$ under every environment.

Theorem 3.49. Let $\mathcal{P}$ and $\mathcal{Q}$ be divergent-free components such that $\delta \notin \mathcal{A}_{\mathcal{P}} \cup \mathcal{A}_{\mathcal{Q}}$. Then:

$$\mathcal{Q} \sqsubseteq_{imp}^{l} \mathcal{P} \text{ iff } \forall \mathcal{R} \cdot \mathcal{A}_{\mathcal{R}}^{O} = \mathcal{A}_{\mathcal{P}}^{I} \cup \{\delta\} \text{ and } \mathcal{A}_{\mathcal{R}}^{I} = \mathcal{A}_{\mathcal{P}}^{O} \implies \mathcal{Q} \,||\, \mathcal{R} \sqsubseteq_{imp}^{F,l} \mathcal{P} \,||\, \mathcal{R}.$$

Proof. For the only if direction, note that the addition of δ to the interface of $\mathcal{R}$ cannot prevent $\mathcal{Q} \,||\, \mathcal{R} \sqsubseteq_{imp}^{F} \mathcal{P} \,||\, \mathcal{R}$ from holding, and similarly, for the if direction, it must be the case that $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ holds (for the substitutive conditions).

Now for the only if direction, suppose that $\mathcal{Q} \sqsubseteq_{imp}^{l} \mathcal{P}$, and there exists $\mathcal{R}$ such that $\mathcal{Q} \,||\, \mathcal{R}$ is quiescent on a trace t, while $\mathcal{P} \,||\, \mathcal{R}$ is not quiescent on t. Then clearly t must be a trace of $\mathcal{P} \,||\, \mathcal{R}$ by Theorem 3.30. If t is quiescent in $\mathcal{Q} \,||\, \mathcal{R}$ because it is divergent, then t must be divergent in $\mathcal{R}$, which implies t is divergent in $\mathcal{P} \,||\, \mathcal{R}$, and so we reach a contradiction. Therefore, $\mathcal{Q} \,||\, \mathcal{R}$ cannot diverge, meaning that both $\mathcal{Q}$ and $\mathcal{R}$ must be quiescent. But from $\mathcal{Q}$ being quiescent it follows that $\mathcal{P}$ is quiescent, hence $\mathcal{P} \,||\, \mathcal{R}$ is quiescent. This is also contradictory, so $\mathcal{R}$ cannot exist.

For the if direction, suppose there exists a trace t that is quiescent in $\mathcal{Q}$ and non-quiescent in $\mathcal{P}$. Then we can construct an environment $\mathcal{R}$ that is the least prefix-closed and input-receptive set containing the trace t such that any trace of $\mathcal{R}$ (excluding t) can be extended by δ and is non-quiescent. Then, because t is quiescent in $\mathcal{Q}$, it follows that t is a quiescent trace of $\mathcal{Q} \,||\, \mathcal{R}$.

[1] I.e., is reachable in both, or is reachable in neither.

However, as t is non-quiescent in $\mathcal{P}$, every trace of $\mathcal{P} \parallel \mathcal{R}$ must be non-quiescent. Therefore, $\mathcal{Q} \parallel \mathcal{R} \not\sqsubseteq^{F,l}_{imp} \mathcal{P} \parallel \mathcal{R}$ as required. □

The only way of distinguishing divergence in our theory is to check: t diverges in $\mathcal{Q} \parallel \mathcal{R}$ implies t diverges in $\mathcal{P} \parallel \mathcal{R}$, for each trace t. However, such a check is not basic enough for full abstraction, as it is essentially unchanged from the containment check on divergence performed as part of $\sqsubseteq^{l}_{imp}$. This is a consequence of environments not being able to suppress the behaviour of divergence nor the quiescence it can introduce.

To obtain a full abstraction result, it is necessary to equate divergence with inconsistency (as in other frameworks, such as the receptive process theory [Jos92]); however, there is a good reason for not doing that. A formalism should be guided by its suitability to modelling systems, rather than the algebraic properties that can be forced out of it. As in the substitutive framework, divergence cannot make anything bad happen, as it is non-observable, which is why we do not equate it with inconsistency. The only effect that divergence can have is to prevent progress being made. During refinement, divergent behaviours can be removed, to leave progress-making behaviours. If divergence is equated with inconsistency, then divergent behaviours can be refined into arbitrary interactive behaviour, which can be highly undesirable.

We now remark that, under the assumption of divergence-freedom, progress-sensitive equivalence is fully abstract with respect to contextual observation of inconsistency and quiescence for the specification theory.

Corollary 3.50. Progress-sensitive equivalence $\equiv^{l}_{imp}$ is fully abstract for parallel composition, conjunction, disjunction and quotient with respect to observational equivalence of inconsistency and quiescence, subject to divergence-freedom.

Proof. Based on Theorem 3.49 and the congruence results throughout Section 3.2. □

3.3 Summary

This chapter has developed a compositional specification theory capable of modelling componentised systems with asynchronous communication, such as hardware circuits and distributed software systems. Components are modelled in an abstract manner by means of traces. The formalism highlights the algebraic properties of the compositional operators, which include parallel composition, conjunction, disjunction, hiding and quotient. Two linear-time refinement relations are provided, which are the weakest preorders for substitutivity and progress-sensitive substitutivity of components respectively. Based on these, a full abstraction result is provided by showing that substitutive equivalence forms a congruence with respect to the operations defined on the specification theory.

The trace-based nature of the formalism captures essential information for inferring substitutability of components; however, it is not very amenable to modelling in a practical sense, due to the difficulty of specifying behaviours in terms of trace sets. In Chapter 4, the theory is concretised by providing an operational representation that closely mirrors actual implementations.

Contrasting with related work, the definition of a component in the substitutive setting matches that of the prefix-closed trace structures of Dill [Dil88], although our definition was derived independently (based on Logic LTSs [LV07]). The safe representation of a component in our framework is referred to as a canonical prefix-closed trace structure by Dill, which allows conformance (corresponding to our refinement) to be defined in terms of trace containment (although Dill requires equality of interfaces, whereas we have a covariant inclusion on inputs and contravariant inclusion on outputs). While Dill considers the operations of parallel composition and hiding on prefix closed trace structures, he does not consider the arguably more interesting operations of conjunction, disjunction and quotient, which support independent and incremental development.

Our progress-sensitive framework shares similarities with the receptive process theory of Josephs [Jos92]. A receptive process captures the failures and divergences of the component that it models. Divergences are used to encode undesirable behaviour (i.e., inconsistencies), while the failures include the divergent and quiescent behaviours. As the receptive process theory is built on the failures-divergences model of CSP [Hoa85], refinement is similar to our progress-sensitive preorder, in that it consists of divergence and failure containment. However, a notable exception is that inconsistencies are not propagated backwards over outputs in the receptive process theory, meaning that the refinement distinguishes too many components, and our theory disambiguates inconsistency from divergence, since the latter does not affect substitutivity. As a consequence of the receptive process theory's formulation, it is not possible to distinguish the component that can repeatedly produce any output from an inconsistent component, whereas our theory is able to distinguish such behaviours because we record more information. As a final remark, conjunction, disjunction and quotient are not defined for the receptive process theory, although a definition of quotient has been formulated for delay-insensitive processes [JK07].

CHAPTER

FOUR

Operational Theory of Components

In this chapter, we outline an operational representation for components, and demonstrate the relationship between these operational models and the trace-based models of Sections 3.1 and 3.2. Based on this, we supply operational definitions for the compositional operators of our theory (Sections 4.2-4.6), and demonstrate that the full abstraction result continues to hold (Section 4.7). A comparison of our operational framework with the interface automata of de Alfaro and Henzinger [dAH05] is provided in Section 4.8. In particular, we show that our refinement relation is weaker than the classical alternating simulation defined on interface automata, except when components are deterministic, in which case they coincide.

Definition 4.1. An *operational component* P is a tuple $\langle \mathcal{A}^I_\mathsf{P}, \mathcal{A}^O_\mathsf{P}, S_\mathsf{P}, \rightarrow_\mathsf{P}, s^0_\mathsf{P}, \perp_\mathsf{P} \rangle$, where:

- $\mathcal{A}^I_\mathsf{P}$ is a finite set of visible input actions (excluding the hidden action τ)
- $\mathcal{A}^O_\mathsf{P}$ is a finite set of visible output actions (excluding the hidden action τ), disjoint from $\mathcal{A}^I_\mathsf{P}$, where $\mathcal{A}_\mathsf{P} \triangleq \mathcal{A}^I_\mathsf{P} \cup \mathcal{A}^O_\mathsf{P}$
- S_P is a finite set of states
- $\rightarrow_\mathsf{P} \subseteq S_\mathsf{P} \times (\mathcal{A}_\mathsf{P} \cup \{\tau\}) \times S_\mathsf{P}$ is the transition relation
- $s^0_\mathsf{P} \in S_\mathsf{P}$ is the designated initial state
- $\perp_\mathsf{P} \in S_\mathsf{P}$ is the designated inconsistent state.

The transition relation satisfies the properties that: (i) $\perp_\mathsf{P} \xrightarrow{a}_\mathsf{P} \perp_\mathsf{P}$ for each $a \in \mathcal{A}_\mathsf{P} \cup \{\tau\}$; 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 state is chaotic. $\diamond$

It is important that the set of states S_P is finite, so that divergence of a state can be determined in finite time (a state is said to be divergent if it has an infinite sequence of τ-transitions emanating from it). This allows us to decide which inputs are safe, and which outputs may eventually be issued, for a particular state.

Notation For a compositional operator $\oplus$, and sets A and B, we write $A \oplus B$ for the set $\{a \oplus b : a \in A \text{ and } b \in B\}$. A relation $\stackrel{\epsilon}{\Longrightarrow}_{\mathsf{P}} \subseteq S_{\mathsf{P}} \times S_{\mathsf{P}}$ is defined by $p \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{P}} p'$ iff $p(\stackrel{\tau}{\rightarrow}_{\mathsf{P}})^* p'$. Generalising $\stackrel{\epsilon}{\Longrightarrow}_{\mathsf{P}}$ for visible actions $a \in \mathcal{A}$, we obtain $p \stackrel{a}{\Longrightarrow}|_{\mathsf{P}} p'$ iff there exists p_a such that $p \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{P}} p_a \stackrel{a}{\rightarrow}_{\mathsf{P}} p'$, and $p \stackrel{a}{\Longrightarrow}_{\mathsf{P}} p'$ iff there exists p_a such that $p \stackrel{a}{\Longrightarrow}|_{\mathsf{P}} p_a \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{P}} p'$. The extension to words $w = a_1 \ldots a_n$ is defined in the natural way by $p \stackrel{w}{\Longrightarrow}_{\mathsf{P}} p'$ iff $p \stackrel{a_1}{\Longrightarrow}_{\mathsf{P}} \ldots \stackrel{a_n}{\Longrightarrow}_{\mathsf{P}} p'$.

Henceforth, let P, Q and R be operational components with signatures $\langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, S_{\mathsf{P}}, \rightarrow_{\mathsf{P}}, s^0_{\mathsf{P}}, \bot_{\mathsf{P}} \rangle$, $\langle \mathcal{A}^I_{\mathsf{Q}}, \mathcal{A}^O_{\mathsf{Q}}, S_{\mathsf{Q}}, \rightarrow_{\mathsf{Q}}, s^0_{\mathsf{Q}}, \bot_{\mathsf{Q}} \rangle$ and $\langle \mathcal{A}^I_{\mathsf{R}}, \mathcal{A}^O_{\mathsf{R}}, S_{\mathsf{R}}, \rightarrow_{\mathsf{R}}, s^0_{\mathsf{R}}, \bot_{\mathsf{R}} \rangle$ 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 4.2. 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 $T_{[\![\mathsf{P}]\!]} = \{t : s^0_{\mathsf{P}} \stackrel{t}{\Longrightarrow}_{\mathsf{P}}\}$ and $F_{[\![\mathsf{P}]\!]} = \{t : s^0_{\mathsf{P}} \stackrel{t}{\Longrightarrow}_{\mathsf{P}} \bot_{\mathsf{P}}\}$. ◇

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

Definition 4.3. 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:

- $D_{[\![\mathsf{P}]\!]^l} = \{t : \exists s' \cdot s^0_{\mathsf{P}} \stackrel{t}{\Longrightarrow}_{\mathsf{P}} s' \text{ and } s' \text{ can diverge}\}$
- $K_{[\![\mathsf{P}]\!]^l} = \{t : \exists s' \cdot s^0_{\mathsf{P}} \stackrel{t}{\Longrightarrow}_{\mathsf{P}} s' \text{ and } \nexists o \in \mathcal{A}^O_{\mathsf{P}} \cup \{\tau\} \cdot s' \stackrel{o}{\rightarrow}_{\mathsf{P}}\} \cup D_{[\![\mathsf{P}]\!]^l}$. ◇

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 3.32). Note that $D_{[\![\mathsf{P}]\!]^l}$ includes all inconsistent traces, since $\bot_{\mathsf{P}}$ 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 [Hoa85].

Based on these mappings to trace-based models, we can formulate definitions of refinement on operational models.

Definition 4.4. 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 $[\![\mathsf{Q}]\!]^l \sqsubseteq^l_{imp} [\![\mathsf{P}]\!]^l$. ◇

Justification of these mappings is presented in Section 4.7. But first we present operational definitions for all of 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 4.5. Let P and Q be components composable for parallel. Then the parallel composition of P and Q is the component $\mathsf{P} \parallel \mathsf{Q} = \mathsf{P} \parallel_l \mathsf{Q} = \langle \mathcal{A}^I, \mathcal{A}^O, S, \rightarrow, s_0, \bot \rangle$, where:

- $\mathcal{A}^I = (\mathcal{A}^I_{\mathsf{P}} \cup \mathcal{A}^I_{\mathsf{Q}}) \setminus (\mathcal{A}^O_{\mathsf{P}} \cup \mathcal{A}^O_{\mathsf{Q}})$
- $\mathcal{A}^O = \mathcal{A}^O_{\mathsf{P}} \cup \mathcal{A}^O_{\mathsf{Q}}$
- $S = S_{\mathsf{P}} \parallel S_{\mathsf{Q}}$
- $\rightarrow$ is the smallest relation 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 $p \parallel q \xrightarrow{a} p' \parallel q$

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

  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 $p \parallel q \xrightarrow{a} p' \parallel q'$.

- $s_0 = s^0_{\mathsf{P}} \parallel s^0_{\mathsf{Q}}$
- $\{\bot\} = (S_{\mathsf{P}} \parallel \{\bot_{\mathsf{Q}}\}) \cup (\{\bot_{\mathsf{P}}\} \parallel S_{\mathsf{Q}})$[1]. ◇

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 4.6. Let P and Q be components composable for parallel composition. Then $[\![\mathsf{P} \parallel \mathsf{Q}]\!] = [\![\mathsf{P}]\!] \parallel [\![\mathsf{Q}]\!]$ and $[\![\mathsf{P} \parallel_l \mathsf{Q}]\!]^l = [\![\mathsf{P}]\!]^l \parallel_l [\![\mathsf{Q}]\!]^l$.

[1] $\bot$ is a state, so the formulation involving $\{\bot\}$ essentially defines an equivalence on states.

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 4.7. Let P and Q be components composable for conjunction. Then the substitutive conjunction of P and Q is a component $\mathsf{P} \wedge \mathsf{Q} = \langle \mathcal{A}^I_\mathsf{P} \cup \mathcal{A}^I_\mathsf{Q}, \mathcal{A}^O_\mathsf{P} \cap \mathcal{A}^O_\mathsf{Q}, S, \longrightarrow, s_0, \bot \rangle$, where:

- $S = S_\mathsf{P} \wedge S_\mathsf{Q}$
- $\rightarrow$ is the smallest relation satisfying the following rules:
  - C1. If $a \in \mathcal{A}_\mathsf{P} \cap \mathcal{A}_\mathsf{Q}$, $p \overset{a}{\Longrightarrow\!\!\!|}_\mathsf{P} p'$ and $q \overset{a}{\Longrightarrow\!\!\!|}_\mathsf{Q} q'$, then $p \wedge q \xrightarrow{a} p' \wedge q'$
  - C2. If $a \in \mathcal{A}^I_\mathsf{P} \setminus \mathcal{A}^I_\mathsf{Q}$ and $p \overset{a}{\Longrightarrow\!\!\!|}_\mathsf{P} p'$, then $p \wedge q \xrightarrow{a} p' \wedge \bot_\mathsf{Q}$
  - C3. If $a \in \mathcal{A}^I_\mathsf{Q} \setminus \mathcal{A}^I_\mathsf{P}$ and $q \overset{a}{\Longrightarrow\!\!\!|}_\mathsf{Q} q'$, then $p \wedge q \xrightarrow{a} \bot_\mathsf{P} \wedge q'$
  - C4. If p does not diverge and $p \xrightarrow{\tau}_\mathsf{P} p'$, then $p \wedge q \xrightarrow{\tau} p' \wedge q$
  - C5. If q does not diverge and $q \xrightarrow{\tau}_\mathsf{Q} q'$, then $p \wedge q \xrightarrow{\tau} p \wedge q'$
  - C6. If p diverges and q diverges, then $p \wedge q \xrightarrow{\tau} p \wedge q$.
- $s_0 = s^0_\mathsf{P} \wedge s^0_\mathsf{Q}$
- $\bot = \bot_\mathsf{P} \wedge \bot_\mathsf{Q}$. ◇

In contrast to the definition in [CCJK12], 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 4.8. 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 F, the smallest set defined inductively by:

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

The first two conditions capture conjunctive states $p \wedge q$ that are quiescent (through all output successors needing to be pruned, or there being no output successors) when at least one of p or q is non-quiescent. The third condition is responsible for pruning backwards over input transitions until a non-deterministic choice is offered on inputs, from which there is at least one choice that does not have to be pruned.

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 4.9. 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 $s^0_{\mathsf{P}} \wedge s^0_{\mathsf{Q}} \overset{t}{\Longrightarrow}_{\mathsf{P}\wedge\mathsf{Q}} p \wedge q$. If $s^0_{\mathsf{P}} \overset{t}{\Longrightarrow}_{\mathsf{P}} p$, then $t \in T_{[\![\mathsf{P}]\!]}$, while if $s^0_{\mathsf{P}} \overset{t}{\not\Longrightarrow}_{\mathsf{P}} p$, 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 $\forall p, q \cdot s^0_{\mathsf{P}} \wedge s^0_{\mathsf{Q}} \overset{t}{\Longrightarrow}_{\mathsf{P}\wedge\mathsf{Q}} p \wedge q$ implies $p \wedge q \in F$. This can be demonstrated in a straightforward manner using an inductive argument by approximating Err and F, which are both obtained as fixed points. $\square$

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 4.10. Let P and Q be components composable for disjunction. Then the disjunction of P and Q is the component $\mathsf{P} \vee \mathsf{Q} = \mathsf{P} \vee_l \mathsf{Q} = \langle \mathcal{A}^I_{\mathsf{P}} \cap \mathcal{A}^I_{\mathsf{Q}}, \mathcal{A}^O_{\mathsf{P}} \cup \mathcal{A}^O_{\mathsf{Q}}, S, \longrightarrow, s_0, \bot \rangle$, 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 containing $\rightarrow_{\mathsf{P}}$ and $\rightarrow_{\mathsf{Q}}$ restricted to $\mathcal{A}_{\mathsf{P}\vee\mathsf{Q}}$, and the transitions $s_0 \xrightarrow{\tau} s^0_{\mathsf{P}}$ and $s_0 \xrightarrow{\tau} s^0_{\mathsf{Q}}$
- $\{\bot\} = \{\bot_{\mathsf{P}}, \bot_{\mathsf{Q}}\}$. ◇

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

Theorem 4.11. 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 4.12. Let P be a component and let b be an action. The hiding of b from P is the component $\mathsf{P}/b = \mathsf{P} /_l b = \langle \mathcal{A}^I_{\mathsf{P}} \setminus \{b\}, \mathcal{A}^O_{\mathsf{P}} \setminus \{b\}, S_{\mathsf{P}}, \longrightarrow, s^0_{\mathsf{P}}, \bot_{\mathsf{P}} \rangle$, 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 4.13. 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 4.14. Let P and R be deterministic components such that $\mathcal{A}^O_{\mathsf{P}} \subseteq \mathcal{A}^O_{\mathsf{R}}$. Then the quotient is the component $\mathsf{R}/\mathsf{P} = \langle \mathcal{A}^I_{\mathsf{R}/\mathsf{P}}, \mathcal{A}^O_{\mathsf{R}/\mathsf{P}}, S_{\mathsf{R}/\mathsf{P}}, \rightarrow, s_0, \bot_{\mathsf{R}/\mathsf{P}} \rangle$, where:

- $\mathcal{A}^I_{\mathsf{R}/\mathsf{P}} = \mathcal{A}^I_{\mathsf{R}} \cup \mathcal{A}^O_{\mathsf{P}}$
- $\mathcal{A}^O_{\mathsf{R}/\mathsf{P}} = \mathcal{A}^O_{\mathsf{R}} \setminus \mathcal{A}^O_{\mathsf{P}}$
- $S_{\mathsf{R}/\mathsf{P}} = (S_{\mathsf{R}}/S_{\mathsf{P}}) \setminus F$
- $\rightarrow$ is the smallest relation 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 $r/p \xrightarrow{a} \bot_{\mathsf{R}/\mathsf{P}}$.
- $s_0 = s^0_{\mathsf{R}}/s^0_{\mathsf{P}}$, providing $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \notin F$, and the quotient is unrealisable otherwise
- $\{\bot_{\mathsf{R}/\mathsf{P}}\} = \{\bot_{\mathcal{E}(\mathsf{R})}\}/S_{\mathsf{P}}$
- $F \subseteq S_{\mathsf{R}}/S_{\mathsf{P}}$ is the smallest set satisfying:

  F1. If $r \neq \bot_{\mathsf{R}}$ and $p = \bot_{\mathsf{P}}$, then $r/p \in F$

  F2. 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 $r/p \in F$

  F3. If $r/p \xrightarrow{a} r'/p'$, $a \in \mathcal{A}_{\mathsf{R}} \setminus \mathcal{A}^O_{\mathsf{R}/\mathsf{P}}$ and $r'/p' \in F$, then $r/p \in F$. ◇

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 F1 and F2 capture situations where substitutivity would be violated, while F3 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 4.15. 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 $\mathsf{R} \;/_l\; \mathsf{P}$ obtained from R/P by removing states contained within the smallest F-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 $r'/p' \notin F$, then $r/p \in F$
- If $r/p \xrightarrow{a} r'/p'$, $a \in \mathcal{A}_{\mathsf{R}} \setminus \mathcal{A}^O_{\mathsf{R}/\mathsf{P}}$ and $r'/p' \in F$, then $r/p \in F$. $\diamond$

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

Theorem 4.16. 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 that $t \in T_{[\![\mathsf{R}/\mathsf{P}]\!]} \iff t \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ and $t \in F_{[\![\mathsf{R}/\mathsf{P}]\!]} \iff t \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ by induction on the length of the trace t. We use $L(t)$ as shorthand for the predicate $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in F_{[\![\mathsf{P}]\!]} \implies t \in F_{[\![\mathcal{E}(\mathsf{R})]\!]}$ and $t \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{[\![\mathsf{P}]\!]} \implies t \in T_{[\![\mathsf{R}]\!]}$.

Case $t \equiv \epsilon$. Suppose that $\epsilon \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Then by Definition 3.23, $\epsilon \in F_{\mathcal{E}([\![\mathsf{R}]\!])}$ or $\epsilon \notin T_{[\![\mathsf{P}]\!]}$. If the former holds, then $s^0_{\mathsf{R}} = \bot_{\mathcal{E}(\mathsf{R})}$, hence $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} = \bot_{\mathsf{R}/\mathsf{P}}$, meaning $\epsilon \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$. If instead $\epsilon \notin T_{[\![\mathsf{P}]\!]}$, then s^0_{P} is not defined, so $s^0_{\mathsf{R}/\mathsf{P}} = \bot_{\mathsf{R}/\mathsf{P}}$, meaning $\epsilon \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$.

Now suppose that $\epsilon \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$. Then $s^0_{\mathsf{R}/\mathsf{P}} = \bot_{\mathsf{R}/\mathsf{P}}$, so P is unrealisable or $s^0_{\mathsf{R}} = \bot_{\mathcal{E}(\mathsf{R})}$. If the former holds, then $\epsilon \notin T_{[\![\mathsf{P}]\!]}$, hence $\epsilon \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. If instead $s^0_{\mathsf{R}} = \bot_{\mathcal{E}(\mathsf{R})}$, then $\epsilon \in F_{\mathcal{E}([\![\mathsf{R}]\!])}$, hence $\epsilon \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

Suppose that $\epsilon \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Then for all $t' \in (\mathcal{A}_{\mathsf{R}} \setminus \mathcal{A}^O_{\mathsf{R}/\mathsf{P}})^*$, $L(t')$ holds. So $s^0_{\mathsf{R}/\mathsf{P}}$ is defined and $s^0_{\mathsf{R}/\mathsf{P}} \xRightarrow{t'}_{\mathsf{R}/\mathsf{P}} s_{\mathsf{R}}/s_{\mathsf{P}}$ implies $s_{\mathsf{R}}/s_{\mathsf{P}} \notin F$. Hence $\epsilon \in T_{[\![\mathsf{R}/\mathsf{P}]\!]}$.

Now suppose that $\epsilon \in T_{[\![\mathsf{R}/\mathsf{P}]\!]}$. Then for all $t' \in (\mathcal{A}_{\mathsf{R}} \setminus \mathcal{A}^O_{\mathsf{R}/\mathsf{P}})^*$, if $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \xRightarrow{t'}_{\mathsf{R}/\mathsf{P}} s_{\mathsf{R}}/s_{\mathsf{P}}$, then $s_{\mathsf{R}}/s_{\mathsf{P}} \notin F$. Hence $t' \upharpoonright \mathcal{A}_{\mathsf{P}} \notin F_{[\![\mathsf{P}]\!]}$ or $t' \in F_{\mathcal{E}(\mathsf{R})}$ since $s_{\mathsf{R}}/s_{\mathsf{P}} \notin F$, and moreover, $t' \in T_{\mathcal{E}(\mathsf{R})}$ as $s^0_{\mathsf{R}} \xRightarrow{t'}_{\mathsf{R}} s_{\mathsf{R}}$. Hence $L(t')$ holds. If $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \not\xRightarrow{t'}_{\mathsf{R}/\mathsf{P}}$, then it follows that $s^0_{\mathsf{P}} \not\xRightarrow{t' \upharpoonright \mathcal{A}_{\mathsf{P}}}_{\mathsf{P}} s_{\mathsf{P}}$, since if $s^0_{\mathsf{P}} \xRightarrow{t' \upharpoonright \mathcal{A}_{\mathsf{P}}}_{\mathsf{P}} s_{\mathsf{P}}$ and $s^0_{\mathsf{R}} \not\xRightarrow{t'}_{\mathsf{R}} s_{\mathsf{R}}$ then it must be because P makes an output move that R cannot match. But then the previous composite state would be in F, which is contradictory. Hence $\epsilon \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

Case $t \equiv t'o$ with $o \in \mathcal{A}^O_{\mathsf{R}/\mathsf{P}}$. Suppose that $t'o \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Then $t' \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$, so by the induction hypothesis we derive $t' \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$. Therefore, $s^0_{\mathsf{R}/\mathsf{P}} \xRightarrow{t'}_{\mathsf{R}/\mathsf{P}} \bot_{\mathsf{R}/\mathsf{P}}$ and so $s^0_{\mathsf{R}/\mathsf{P}} \xRightarrow{t'o}_{\mathsf{R}/\mathsf{P}} \bot_{\mathsf{R}/\mathsf{P}}$. Thus, $t'o \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$.

Now suppose that $t'o \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$. Then $s^0_{\mathsf{R}/\mathsf{P}} \xRightarrow{t'o}_{\mathsf{R}/\mathsf{P}} \bot_{\mathsf{R}/\mathsf{P}}$. By the definition of $\bot_{\mathsf{R}/\mathsf{P}}$ (defined in terms of $\bot_{\mathcal{E}(\mathsf{R})}$), it follows that $s^0_{\mathsf{R}/\mathsf{P}} \xRightarrow{t'}_{\mathsf{R}/\mathsf{P}} \bot_{\mathsf{R}/\mathsf{P}}$, and so $t' \in F_{[\![\mathsf{R}/\mathsf{P}]\!]}$. By the induction hypothesis, it follows that $t' \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Hence, $t'o \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

Now suppose that $t'o \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Then by the induction hypothesis we know that $t' \in T_{[\![\mathsf{R}/\mathsf{P}]\!]}$.

Moreover, for all $t'' \in (\mathcal{A}_{\mathsf{R}} \setminus \mathcal{A}^O_{\mathsf{R/P}})^*$ it follows that $L(t'ot'')$ holds. So, if $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \stackrel{t'ot''}{\Longrightarrow}_{\mathsf{R/P}} s_{\mathsf{R}}/s_{\mathsf{P}}$, then certainly $s_{\mathsf{R}}/s_{\mathsf{P}} \notin F$. Furthermore, $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \stackrel{t'o}{\Longrightarrow}_{\mathsf{R/P}} s'_{\mathsf{R}}/s'_{\mathsf{P}}$ for some $s'_{\mathsf{R}}/s'_{\mathsf{P}}$, since $s^0_{\mathsf{P}} \stackrel{t'}{\Longrightarrow}_{\mathsf{P}} s''_{\mathsf{P}}$ for some s''_{P} as $o \notin \mathcal{A}_{\mathsf{P}}$ or $o \in \mathcal{A}^I_{\mathsf{P}}$.

Finally, suppose that $t'o \in T_{[\![\mathsf{R/P}]\!]}$. Then by the induction hypothesis, we know that $t' \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. As $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \stackrel{t'o}{\Longrightarrow}_{\mathsf{R/P}} s_{\mathsf{R}}/s_{\mathsf{P}}$ for some state $s_{\mathsf{R}}/s_{\mathsf{P}}$, it follows that $s_{\mathsf{R}}/s_{\mathsf{P}} \notin F$. Therefore, for any state $s'_{\mathsf{R}}/s'_{\mathsf{P}}$ such that $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \stackrel{t'ot''}{\Longrightarrow}_{\mathsf{R/P}} s'_{\mathsf{R}}/s'_{\mathsf{P}}$ with $t'' \in (\mathcal{A}_{\mathsf{R}} \setminus \mathcal{A}^O_{\mathsf{R/P}})^*$ we know that $s'_{\mathsf{R}}/s'_{\mathsf{P}} \notin F$. Consequently, if $t'ot'' \upharpoonright \mathcal{A}_{\mathsf{P}} \in F_{\mathsf{P}}$ then $t'ot'' \in F_{\mathcal{E}(\mathsf{R})}$, and if $t'ot'' \upharpoonright \mathcal{A}_{\mathsf{P}} \in T_{\mathsf{P}}$, then $t'ot'' \in T_{\mathcal{E}(\mathsf{R})}$. This means that $L(t'ot'')$ holds, and so we derive $t'o \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

Case $t \equiv t'i$ with $i \in \mathcal{A}^I_{\mathsf{R/P}}$. Suppose that $t'i \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. Then $t'i \upharpoonright \mathcal{A}_{\mathsf{P}} \notin T_{\mathsf{P}}$ or $t'i \in F_{\mathcal{E}(\mathsf{R})}$. By the induction hypothesis we know that $t' \in T_{[\![\mathsf{R/P}]\!]}$, which by input receptiveness of components, implies that $t'i \in T_{[\![\mathsf{R/P}]\!]}$. Now, if $t'i \upharpoonright \mathcal{A}_{\mathsf{P}} \notin T_{\mathsf{P}}$, then $t' \upharpoonright \mathcal{A}_{\mathsf{P}} \notin T_{\mathsf{P}}$ when $a \notin \mathcal{A}^O_{\mathsf{P}}$. Hence $t' \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$, which by the induction hypothesis gives $t' \in F_{[\![\mathsf{R/P}]\!]}$, and so $t'i \in F_{[\![\mathsf{R/P}]\!]}$. When $a \in \mathcal{A}^O_{\mathsf{P}}$, condition Q3 ensures that $t'i \in F_{[\![\mathsf{R/P}]\!]}$. If instead $t'i \in F_{\mathcal{E}(\mathsf{R})}$, then as $t'i \in T_{[\![\mathsf{R/P}]\!]}$, we know $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \stackrel{t'i}{\Longrightarrow}_{\mathsf{R/P}} s_{\mathsf{R}}/s_{\mathsf{P}}$. But $t'i \in F_{\mathcal{E}(\mathsf{R})}$ implies $s_{\mathsf{R}} = \bot_{\mathcal{E}(\mathsf{R})}$, hence $s_{\mathsf{R}}/s_{\mathsf{P}} = \bot_{\mathsf{R/P}}$, meaning $t'i \in F_{[\![\mathsf{R/P}]\!]}$.

Now suppose that $t'i \in F_{[\![\mathsf{R/P}]\!]}$. By the induction hypothesis and input receptiveness of components it follows that $t'i \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$. As $t'i \in F_{[\![\mathsf{R/P}]\!]}$, it follows that $s^0_{\mathsf{R}}/s^0_{\mathsf{P}} \stackrel{t'i}{\Longrightarrow}_{\mathsf{R/P}} \bot_{\mathsf{R/P}}$. But $\bot_{\mathsf{R/P}} = \bot_{\mathcal{E}(\mathsf{R})}/s_{\mathsf{P}}$ for some s_{P}. Hence, $t'i \in F_{\mathcal{E}(\mathsf{R})}$, which implies $t'i \in F_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$.

Showing that $t'i \in T_{[\![\mathsf{R}]\!]/[\![\mathsf{P}]\!]}$ iff $t'i \in T_{[\![\mathsf{R/P}]\!]}$ follows by the induction hypothesis and input receptiveness of components.

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

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 4.17. 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 $\bot_{\mathsf{Q}}$ is reachable from s^0_{Q} by hidden and output

actions implies $\bot_{\mathsf{P}}$ is reachable from s^0_{P} by hidden and output actions. $\diamond$

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 4.18. 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:

$$\mathsf{Q} \sqsubseteq_{op} \mathsf{P} \text{ iff } \forall \mathsf{R} \cdot \mathcal{A}^O_{\mathsf{R}} = \mathcal{A}^I_{\mathsf{P}} \text{ and } \mathcal{A}^I_{\mathsf{R}} = \mathcal{A}^O_{\mathsf{Q}} \implies \mathsf{Q} \parallel \mathsf{R} \sqsubseteq^F_{op} \mathsf{P} \parallel \mathsf{R}.$$

Proof. A straightforward modification to Theorem 3.30. $\square$

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

Corollary 4.19. 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 3.31 (with updated references). $\square$

4.8 On the Relationship with Interface Automata

In this section, we relate our operational theory of components to the interface automata of de Alfaro and Henzinger [dAH01]. 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.

We recall a general definition of interface automata [dAH01], which, unlike the restrictions imposed in [dAH05], 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, and does not require input enabledness in each state.

Definition 4.20. An *interface automaton* P is a tuple $\langle S_{\mathsf{P}}, \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, \rightarrow_{\mathsf{P}}, s^0_{\mathsf{P}} \rangle$, where:

- S_{P} is a finite set of states
- $\mathcal{A}^I_{\mathsf{P}}$ is a finite set of visible input actions (excluding the hidden action τ)
- $\mathcal{A}^O_{\mathsf{P}}$ is a finite set of visible output actions (excluding the hidden action τ), disjoint from $\mathcal{A}^I_{\mathsf{P}}$, where $\mathcal{A}_{\mathsf{P}} \triangleq \mathcal{A}^I_{\mathsf{P}} \cup \mathcal{A}^O_{\mathsf{P}}$
- $\rightarrow_{\mathsf{P}} \subseteq S_{\mathsf{P}} \times (\mathcal{A}_{\mathsf{P}} \cup \{\tau\}) \times S_{\mathsf{P}}$ is the transition relation
- $s^0_{\mathsf{P}} \in S_{\mathsf{P}}$ is the designated initial state. $\diamond$

Substitutive refinement of interface automata is given by means of alternating simulation, with a covariant inclusion on inputs and contravariant inclusion on outputs. Again, we reproduce the general definition from [dAH01], which is free of unnecessary restrictions. First, we introduce two shorthands for simplifying the definition:

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

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 4.21. 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.

4.8.1 Relation with Operational Components

We now indicate how to map interface automata to the operational components described earlier in this chapter. The mapping must add additional transitions for the non-enabled inputs to the special inconsistent state $\perp$.

Definition 4.22. Let P be an interface automaton. Then the corresponding operational component is $[\![\mathsf{P}]\!]^{IA} = \langle S_{\mathsf{P}} \cup \{\perp\}, \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, \longrightarrow, s^0_{\mathsf{P}}, \perp\rangle$, where:

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

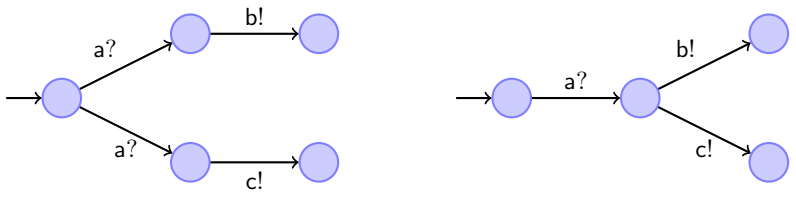

Figure 4.1: Interface automata distinguishing alternating simulation and $\sqsubseteq_{imp}$

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 4.23. 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 $t \in F_{\mathcal{E}([\![\mathsf{Q}]\!]^{IA})}$ and $t \notin F_{\mathcal{E}([\![\mathsf{P}]\!]^{IA})} \cup (T_{\mathcal{E}([\![\mathsf{P}]\!]^{IA})} \uparrow \mathcal{A}^I_{\mathsf{Q}})$. By definition of interface automata, it follows that $t \in F_{[\![\mathsf{Q}]\!]^{IA}}$ and $t \notin F_{[\![\mathsf{P}]\!]^{IA}} \cup (T_{[\![\mathsf{P}]\!]^{IA}} \uparrow \mathcal{A}^I_{\mathsf{Q}})$ 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 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 $q' \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{Q}} \stackrel{a}{\not\rightarrow}_{\mathsf{Q}}$ (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, $F_{[\![\mathsf{Q}]\!]^{IA}} \subseteq F_{[\![\mathsf{P}]\!]^{IA}} \cup (T_{[\![\mathsf{P}]\!]^{IA}} \uparrow \mathcal{A}^I_{\mathsf{Q}})$ as required.

Now suppose that $\mathsf{Q} \sqsubseteq_{IA} \mathsf{P}$ and let t be the smallest trace such that $t \in T_{[\![\mathsf{Q}]\!]^{IA}} \setminus F_{[\![\mathsf{Q}]\!]^{IA}}$ and $t \notin T_{[\![\mathsf{P}]\!]^{IA}} \cup (T_{[\![\mathsf{P}]\!]^{IA}} \uparrow \mathcal{A}^I_{\mathsf{Q}})$. 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 $q' \stackrel{\epsilon}{\Longrightarrow}_{\mathsf{Q}} \stackrel{a}{\rightarrow}_{\mathsf{Q}}$, 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, $T_{[\![\mathsf{Q}]\!]^{IA}} \subseteq T_{[\![\mathsf{P}]\!]^{IA}} \cup (T_{[\![\mathsf{P}]\!]^{IA}} \uparrow \mathcal{A}^I_{\mathsf{Q}})$. □

Being a branching-time relation, alternating refinement is too strong for substitutivity. This is demonstrated by the interface automata in Figure 4.1. 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 4.23 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 4.24. 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 4.23, 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 $q' \stackrel{a}{\not\rightarrow}_{\mathsf{Q}}$ while $p' \stackrel{a}{\rightarrow}_{\mathsf{P}}$ 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 $q' \stackrel{a}{\rightarrow}_{\mathsf{Q}}$ while $p' \stackrel{a}{\not\rightarrow}_{\mathsf{P}}$ 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. □

4.8.2 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 $[\![\mathsf{P}]\!]^{IA} \ || \ [\![\mathsf{Q}]\!]^{IA}$, 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 [BR08], 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.

4.9 Summary

This chapter has presented an operational formulation of the substitutive and progress-sensitive specification theories introduced in Chapter 3. Operational definitions of parallel composition, conjunction, disjunction, hiding and quotient are included, while refinement is still defined in terms of trace containment, so that we obtain the weakest preorder respecting substitutive and progress-sensitive refinement. Correspondences are made explicit between the trace-based and operational definitions of the operators, with respect to substitutive and progress-sensitive equivalence, which allows users to freely move between the formalisms. Consequently, all of the compositionality results for the trace-based frameworks continue to hold in the operational setting. A natural advantage of the operational representation is that it is much more akin to actual implementations (i.e., programs). This makes the theory more amenable to automating model construction and performing component-based design and reasoning.

To make clear the links between our work and the theory of interface automata, we provided an embedding of interface automata within our operational framework, and demonstrated that alternating refinement is a stronger version of our substitutive preorder (after a trivial modification to the alternating refinement relation so as to account for non-enabled inputs). Under the assumption of determinism, it is shown that our linear-time refinement coincides with the branching-time alternating refinement due to de Alfaro and Henzinger.

CHAPTER

FIVE

Assume-Guarantee Reasoning for Components

The components of Chapter 3 can be thought of as both implementations and specifications of systems. They are implementations in the sense that they can interact with their surroundings by accepting input from the environment and producing output. However, the components can also be thought of as specifications, in that they place assumptions on the inputs issued by the environment, and provide guarantees on their own behaviour. Thus, the specification theory of Chapter 3 permits the mixing of specifications and implementations, and allows for the construction of new components from existing ones by means of compositional operators [BCF$^+$08, LNW07, DHJP08, RBB$^+$11]. For such components, treated as specifications, the assumptions and guarantees are merged into a single behavioural representation.

In many cases, combining assumptions and guarantees avoids duplication of common information. However, it can be desirable to manipulate the assumptions and guarantees separately. For instance, we may want to express a simple guarantee (such as "no failure will occur") without having to weave it into a complex assumption. Separation of assumptions from guarantees also supports specification reuse, in that the same guarantees (or assumptions) can be used for several related interfaces, each representing different versions of a component.

In this chapter, we formulate a compositional assume-guarantee (AG) framework for reasoning about the properties satisfied by components, treated as implementations, as modelled in Chapter 3. An AG specification, otherwise known as a contract, consists of an assumption, guarantee and progress property, each of which are explicitly represented by sets of finite traces. This facilitates reasoning about safety and progress properties, and differs from (arguably) more complex approaches based on modal specifications and alternating simulation. Treating contracts as first-class citizens, we define the operators of parallel, conjunction, disjunction and quotient on contracts, and prove compositionality. This is the first work to present such an extensive collection of operators directly on contracts (to our knowledge, quotient has not previously been defined), which supports flexible development and verification of component-based systems using AG principles.

In relating implementations (components) with contracts by means of satisfaction, a notion of refinement corresponding to implementation containment is defined on contracts. Based on this, we formulate a collection of sound and complete AG reasoning rules for the preservation of safety and progress properties under the operations and refinement preorder of the specification theory.

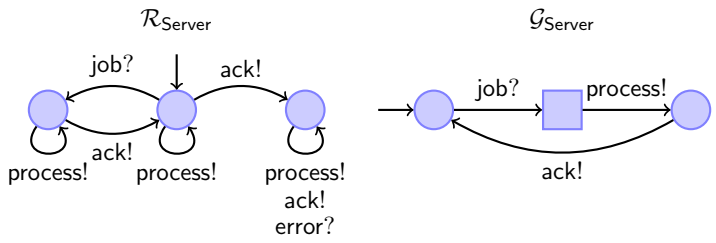

Figure 5.1: Assumption and guarantee of Server

This allows for the use of compositional AG reasoning, which enables the decomposition of the system into smaller components, each of which may be reasoned about in isolation during system development and verification.

Outlining the remainder of this chapter, Section 5.1 introduces the AG framework for safety properties and presents a number of sound and complete rules for the operators of the specification theory, while Section 5.2 extends this framework with progress-sensitivity. Section 5.3 documents a link layer protocol case study demonstrating the features of the assume-guarantee frameworks. Finally, in Section 5.4, we conclude.

5.1 Assume-Guarantee Framework for Safety Properties

To support component-based reasoning, we introduce the concept of a contract, which consists of two prefix-closed sets of traces referred to as the *assumption* and *guarantee*. The assumption specifies the environment's allowable interaction sequences, while the guarantee is a constraint on the component's behaviour. As assumptions and guarantees are prefix-closed, our theory ensures that components preserve (not necessarily regular) safety properties.

Definition 5.1 (Contract). A *contract* $\mathcal{S}$ is a tuple $\langle \mathcal{A}^I_{\mathcal{S}}, \mathcal{A}^O_{\mathcal{S}}, \mathcal{R}_{\mathcal{S}}, \mathcal{G}_{\mathcal{S}} \rangle$, in which $\mathcal{A}^I_{\mathcal{S}}$ and $\mathcal{A}^O_{\mathcal{S}}$ are disjoint sets (whose union is $\mathcal{A}_{\mathcal{S}}$), referred to as the inputs and outputs respectively, and $\mathcal{R}_{\mathcal{S}}$ and $\mathcal{G}_{\mathcal{S}}$ are prefix closed subsets of $\mathcal{A}^*_{\mathcal{S}}$, referred to as the assumption and guarantee respectively, such that $t \in \mathcal{R}_{\mathcal{S}}$ and $t' \in (\mathcal{A}^O_{\mathcal{S}})^*$ implies $tt' \in \mathcal{R}_{\mathcal{S}}$. ◇

Since outputs are controlled by the component, we insist that assumptions are closed under output-extensions. On the other hand, we need not insist that the guarantee is closed under input-extensions, since the assumption can select inputs under which the guarantee is given. This contrasts with the work of [LNW06], in which guarantees must be closed under input-extensions; one of our contributions is to show that this is not necessary, thus allowing significantly more flexibility when formulating contracts.

Example 5.2. Figure 5.1 presents a contract for a Server, which can receive jobs, process jobs, acknowledge the processing of a job, and be placed in error mode. The interface is given by all the

actions appearing in the diagram, with the convention that actions followed by ? (resp. !) are inputs (resp. outputs). At this stage, the distinction between square and circle nodes is irrelevant, but will be explained in Example 5.38, Section 5.2. The assumption leaves process unconstrained, but ensures that error will never be sent providing job and ack alternate in that order. The guarantee requires that any job received can only be acknowledged after having been processed, and a new job can only arrive after the previous one has been acknowledged. ◇

Given a contract $\mathcal{S}$, we want to be able to say whether a component $\mathcal{P}$ satisfies $\mathcal{S}$. Informally, $\mathcal{P}$ satisfies $\mathcal{S}$ if, for any interaction between $\mathcal{P}$ and the environment characterised by a trace t, if $t \in \mathcal{R}_\mathcal{S}$, then $t \in \mathcal{G}_\mathcal{S}$ and t cannot become inconsistent in $\mathcal{P}$ without further stimulation from the environment. Components can thus be thought of as implementations of contracts.

Definition 5.3 (Satisfaction). A component $\mathcal{P}$ *satisfies* the contract $\mathcal{S}$, written $\mathcal{P} \models \mathcal{S}$, iff:

S1. $\mathcal{A}^I_\mathcal{S} \subseteq \mathcal{A}^I_\mathcal{P}$

S2. $\mathcal{A}^O_\mathcal{P} \subseteq \mathcal{A}^O_\mathcal{S}$

S3. $\mathcal{A}^I_\mathcal{P} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$

S4. $\mathcal{R}_\mathcal{S} \cap T_\mathcal{P} \subseteq \mathcal{G}_\mathcal{S} \cap \overline{F_\mathcal{P}}$. ◇

By output-extension closure of assumptions, condition S4 is equivalent to checking $\mathcal{R}_\mathcal{S} \cap T_\mathcal{P} \subseteq \mathcal{G}_\mathcal{S} \cap \overline{F_{\mathcal{E}(\mathcal{P})}}$, which involves the safe representation $\mathcal{E}(\mathcal{P})$ of $\mathcal{P}$ (see Definition 3.3). The following lemma shows that this definition of satisfaction is preserved under the component-based refinement corresponding to safe-substitutivity, subject to compatibility.

Lemma 5.4. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}$ be a contract. If $\mathcal{P} \models \mathcal{S}$, $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ and $\mathcal{A}^I_\mathcal{Q} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$, then $\mathcal{Q} \models \mathcal{S}$.

Proof. We show that $\mathcal{R}_\mathcal{S} \cap T_\mathcal{Q} \subseteq \mathcal{G}_\mathcal{S} \cap \overline{F_\mathcal{Q}}$. Let $t \in \mathcal{R}_\mathcal{S} \cap T_\mathcal{Q}$. From $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$ it follows that $t \in T_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$. But, in fact, $t \in T_{\mathcal{E}(\mathcal{P})}$ as $\mathcal{A}^I_\mathcal{S} \subseteq \mathcal{A}^I_\mathcal{P} \subseteq \mathcal{A}^I_\mathcal{Q}$, $t \in \mathcal{A}^*_\mathcal{S}$ and $\mathcal{A}^I_\mathcal{Q} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Therefore, either $t \in T_\mathcal{P}$ or $t \in F_{\mathcal{E}(\mathcal{P})}$. For the former, $t \in \mathcal{R}_\mathcal{S} \cap T_\mathcal{P}$ implies $t \in \mathcal{G}_\mathcal{S} \cap \overline{F_{\mathcal{E}(\mathcal{P})}}$. As $t \notin F_{\mathcal{E}(\mathcal{P})}$ (and moreover $t \notin T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q}$) it follows that $t \notin F_{\mathcal{E}(\mathcal{Q})}$ since $\mathcal{Q} \sqsubseteq_{imp} \mathcal{P}$. Hence $t \in \mathcal{G}_\mathcal{S} \cap \overline{F_\mathcal{Q}}$ as required. If instead $t \in F_{\mathcal{E}(\mathcal{P})}$, then either $t \equiv \epsilon$, or there is some prefix $t'i$ of t with $i \in \mathcal{A}^I_\mathcal{P}$ such that $t' \notin F_{\mathcal{E}(\mathcal{P})}$ while $t'i \in F_{\mathcal{E}(\mathcal{P})}$. For both cases $\mathcal{P} \not\models \mathcal{S}$, which is contradictory (the latter because $t'i \in T_\mathcal{P}$). □

Based on this result, a contract can be characterised by its least refined satisfying component, which is the minimal satisfying component under the substitutive refinement preorder. Note that every contract has at least one satisfying component, although it may not be realisable. In the case that a contract has a realisable satisfying component, the contract is said to be implementable, and

such a component is said to be an implementation. In order to construct such a component, it is necessary to determine the set of violating traces of the contract. These are traces that cannot be in any satisfying component, because they will violate the guarantee.

Definition 5.5. Let $\mathcal{S}$ be a contract. Then $\mathsf{violations}(\mathcal{S})$ is defined as $\{t \in \mathcal{A}_{\mathcal{S}}^* : \exists t' \in (\mathcal{A}_{\mathcal{S}}^I)^* \cdot tt' \in \mathcal{R}_{\mathcal{S}} \cap \overline{\mathcal{G}_{\mathcal{S}}}\} \cdot \mathcal{A}_{\mathcal{S}}^*$. ◇

Clearly, if $t \in \mathcal{R}_{\mathcal{S}} \cap \overline{\mathcal{G}_{\mathcal{S}}}$, then t cannot be a trace of any implementation of $\mathcal{S}$. Moreover, if there is a trace that can be extended by a sequence of inputs to become t, then this also cannot be in a satisfying component, due to input-receptiveness of components. Therefore $\mathsf{violations}(\mathcal{S})$ consists of all traces from which the environment can, under its own control, violate the guarantee. The following definition shows how to construct the minimal satisfying component for a contract, with respect to substitutive refinement.

Definition 5.6. Let $\mathcal{S}$ be a contract. Then the *least refined component satisfying* $\mathcal{S}$ is the component $\mathcal{I}(\mathcal{S}) = \langle \mathcal{A}_{\mathcal{S}}^I, \mathcal{A}_{\mathcal{S}}^O, T_{\mathcal{I}(\mathcal{S})}, F_{\mathcal{I}(\mathcal{S})} \rangle$, where:

- $T_{\mathcal{I}(\mathcal{S})} = \overline{\mathsf{violations}(\mathcal{S})}$
- $F_{\mathcal{I}(\mathcal{S})} = \overline{\mathsf{violations}(\mathcal{S})} \cap \overline{\mathcal{R}_{\mathcal{S}}}$. ◇

The traces of $\mathcal{I}(\mathcal{S})$ are simply the behaviours that will never violate the contract. This means that, if a trace of $\mathcal{I}(\mathcal{S})$ is in the assumption, then it must also be in the guarantee, which ensures that $\mathcal{I}(\mathcal{S})$ satisfies the safety constraints of $\mathcal{S}$.

We now state the properties of the least refined satisfying component.

Lemma 5.7. Let $\mathcal{S}$ be a contract, and let $\mathcal{P}$ be a component. Then:

- $\mathcal{I}(\mathcal{S})$ is non-realisable implies $\mathcal{S}$ is non-implementable;
- $\mathcal{I}(\mathcal{S}) \models \mathcal{S}$; and
- $\mathcal{P} \models \mathcal{S}$ iff $\mathcal{P} \sqsubseteq_{imp} \mathcal{I}(\mathcal{S})$.

Proof. For the first claim, note that, if $\epsilon \notin T_{\mathcal{I}(\mathcal{S})}$, then there exists $t \in (\mathcal{A}_{\mathcal{S}}^I)^*$ such that $t \in \mathcal{R}_{\mathcal{S}} \cap \overline{\mathcal{G}_{\mathcal{S}}}$. As every realisable implementation must have t in its T-set, it follows that $\mathcal{S}$ has no implementations.

For the second claim, suppose $t \in \mathcal{R}_{\mathcal{S}} \cap T_{\mathcal{I}(\mathcal{S})}$. Then $t \in \mathcal{R}_{\mathcal{S}} \cap \overline{\mathsf{violations}(\mathcal{S})}$, which implies $t \in \mathcal{G}_{\mathcal{S}} \cap \overline{F_{\mathcal{I}(\mathcal{S})}}$.

For the third claim, the if direction follows by the previous claim and Lemma 5.4. For the only if direction, we need to show that $T_{\mathcal{E}(\mathcal{P})} \subseteq T_{\mathcal{I}(\mathcal{S})} \cup (T_{\mathcal{I}(\mathcal{S})} \uparrow \mathcal{A}_{\mathcal{P}}^I)$ and $F_{\mathcal{E}(\mathcal{P})} \subseteq F_{\mathcal{I}(\mathcal{S})} \cup (T_{\mathcal{I}(\mathcal{S})} \uparrow \mathcal{A}_{\mathcal{P}}^I)$. If $t \in T_{\mathcal{E}(\mathcal{P})}$ and $t \notin \mathcal{A}_{\mathcal{S}}^*$, then there is a prefix $t'a$ of t such that $t' \in \mathcal{A}_{\mathcal{S}}^*$ and $a \in \mathcal{A}_{\mathcal{P}}^I \setminus \mathcal{A}_{\mathcal{S}}$,

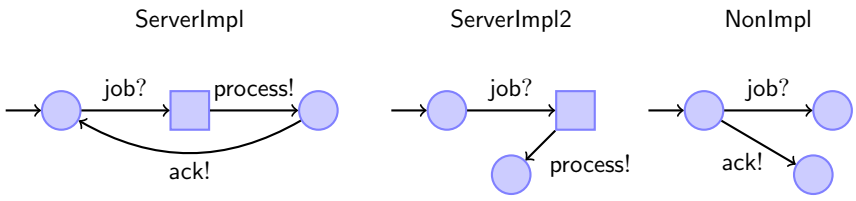

Figure 5.2: Implementations and non-implementation of Server

which by an inductive argument that assumes the result holds for all strict prefixes allows us to derive $t \in T_{\mathcal{I}(\mathcal{S})} \uparrow \mathcal{A}^I_{\mathcal{P}}$. So suppose that $t \in T_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}}$. Then since $\mathcal{P} \models \mathcal{S}$, it follows that $t \notin \mathsf{violations}(\mathcal{S})$. Hence $t \in T_{\mathcal{I}(\mathcal{S})}$. Now suppose that $t \in F_{\mathcal{E}(\mathcal{P})} \cap \mathcal{A}^*_{\mathcal{S}}$. Then as $\mathcal{P} \models \mathcal{S}$, it follows that $t \notin \mathsf{violations}(\mathcal{S})$ and $t \notin \mathcal{R}_{\mathcal{S}}$. Consequently, $t \in F_{\mathcal{I}(\mathcal{S})}$. □

Example 5.8. ServerImpl in Figure 5.2 is the least refined component satisfying the contract Server of Figure 5.1. As a convention, we omit input transitions to inconsistent states when drawing components (consequently, there are implicit inconsistent job transitions from the middle and last states and implicit inconsistent error transitions from all states). As ServerImpl2 $\sqsubseteq_{imp}$ ServerImpl, ServerImpl2 is also an implementation of Server, even though no acknowledgement is performed (since this is a contraction of the allowable output behaviour of Server). NonImpl is not an implementation, because $\langle \mathsf{ack} \rangle \in \mathsf{violations}(\mathsf{Server})$, since $\langle \mathsf{ack}, \mathsf{error} \rangle \in \mathcal{R}_{\mathsf{Server}} \cap T_{\mathsf{NonImpl}}$, while $\langle \mathsf{ack}, \mathsf{error} \rangle \notin \mathcal{G}_{\mathsf{Server}}$. ◇

5.1.1 Refinement

Satisfaction of a contract by a component allows us to define a natural hierarchy on contracts corresponding to implementation containment. A constructive definition for this refinement relation follows.

Definition 5.9 (Refinement). Let $\mathcal{S}$ and $\mathcal{T}$ be contracts. $\mathcal{S}$ is said to be a *refinement* of $\mathcal{T}$, written $\mathcal{S} \sqsubseteq \mathcal{T}$, iff:

R1. $\mathcal{A}^I_{\mathcal{T}} \subseteq \mathcal{A}^I_{\mathcal{S}}$

R2. $\mathcal{A}^O_{\mathcal{S}} \subseteq \mathcal{A}^O_{\mathcal{T}}$

R3. $\mathcal{A}^I_{\mathcal{S}} \cap \mathcal{A}^O_{\mathcal{T}} = \emptyset$

R4. $\mathsf{violations}(\mathcal{T}) \cap \mathcal{A}^*_{\mathcal{S}} \subseteq \mathsf{violations}(\mathcal{S})$

R5. $\mathcal{R}_{\mathcal{T}} \cap \mathcal{A}^*_{\mathcal{S}} \subseteq \mathcal{R}_{\mathcal{S}} \cup \mathsf{violations}(\mathcal{S})$. ◇

It is our intention that $\mathcal{S} \sqsubseteq \mathcal{T}$ iff the implementations of $\mathcal{S}$ are contained within the implementations of $\mathcal{T}$ (subject to compatibility). Conditions R1-R3 impose necessary conditions on the alphabets to uphold this principle. For condition R4, any component having a trace $t \in \mathsf{violations}(\mathcal{T}) \cap \mathcal{A}_{\mathcal{S}}^*$ cannot be an implementation of $\mathcal{T}$, so it should not be an implementation of $\mathcal{S}$. For this to be the case, the component must violate the guarantee on $\mathcal{S}$, i.e., $t \in \mathsf{violations}(\mathcal{S})$. Condition R5 deals with inconsistent traces. If a component has an inconsistent trace $t \in \mathcal{R}_{\mathcal{T}} \cap \mathcal{A}_{\mathcal{S}}^*$, then this cannot be an implementation of $\mathcal{T}$. Consequently, the component must not be an implementation of $\mathcal{S}$, so either t must violate the guarantee of $\mathcal{S}$, i.e., $t \in \mathsf{violations}(\mathcal{S})$, or t must be in $\mathcal{R}_{\mathcal{S}}$, so that the component cannot satisfy $\mathcal{S}$. These conditions guarantee implementation containment, and also that refinement is a preorder (subject to compatibility).

Definition 5.9 gives a sound and complete characterisation of refinement, as proven in Lemma 5.10. In related work, one often sees sound and incomplete characterisations, which may be more intuitive. One possibility is to replace conditions R4 and R5 by $\mathcal{R}_{\mathcal{T}} \subseteq \mathcal{R}_{\mathcal{S}}$ and $(\mathcal{G}_{\mathcal{S}} \cap \mathcal{R}_{\mathcal{T}}) \subseteq \mathcal{G}_{\mathcal{T}}$ (assuming identical interfaces of $\mathcal{S}$ and $\mathcal{T}$). This defines an equivalence on contracts with the same assumptions, where the guarantees differ only outside the assumptions. More formally, $\mathcal{S}$ and $\mathcal{T}$ are equivalent if $\mathcal{R}_{\mathcal{S}} = \mathcal{R}_{\mathcal{T}}$ and $(\mathcal{G}_{\mathcal{S}} \cap \mathcal{R}_{\mathcal{S}}) = (\mathcal{G}_{\mathcal{T}} \cap \mathcal{R}_{\mathcal{T}})$.

Lemma 5.10. Refinement captures implementation containment:

$$\mathcal{S} \sqsubseteq \mathcal{T} \iff \{\mathcal{P} : \mathcal{P} \models \mathcal{S} \text{ and } \mathcal{A}_{\mathcal{P}}^I \cap \mathcal{A}_{\mathcal{T}}^O = \emptyset\} \subseteq \{\mathcal{P} : \mathcal{P} \models \mathcal{T}\}.$$

Proof. For the only if direction, suppose that $\mathcal{P} \models \mathcal{S}$, $\mathcal{S} \sqsubseteq \mathcal{T}$ and $\mathcal{A}_{\mathcal{P}}^I \cap \mathcal{A}_{\mathcal{T}}^O = \emptyset$. We first show that $\mathcal{P} \models \mathcal{T}$, so suppose that $t \in \mathcal{R}_{\mathcal{T}} \cap T_{\mathcal{P}}$. Then, by the definition of $\sqsubseteq$, it follows that $t \in \mathcal{R}_{\mathcal{S}} \cup \mathsf{violations}(\mathcal{S})$ by condition R5. If $t \in \mathsf{violations}(\mathcal{S})$, then $t \notin T_{\mathcal{P}}$, since $\mathcal{P} \models \mathcal{S}$, which is contradictory. Therefore, $t \in \mathcal{R}_{\mathcal{S}}$, which from $\mathcal{P} \models \mathcal{S}$ implies $t \in \mathcal{G}_{\mathcal{S}} \cap \overline{F_{\mathcal{P}}}$. But as $t \notin \mathsf{violations}(\mathcal{S})$, it follows that $t \notin \mathsf{violations}(\mathcal{T})$ and so $t \in \mathcal{G}_{\mathcal{T}}$. Hence, $t \in \mathcal{G}_{\mathcal{T}} \cap \overline{F_{\mathcal{P}}}$ as required, which implies $\mathcal{P} \models \mathcal{T}$.

For the if direction, Lemmas 5.4 and 5.7 allow us to deduce that $\mathcal{I}(\mathcal{S}) \sqsubseteq_{imp} \mathcal{I}(\mathcal{T})$. Suppose that $t \in \mathsf{violations}(\mathcal{T}) \cap \mathcal{A}_{\mathcal{S}}^*$. Then $t \notin T_{\mathcal{I}(\mathcal{T})}$, hence $t \notin T_{\mathcal{I}(\mathcal{S})}$, meaning $t \in \mathsf{violations}(\mathcal{S})$. Now suppose that $t \in \mathcal{R}_{\mathcal{T}} \cap \mathcal{A}_{\mathcal{S}}^*$. Then $t \notin F_{\mathcal{I}(\mathcal{T})}$, which implies $t \notin F_{\mathcal{I}(\mathcal{S})}$, hence $t \notin \overline{\mathcal{R}_{\mathcal{S}}} \cap \overline{\mathsf{violations}(\mathcal{S})}$ i.e., $t \in \mathcal{R}_{\mathcal{S}} \cup \mathsf{violations}(\mathcal{S})$. Thus, $\mathcal{S} \sqsubseteq \mathcal{T}$. □

[LNW06] give a sound and complete characterisation of their refinement relation (which corresponds to implementation containment, as in this chapter) by means of conformance tests. The definition assumes equality of interfaces, so does not need to deal with issues of compatibility or the complexities of both covariant and contravariant inclusion of inputs and outputs respectively (i.e., conditions R1-R3). Thus, their definition largely corresponds to condition R4. Condition R5 is not necessary in that setting, as implementation models are required to be input-enabled.

Refinement can be shown to be a preorder, provided that we add the minor technical condition that compatibility of components is maintained.

Lemma 5.11 (Weak transitivity). Let $\mathcal{S}$, $\mathcal{T}$ and $\mathcal{U}$ be contracts such that $\mathcal{A}_{\mathcal{S}}^{I} \cap \mathcal{A}_{\mathcal{U}}^{O} = \emptyset$. If $\mathcal{S} \sqsubseteq \mathcal{T}$ and $\mathcal{T} \sqsubseteq \mathcal{U}$, then $\mathcal{S} \sqsubseteq \mathcal{U}$.

Proof. Essentially follows from transitivity of $\subseteq$. □

Example 5.12. A new contract Server2 (based on Server in Figure 5.1, with assumption $\mathcal{R}_{\mathsf{Server}}$ and guarantee obtained from $\mathcal{G}_{\mathsf{Server}}$ by removing the ack transition) is a refinement of Server, since it has fewer implementations. In particular, $\mathcal{I}(\mathsf{Server2}) = \mathsf{ServerImpl2}$. ServerImpl is not an implementation of Server2 because $\langle \mathsf{job}, \mathsf{process}, \mathsf{ack} \rangle \in \mathsf{violations}(\mathsf{Server2})$. ◇

As we can represent a contract by its most general satisfying component, we can also do the reverse and represent a component by its most general contract. This can be found by examining the component's safe traces.

Definition 5.13. The *characteristic contract* for component $\mathcal{P}$ is a contract $\mathcal{AG}(\mathcal{P}) = \langle \mathcal{A}_{\mathcal{P}}^{I}, \mathcal{A}_{\mathcal{P}}^{O}, \mathcal{R}_{\mathcal{AG}(\mathcal{P})}, \mathcal{G}_{\mathcal{AG}(\mathcal{P})} \rangle$, where $\mathcal{R}_{\mathcal{AG}(\mathcal{P})} = \mathcal{A}_{\mathcal{P}}^{*} \setminus F_{\mathcal{E}(\mathcal{P})}$ and $\mathcal{G}_{\mathcal{AG}(\mathcal{P})} = T_{\mathcal{P}} \setminus F_{\mathcal{E}(\mathcal{P})}$. ◇

The largest assumption safe for component $\mathcal{P}$ is the set of all traces that cannot become inconsistent under $\mathcal{P}$'s own control, while the guarantee is this same set of traces constrained to the behaviour of $\mathcal{P}$. The following lemma shows the properties of the characteristic contract.

Lemma 5.14. Let $\mathcal{P}$ be a component and let $\mathcal{S}$ be a contract. Then:

- $\mathcal{P} \models \mathcal{AG}(\mathcal{P})$; and
- $\mathcal{P} \models \mathcal{S}$ iff $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}$.

Proof. For the first claim, let $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})} \cap T_{\mathcal{P}}$. Then, as $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$, it follows $t \in \overline{F_{\mathcal{E}(\mathcal{P})}}$. Given $t \in T_{\mathcal{P}}$, it thus follows $t \in \mathcal{G}_{\mathcal{AG}(\mathcal{P})} \cap \overline{F_{\mathcal{E}(\mathcal{P})}}$ as required.

For the second claim, the if direction follows by the previous claim and Lemma 5.10. For the only if direction, first suppose $t \in \mathsf{violations}(\mathcal{S}) \cap \mathcal{A}_{\mathcal{P}}^{*}$. Then $t \notin T_{\mathcal{P}} \cup F_{\mathcal{E}(\mathcal{P})}$ as $\mathcal{P} \models \mathcal{S}$, which implies $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})}}$. Hence $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$. Now suppose that $t \in \mathcal{R}_{\mathcal{S}} \cap \mathcal{A}_{\mathcal{P}}^{*}$. Then $\mathcal{P} \models \mathcal{S}$ implies $t \notin T_{\mathcal{P}}$ or $t \notin F_{\mathcal{E}(\mathcal{P})}$. Note that $t \notin T_{\mathcal{P}}$ implies $t \notin F_{\mathcal{E}(\mathcal{P})}$ (consider a prefix in $T_{\mathcal{P}} \cap F_{\mathcal{E}(\mathcal{P})}$), so $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$. Therefore, we derive $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}$. □

The final point in the previous lemma shows that satisfaction of a contract by a component is equivalent to checking whether the characteristic contract of the component is a refinement of the contract. This means that implementability of contracts, built up compositionally, follows immediately from compositionality results on contracts.

In the subsequent sections, we define the compositional operators of the specification theory directly on contracts. The operators are only defined when the contracts to be composed are

composable (the conditions being specified as part of the definitions). We also present a number of sound and complete AG rules for inferring properties of composite systems from the properties of their subcomponents.

5.1.2 Parallel Composition

The parallel composition of contracts is defined as the least refined contract satisfying independent implementability. Therefore, $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q}$ is the smallest contract having $\mathcal{P} \mid\mid \mathcal{Q}$ as an implementation whenever $\mathcal{P} \models \mathcal{S}_\mathcal{P}$ and $\mathcal{Q} \models \mathcal{S}_\mathcal{Q}$. A constructive definition of contract composition is based on the well-established theorem of [AL93], which has appeared in several forms [Col93, AL95, JT96]. The composed contract has the largest assumption that prevents any implementation (say $\mathcal{P}$) of one contract ($\mathcal{S}_\mathcal{P}$) producing behaviour observable by the other contract ($\mathcal{S}_\mathcal{Q}$) that is outside of its assumption ($\mathcal{R}_{\mathcal{S}_\mathcal{Q}}$). The guarantee of the composition, on the other hand, is constrained to what can be guaranteed by both contracts to be composed.

Definition 5.15. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ be contracts composable for parallel composition (i.e., $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} = \emptyset$). Then $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}, \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}, \mathcal{G}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \rangle$, where:

- $\mathcal{A}^I_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}) \setminus (\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}})$
- $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = \mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$
- $\mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ is the largest prefix closed set such that $\mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}(\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})^*$ is contained within the union of:
  - $(\mathcal{R}_{\mathcal{S}_\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}) \cap (\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})$
  - $\mathsf{violations}(\mathcal{S}_\mathcal{P}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$
  - $\mathsf{violations}(\mathcal{S}_\mathcal{Q}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$
- $\mathcal{G}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \cap \overline{(\mathsf{violations}(\mathcal{S}_\mathcal{P}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})} \cap \overline{(\mathsf{violations}(\mathcal{S}_\mathcal{Q}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})}$. ◇

The assumption $\mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ captures all behaviours whose projections onto $\mathcal{A}_{\mathcal{S}_\mathcal{P}}$ and $\mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ are either contained within the assumptions $\mathcal{R}_{\mathcal{S}_\mathcal{P}}$ and $\mathcal{R}_{\mathcal{S}_\mathcal{Q}}$, or have violated at least one of the contracts. This rules out a trace t that has not violated either of the contracts, but is no longer within both assumptions (say $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$). For such a trace, no guarantee can be given, since $\mathcal{S}_\mathcal{P}$ can have an implementation with the inconsistent trace $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}}$, while $\mathcal{S}_\mathcal{Q}$ can have an implementation with the trace $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$. The parallel composition of these two components would thus be inconsistent on t, and so would not satisfy $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q}$ if $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$.

The guarantee $\mathcal{G}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ is constrained to the traces in $\mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ that do not violate either $\mathcal{S}_\mathcal{P}$ or $\mathcal{S}_\mathcal{Q}$. Any trace in an implementation of a contract must not be allowed to violate the contract, meaning that it must suppress an output before a violation can occur. Consequently, the parallel

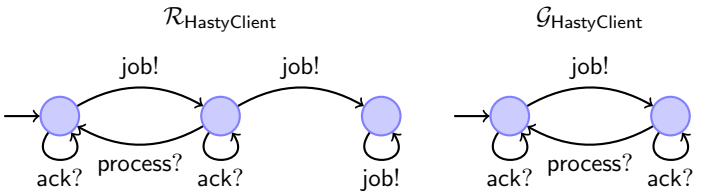

Figure 5.3: Assumption and guarantee of HastyClient

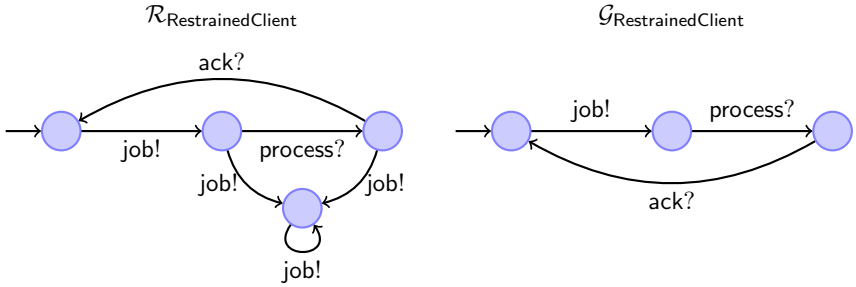

Figure 5.4: Assumption and guarantee of RestrainedClient

composition of such an implementation with an implementation of the other contract cannot proceed beyond this suppressed output, so $\mathcal{G}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ need not guarantee anything beyond that output. Thus, $\mathcal{G}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ contains only traces reachable by the composition of any two implementations of the respective contracts that are in the assumption $\mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$.

Example 5.16. Figure 5.3 presents a contract HastyClient that can send a job to a server whenever the last job has been processed, regardless of whether it has been acknowledged or not. The composition of HastyClient with Server is a contract for which nothing can be assumed or guaranteed, since the output sequence $\langle \mathsf{job!}, \mathsf{process!}, \mathsf{job!} \rangle$ is not in $\mathsf{violations}(\mathsf{HastyClient})$, but is also not in $\mathcal{R}_{\mathsf{Server}}$ or $\mathsf{violations}(\mathsf{Server})$. This is problematic because $\langle \mathsf{job!}, \mathsf{process?}, \mathsf{job!} \rangle$ can be a trace in an implementation of HastyClient, while $\langle \mathsf{job?}, \mathsf{process!}, \mathsf{job?} \rangle$ can be an inconsistent trace in an implementation of Server (providing $\langle \mathsf{job?}, \mathsf{process!} \rangle$ is consistent, since $\langle \mathsf{job?}, \mathsf{process!}, \mathsf{job?} \rangle \notin \mathcal{R}_{\mathsf{Server}}$). Note that $\langle \mathsf{job!}, \mathsf{process!}, \mathsf{job!} \rangle$ is an inconsistent trace in the parallel composition of the two implementations, which explains why the assumption must be empty. ◇

Example 5.17. In contrast to HastyClient, the composition of RestrainedClient (Figure 5.4) and Server is a contract with a completely open assumption (anything may be assumed), since the allowed behaviours of each contract cannot violate, or fall outside the assumption of, the other contract). The guarantee is equivalent to $\mathcal{G}_{\mathsf{Server}}$, having converted all actions to outputs. ◇

Subject to suitable constraints on the interfaces of contracts, it can be shown that parallel composition is monotonic under refinement.

Theorem 5.18. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for parallel composition, such that $\mathcal{A}_{\mathcal{S}'_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = \emptyset$. If $\mathcal{S}'_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$, then $\mathcal{S}'_\mathcal{P} \,||\, \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \,||\, \mathcal{S}_\mathcal{Q}$.

Proof. Note that the alphabet constraints are satisfied, so first show $\mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \subseteq \mathcal{R}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \cup \mathsf{violations}(\mathcal{S}'_\mathcal{P} \,||\, \mathcal{S}'_\mathcal{Q})$. Suppose $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}$, and all strict prefixes of t are in $\mathcal{R}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \cap \overline{\mathsf{violations}(\mathcal{S}'_\mathcal{P} \,||\, \mathcal{S}'_\mathcal{Q})}$. If $t \notin \mathcal{R}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}$, then there exists $t' \in (\mathcal{A}^O_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}})^*$ such that, without loss of generality, $tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}'_\mathcal{P}} \cup \mathsf{violations}(\mathcal{S}'_\mathcal{P})$ and $tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \notin \mathsf{violations}(\mathcal{S}'_\mathcal{Q})$. As $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} = tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}}$ and $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{Q}}$, it follows that $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{P})$ since $\mathcal{S}'_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P}$, and $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{Q})$ since $\mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$. Hence, $tt' \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$, which implies $t \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ as $t' \in (\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})^*$, but this is contradictory.

Now suppose $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \,||\, \mathcal{S}_\mathcal{Q}) \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}$ and there is no strict prefix of t for which this holds. Then there exists $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})^*$ such that $tt' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}}$. Consequently, without loss of generality, $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$, which means $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$. From $\mathcal{S}'_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P}$, it follows that $t \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}'_\mathcal{P})$, and so $t \in \mathcal{R}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}}$. Hence $t \in \mathsf{violations}(\mathcal{S}'_\mathcal{P} \,||\, \mathcal{S}'_\mathcal{Q})$. □

In this theorem, the condition $\mathcal{A}^I_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = \emptyset$ ensures compatibility of $\mathcal{S}'_\mathcal{P} \,||\, \mathcal{S}'_\mathcal{Q}$ and $\mathcal{S}_\mathcal{P} \,||\, \mathcal{S}_\mathcal{Q}$, which does not necessarily follow from $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P}$, along with $\mathcal{S}_\mathcal{Q}$ and $\mathcal{S}'_\mathcal{Q}$, agreeing. The remaining condition is standard for compositionality of parallel composition (cf [dAH01]), and ensures that, for any trace $t \in (\mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}})^*$, $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} = t \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}}$ and $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = t \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{Q}}$. Based on the monotonicity result, a sound and complete AG rule can be formulated for parallel composition.

Theorem 5.19. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}_\mathcal{P}$, $\mathcal{S}_\mathcal{Q}$ and $\mathcal{S}$ be contracts such that $\mathcal{A}_\mathcal{P} \cap \mathcal{A}_\mathcal{Q} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Safe-Parallel}\ \frac{\mathcal{P} \models \mathcal{S}_\mathcal{P} \qquad \mathcal{Q} \models \mathcal{S}_\mathcal{Q} \qquad \mathcal{S}_\mathcal{P} \,||\, \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}}{\mathcal{P} \,||\, \mathcal{Q} \models \mathcal{S}}.$$

Proof. For soundness, we know $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{Q}$. By the theorem conditions, the conditions for Theorem 5.18 are satisfied, so $\mathcal{AG}(\mathcal{P}) \,||\, \mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{P} \,||\, \mathcal{S}_\mathcal{Q}$. From compatibility of $\mathcal{P} \,||\, \mathcal{Q}$ and $\mathcal{S}$, we obtain $\mathcal{AG}(\mathcal{P}) \,||\, \mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}$ by weak transitivity. Now by Lemma 5.20 (below), we derive $\mathcal{AG}(\mathcal{P} \,||\, \mathcal{Q}) \sqsubseteq \mathcal{S}$ by transitivity, given that the alphabets of $\mathcal{AG}(\mathcal{P} \,||\, \mathcal{Q})$ coincide with those of $\mathcal{AG}(\mathcal{P}) \,||\, \mathcal{AG}(\mathcal{Q})$.

For completeness, take $\mathcal{S}_\mathcal{P} = \mathcal{AG}(\mathcal{P})$ and $\mathcal{S}_\mathcal{Q} = \mathcal{AG}(\mathcal{Q})$. Then, by transitivity, the result follows from Lemma 5.20. □

Lemma 5.20. $\mathcal{AG}(\mathcal{P} \mid\mid \mathcal{Q}) \sqsubseteq \mathcal{AG}(\mathcal{P}) \mid\mid \mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{AG}(\mathcal{P} \mid\mid \mathcal{Q})$.

Proof. First suppose that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})}$ and $t \notin \mathsf{violations}(\mathcal{AG}(\mathcal{P}) \mid\mid \mathcal{AG}(\mathcal{Q}))$. Then $t \restriction \mathcal{A}_\mathcal{P} \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$, which implies that $t \restriction \mathcal{A}_\mathcal{P} \notin F_{\mathcal{E}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \notin F_{\mathcal{E}(\mathcal{Q})}$. Hence, $t \notin F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$, from which it follows that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}$. For the other direction, suppose $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}$ and $t \notin \mathsf{violations}(\mathcal{AG}(\mathcal{P} \mid\mid \mathcal{Q}))$. Then, $t \in \mathcal{G}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}$, which implies $t \in T_{\mathcal{P}||\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$, which means that $t \restriction \mathcal{A}_\mathcal{P} \notin F_{\mathcal{E}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \notin F_{\mathcal{E}(\mathcal{Q})}$ i.e., $t \restriction \mathcal{A}_\mathcal{P} \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$. From this it follows that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})}$, having noticed that no output extension of t can violate this constraint.

For the violations set containments, suppose that $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}) \mid\mid \mathcal{AG}(\mathcal{Q}))$ and $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})} \cap \mathcal{R}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}$. Thus, there exists $t' \in (\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}})^*$ such that $tt' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})}}$. Consequently, without loss of generality, $tt' \restriction \mathcal{A}_\mathcal{P} \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$, which implies $t \restriction \mathcal{A}_\mathcal{P} \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$. Suppose for a contradiction that $t \in \mathcal{G}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}$. Then $t \in T_{\mathcal{P}||\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$, which implies $t \restriction \mathcal{A}_\mathcal{P} \in T_\mathcal{P}$. But, as $t \restriction \mathcal{A}_\mathcal{P} \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$, it follows that $\mathcal{P} \not\models \mathcal{AG}(\mathcal{P})$, which is contradictory. Therefore, $t \notin \mathcal{G}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}$ and so $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P} \mid\mid \mathcal{Q}))$.

For the other direction of the containment, suppose $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P} \mid\mid \mathcal{Q}))$ and $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})} \cap \mathcal{R}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}$. Then there exists $t' \in (\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}})^*$ such that $tt' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P}||\mathcal{Q})}}$. Hence, $tt' \notin T_{\mathcal{P}||\mathcal{Q}} \cup F_{\mathcal{E}(\mathcal{P}||\mathcal{Q})}$, which implies without loss of generality that $tt' \restriction \mathcal{A}_\mathcal{P} \notin T_\mathcal{P} \cup F_{\mathcal{E}(\mathcal{P})}$. Hence, $tt' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})}}$, which implies $tt' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$. Therefore, $t \restriction \mathcal{A}_\mathcal{P} \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$, which implies $t \notin \mathcal{G}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})}$. Consequently, $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}) \mid\mid \mathcal{AG}(\mathcal{Q}))$ as we are assuming that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||\mathcal{AG}(\mathcal{Q})}$. □

Abadi and Lamport [AL93] show soundness of their parallel composition rule, while Maier [Mai03] demonstrates that compositional circular AG rules cannot be both sound and complete. Namjoshi and Trefler [NT10] include a circular sound and complete rule for parallel composition, but it is not compositional. These results seem at odds with our rule, but in our setting circularity is broken, since a safety property cannot be simultaneously violated by two or more components. This is due to an output being under the control of at most one component.

5.1.3 Conjunction

In this section, we define a conjunctive operator on contracts for combining independently developed requirements. From this, we show that the operator is compositional and corresponds to the meet operation on the refinement relation. This allows us to conclude that implementations of a conjunctive contract must be implementations of both contracts to be conjoined. Based on this, we formulate a sound and complete AG rule for conjunction.

Definition 5.21. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ be contracts composable for conjunction. Then $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}, \mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}, \mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} \rangle$ defined by:

- $\mathcal{A}^I_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} = \mathcal{A}^I_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$
- $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} = \mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$
- $\mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} = \big(\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathcal{R}_{\mathcal{S}_\mathcal{Q}}\big) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$
- $\mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$ is the intersection of the following sets:
  - $\mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$
  - $\overline{\mathsf{violations}(\mathcal{S}_\mathcal{P})} \cup (\overline{\mathsf{violations}(\mathcal{S}_\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}})$
  - $\overline{\mathsf{violations}(\mathcal{S}_\mathcal{Q})} \cup (\overline{\mathsf{violations}(\mathcal{S}_\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{S}_\mathcal{P}})$. ◇

The assumption $\mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$ encompasses all of the assumptions made by either $\mathcal{S}_\mathcal{P}$ or $\mathcal{S}_\mathcal{Q}$ (restricted to $(\mathcal{A}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}})^*$), while the guarantee $\mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$ is the largest subset of $\mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$ that cannot violate the guarantees of $\mathcal{S}_\mathcal{P}$ or $\mathcal{S}_\mathcal{Q}$.

The next theorem shows that our definition of conjunction corresponds to the meet operator on the refinement relation, and is compositional under refinement. Consequently, the set of implementations for $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$ is the intersection of the implementation sets for $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, which means that $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$ is only implementable providing $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ share a common implementation. In this AG framework for safety, if both $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ are implementable, then $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$ is implementable.

Theorem 5.22. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for conjunction. Then:

- $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}_\mathcal{R} \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{R} \sqsubseteq \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}_\mathcal{R} \sqsubseteq \mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}'_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}'_\mathcal{P} \wedge \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$.

Proof. First show that $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P}$. Suppose $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$. Then there is a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$ and $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$. Therefore, $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}}$, implying $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q})$. If $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$, then $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$ as required. By similar reasoning $\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$.

For the second claim, suppose $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}) \cap \mathcal{A}^*_\mathcal{R}$. Then there is a prefix t' of t and $t'' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}})^*$ such that $t't'' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}}$. So, without loss of generality, $t't'' \notin \overline{\mathsf{violations}(\mathcal{S}_\mathcal{P})} \cup \overline{\mathsf{violations}(\mathcal{S}_\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$. Therefore, there is a prefix $t_1 \in \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$ of t' such that $t_1 \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$. Hence, $t_1 \in \mathsf{violations}(\mathcal{S}_\mathcal{R})$ by $\mathcal{S}_\mathcal{R} \sqsubseteq \mathcal{S}_\mathcal{P}$, and so $t \in \mathsf{violations}(\mathcal{S}_\mathcal{R})$ as

required. Now suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{R}}$. Then without loss of generality, $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{R}}$, so from $\mathcal{S}_\mathcal{R} \sqsubseteq \mathcal{S}_\mathcal{P}$, we derive $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{R})$.

For the third claim, by the first claim we have $\mathcal{S}'_\mathcal{P} \wedge \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P} \wedge \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}'_\mathcal{Q}$. Now by transitivity, we see that $\mathcal{S}'_\mathcal{P} \wedge \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P} \wedge \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$ providing $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^I_{\mathcal{S}'_\mathcal{Q}} = \emptyset$ and $\mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^I_{\mathcal{S}'_\mathcal{P}} = \emptyset$, so by the second claim, it follows that $\mathcal{S}'_\mathcal{P} \wedge \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$ as required. If either of the compatibility conditions are not satisfied, we can obtain new contracts $\mathcal{S}''_\mathcal{P}$ for $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}''_\mathcal{Q}$ for $\mathcal{S}_\mathcal{Q}$ that have output set $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$ and contain all traces from the respective contracts, except for those with an output in $(\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \setminus \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}) \cup (\mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \setminus \mathcal{A}^O_{\mathcal{S}_\mathcal{P}})$ that has been removed from the interface. It is straightforward to show that $\mathcal{S}''_\mathcal{P} \wedge \mathcal{S}''_\mathcal{Q} = \mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}$. □

From these strong algebraic properties, we can formulate an AG rule for conjunction that is both sound and complete.

Theorem 5.23. Let $\mathcal{P}$ be a component, and let $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}$ be contracts such that $\mathcal{A}^I_\mathcal{P} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Safe-Conjunction}\frac{\mathcal{P} \models \mathcal{S}_1 \qquad \mathcal{P} \models \mathcal{S}_2 \qquad \mathcal{S}_1 \wedge \mathcal{S}_2 \sqsubseteq \mathcal{S}}{\mathcal{P} \models \mathcal{S}}.$$

Proof. For soundness, note by the second claim of Theorem 5.22 that $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}_1 \wedge \mathcal{S}_2$. Hence $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}$, as the compatibility constraint for weak transitivity is satisfied. For completeness, the result follows by idempotence of conjunction, having taken $\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{S}$. □

Example 5.24. A Client is assumed to have an interface that can send jobs to, and await acknowledgements from, a server, can login once instructed by a user, and can logout when it pleases. Thus, job and logout are outputs, whereas login and ack are inputs. The combined effect of Client and Server should satisfy the properties:

- Spec1: If the observed behaviour over login and logout is always a prefix of $\langle \mathsf{login}, \mathsf{logout} \rangle^*$, then login and process should alternate.
- Spec2: If the observed behaviour over login and logout is always a prefix of $\langle \mathsf{login}, \mathsf{logout} \rangle^*$, then process and logout should alternate.

Spec1 and Spec2 are represented by the contracts $\langle \mathcal{R}_{\mathsf{Spec}}, \mathcal{G}_{\mathsf{Spec1}} \rangle$ and $\langle \mathcal{R}_{\mathsf{Spec}}, \mathcal{G}_{\mathsf{Spec2}} \rangle$ respectively, as depicted in Figure 5.5. The combined effect of these properties is given by the conjunctive contract $\mathsf{Spec1} \wedge \mathsf{Spec2} = \langle \mathcal{R}_{\mathsf{Spec}}, \mathcal{G}_{\mathsf{Spec1} \wedge \mathsf{Spec2}} \rangle$, the guarantee of which is presented in Figure 5.6. As Spec1 and Spec2 have the same interface, the guarantee of the conjunction is obtained as the intersection of $\mathcal{G}_{\mathsf{Spec1}}$ and $\mathcal{G}_{\mathsf{Spec2}}$. ◇

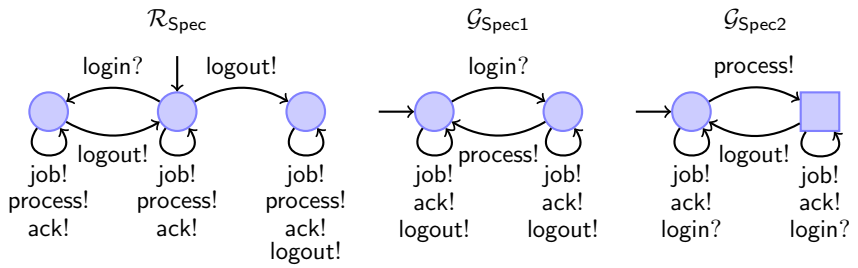

Figure 5.5: Assumption and guarantee of Spec1 and Spec2

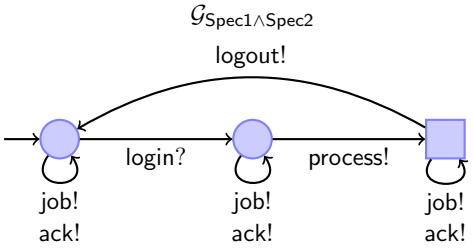

Figure 5.6: Guarantee of Spec1 $\wedge$ Spec2

5.1.4 Disjunction

In this section, we formulate a disjunctive operator on contracts. Whereas conjunction combines requirements in the sense that it strengthens guarantees, disjunction strengthens the assumptions on the environment to the extent that the implementations of the disjunction contains the union of the implementations of the contracts to be composed. Being the dual of conjunction, we show that disjunction is the join operator on the refinement preorder, and provide a sound and complete assume-guarantee rule.

Definition 5.25. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ be contracts composable for disjunction. Then $\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}, \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}, \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} \rangle$ defined by:

- $\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} = \mathcal{A}^I_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$
- $\mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} = \mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$
- $\mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$ is the intersection of the following sets:
  - $\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{P}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{P})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}})$
  - $\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{Q}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{Q})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{P}})$
- $\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} = \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} \cap (\overline{\mathsf{violations}(\mathcal{S}_\mathcal{P})} \cup \overline{\mathsf{violations}(\mathcal{S}_\mathcal{Q})})$. ◇

This definition of disjunction satisfies properties similar to those for conjunction, and hence is the join operator on the refinement preorder.

Theorem 5.26. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for disjunction. Then:

- $\mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{R}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{R}$ implies $\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{R}$
- $\mathcal{S}'_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}'_\mathcal{P} \vee \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$.

Proof. First show that $\mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$. Suppose $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$. Then there is a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}}$. Hence $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$ as required. If instead $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$, then $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{P})$. Showing $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$ is similar.

For the second claim, suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. If $t \equiv \epsilon$, then $\epsilon \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$ trivially, while if $t \equiv t'o$ for $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$, then $t'o \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$ by the induction hypothesis and output extendability of assumptions or extendability of violations. Instead, if $t \equiv t'i$ for $i \in \mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$, then by the induction hypothesis in the difficult case we have $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \overline{\mathsf{violations}(\mathcal{S}_\mathcal{P})}$ and $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cap \overline{\mathsf{violations}(\mathcal{S}_\mathcal{Q})}$. As $i \in \mathcal{A}^I_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$, it follows from $\mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{R}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{R}$ that $t'i \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{R}_{\mathcal{S}_\mathcal{Q}}$. Hence, $t'i \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$.

Now suppose that $t \in \mathsf{violations}(\mathcal{S}_\mathcal{R}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. Then there exists a smallest prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cap \mathsf{violations}(\mathcal{S}_\mathcal{R}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. Suppose all strict prefixes of t' are not in $\mathsf{violations}(\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q})$. Then, by the previous part, it follows that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. If $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$, then from $\mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{R}$ it follows that $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$, and if $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$, then from $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{R}$ it follows that $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{Q})$. Hence $t' \notin \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$ (noting $\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}^*_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$), which implies $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q})$. By extension closure of violations, we have $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q})$.

For the third claim, by the first claim we have that $\mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$. Since the contracts under consideration are composable for disjunction, it follows from $\mathcal{S}'_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{Q}$, along with transitivity (compatibility holds), that $\mathcal{S}'_\mathcal{P} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$. Now by the second claim it is straightforward to derive $\mathcal{S}'_\mathcal{P} \vee \mathcal{S}'_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$. □

Based on the algebraic properties of disjunction, we can formulate a sound and complete assume-guarantee rule. This demonstrates that a disjunctive contract contains the union of the implementations of the contracts to be composed, although there may be additional implementations that are not implementations of either contract.

Theorem 5.27. Let $\mathcal{P}$ be a component, and let $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}$ be contracts such that $\mathcal{S}_1$ and $\mathcal{S}_2$ are composable for disjunction, and $\mathcal{A}^I_\mathcal{P} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Safe-Disjunction}\ \frac{\mathcal{P} \models \mathcal{S}_1 \quad \text{or} \quad \mathcal{P} \models \mathcal{S}_2 \qquad \mathcal{S}_1 \vee \mathcal{S}_2 \sqsubseteq \mathcal{S}}{\mathcal{P} \models \mathcal{S}}.$$

Proof. For soundness, assume $\mathcal{P} \models \mathcal{S}_1$. Then $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}_1$ and $\mathcal{S}_1 \sqsubseteq \mathcal{S}_1 \vee \mathcal{S}_2$ by Theorem 5.26.

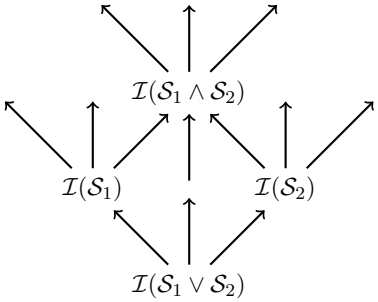

Figure 5.7: Implementations of $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_1 \wedge \mathcal{S}_2$ and $\mathcal{S}_1 \vee \mathcal{S}_2$

Since $\mathcal{A}^I_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}} = \emptyset$, it follows that transitivity holds, and so $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}$, implying $\mathcal{P} \models \mathcal{S}$. For completeness, take $\mathcal{S}_1 = \mathcal{S}_2 = \mathcal{S}$. The result then holds by idempotence of $\vee$. □

The disjunction $\mathcal{S}_1 \vee \mathcal{S}_2$ is the strongest contract containing the union of the implementations for $\mathcal{S}_1$ and $\mathcal{S}_2$. In contrast to conjunction, which precisely characterises the intersection of the implementation sets, there may be implementations of the disjunction that are not implementations of either $\mathcal{S}_1$ or $\mathcal{S}_2$. The Hasse diagram of Figure 5.7 makes this relationship clear by depicting the least refined implementations of the contracts $\mathcal{S}_1$ and $\mathcal{S}_2$, along with their conjunction and disjunction. The implementations of a contract $\mathcal{S}$ are simply those implementations that appear above (i.e., can be reached from) $\mathcal{I}(\mathcal{S})$.

5.1.5 Quotient

The AG rule for parallel composition in Theorem 5.19 makes use of the composition $\mathcal{S}_{\mathcal{P}} \parallel \mathcal{S}_{\mathcal{Q}}$. To support incremental development, we need a way of decomposing the composition to find $\mathcal{S}_{\mathcal{Q}}$ given $\mathcal{S}_{\mathcal{P}}$. We can do this using a quotient operator.

Definition 5.28. Let $\mathcal{S}_{\mathcal{P}}$ and $\mathcal{S}_{\mathcal{W}}$ be contracts. Then the quotient $\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}, \mathcal{A}^O_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}, \mathcal{R}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}, \mathcal{G}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}} \rangle$, defined only when $\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}} \subseteq \mathcal{A}^O_{\mathcal{S}_{\mathcal{W}}}$, where:

- $\mathcal{A}^I_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}} = \mathcal{A}^I_{\mathcal{S}_{\mathcal{W}}} \setminus \mathcal{A}^I_{\mathcal{S}_{\mathcal{P}}}$
- $\mathcal{A}^O_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}} = \mathcal{A}^O_{\mathcal{S}_{\mathcal{W}}} \setminus \mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}}$
- $\mathcal{R}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}} = [\mathcal{R}_{\mathcal{S}_{\mathcal{W}}} \cap (\overline{\mathsf{violations}(\mathcal{S}_{\mathcal{P}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}})] \restriction \mathcal{A}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}$
- $\mathcal{G}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}$ is the largest subset of $\mathcal{R}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}$ disjoint from $[\mathcal{R}_{\mathcal{S}_{\mathcal{W}}} \cap (\overline{\mathsf{violations}(\mathcal{S}_{\mathcal{P}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}}) \cap (\mathsf{violations}(\mathcal{S}_{\mathcal{W}}) \cup (\overline{\mathcal{R}_{\mathcal{S}_{\mathcal{P}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}}))] \restriction \mathcal{A}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}$. ◇

Although not immediately obvious from the formulation of the previous definition, the assumption is closed under output-extensions, and the assumption and guarantee are both prefix-closed. Therefore, the quotient is a well-formed contract. Before explaining the intuition behind

the definition, we introduce the following theorem, which shows that the quotient operator on contracts yields the weakest decomposition of the parallel composition.

Theorem 5.29. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts. Then there exists a contract $\mathcal{S}_\mathcal{Q}$ such that $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}$ iff the following properties hold:

- The quotient $\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ is defined
- $\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}) \sqsubseteq \mathcal{S}_\mathcal{W}$
- $\mathcal{A}^I_{\mathcal{S}_\mathcal{Q}} = \mathcal{A}^I_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$ implies $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$.

Proof. For the first claim, if $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}$, then $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = \mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{W}}$, which implies $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{W}}$. Now suppose that $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{W}}$. Then we construct a contract $\mathcal{S}_\mathcal{Q} = \langle \mathcal{A}^I_{\mathcal{S}_\mathcal{W}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{W}} \setminus \mathcal{A}^O_{\mathcal{S}_\mathcal{P}}, \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}, \emptyset \rangle$, which, having no implementations, implies $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q}$ has no implementations. The constraints R1 to R3 are satisfied, so $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}$ as required.

For the second claim, suppose $t \in \mathcal{R}_{\mathcal{S}_\mathcal{W}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$. If $t \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$, then there exists a prefix t' of t and $t'' \in (\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})})^*$ such that $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$ or $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$, and $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{P})$ and $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$. It follows that $t't'' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$, so $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$, which means $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$. Therefore, $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathcal{G}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$, which implies $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$. But this contradicts $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$. Hence $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$.

Now suppose that $t \in \mathsf{violations}(\mathcal{S}_\mathcal{W}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$. Then, there exists a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}} \cap \mathsf{violations}(\mathcal{S}_\mathcal{W})$. By the previous part, it follows that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$. Now suppose for a contradiction that $t' \in \mathcal{G}_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$. Then $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{P})$ and $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$. But it follows that $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$, since $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}}$. This contradicts $t' \in \mathcal{G}_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$. Hence $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}))$ and so $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}))$.

For the third claim, suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$. Then there exists $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{W}}$ such that $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} = t$ with $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ and $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{P})$. From $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ we derive $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q})$, given that $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}$. If $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$, then it follows that $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{violations}(\mathcal{S}_\mathcal{Q})$. If instead $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q})$, then it follows that $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{violations}(\mathcal{S}_\mathcal{Q})$. Note that $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = t$.

Now suppose that $t \in \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$. Then there exists t' a prefix of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cap \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$. So there is a prefix and input extension t'' of t' such that there exists $t_w \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ with $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} = t''$, $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{P})$, and either $t_w \in \mathsf{violations}(\mathcal{S}_\mathcal{W})$ or $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$. If $t_w \in \mathsf{violations}(\mathcal{S}_\mathcal{W})$, then $t_w \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q})$, since $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}$. Therefore, it follows that $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{violations}(\mathcal{S}_\mathcal{Q})$. Alternatively, if $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$, then if $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{Q})$ it follows that $t_w \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$. Since

$\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}$, it must hold that $t_w \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q})$, which again implies $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{violations}(\mathcal{S}_\mathcal{Q})$. Note that $t_w \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = t''$, so $t \in \mathsf{violations}(\mathcal{S}_\mathcal{Q})$. □

In explaining the intuition behind the definition of quotient, it is necessary to consider the properties of Theorem 5.29 along with the formulation of refinement and parallel composition (Definitions 5.9 and 5.15). To obtain the least refined solution $\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ for $\mathcal{S}_\mathcal{P} \mid\mid X \sqsubseteq \mathcal{S}_\mathcal{W}$, it is essential that the quotient roughly[1] satisfies the following properties for $t \in \mathcal{A}^*_{\mathcal{S}_\mathcal{W}}$:

- If $t \in \mathsf{violations}(\mathcal{S}_\mathcal{W})$ and:
  - $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$, then $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}))$, so there is no need for $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$
  - $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{P})$, then it must hold that $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$ (i.e., take $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}}$ so that $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}))$.
- If $t \in \mathcal{R}_{\mathcal{S}_\mathcal{W}} \setminus \mathsf{violations}(\mathcal{S}_\mathcal{W})$, first attempt to ensure that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})} \setminus \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}))$ holds, and failing that ensure $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}))$:
  - If $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{P})$, then $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$, so there is no need for $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$.
  - If $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{P})$ and $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{P}}$, simply take $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$, so that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})} \setminus \mathsf{violations}(\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}))$.
  - If $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{violations}(\mathcal{S}_\mathcal{P})$ and $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$, then we require $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$, so take $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}}$.

Note that, in the definition of $\mathcal{G}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$, the set required to be disjoint from $\mathcal{R}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$ essentially characterises a subset of traces that must be in $\mathsf{violations}(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$. Furthermore, in the definition of quotient, the set of inputs $\mathcal{A}^I_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$ is taken to be the smallest set such that $\mathcal{A}^I_{\mathcal{S}_\mathcal{W}} \subseteq \mathcal{A}^I_{\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$, the latter being a necessary condition for $\mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}) \sqsubseteq \mathcal{S}_\mathcal{W}$. Yet, in fact, the set of inputs for quotient can be parameterised without affecting the results of Theorem 5.29. This is useful, since enlarging the set of inputs allows for the possibility of the quotient to observe the behaviour of $\mathcal{S}_\mathcal{P}$, which yields a contract with more specific behaviour. Such a contract cannot be obtained through refinement alone, as $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ does not imply $\mathcal{S}_\mathcal{P} \mid\mid \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$ in general, since monotonicity only holds on a restricted set of interfaces (cf Theorem 5.18).

We now present a sound and complete AG rule for quotient on contracts.

Theorem 5.30. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts such that $\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ is defined, let $\mathcal{P}$ range over components having the same interface as $\mathcal{S}_\mathcal{P}$, and let $\mathcal{Q}$ be a component having the same interface as

[1] Exceptions need to be made since the conditions are not mutually exclusive, and properties like prefix closure and output-extendability must be maintained.

$\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ (where the quotient is parameterised on the set $\mathcal{A}^I_\mathcal{Q}$). Then the following AG rule is both sound and complete:

$$\textsc{Safe-Quotient}\ \frac{\forall \mathcal{P} \cdot \mathcal{P} \models \mathcal{S}_\mathcal{P} \text{ implies } \mathcal{P} \mid\mid \mathcal{Q} \models \mathcal{S}_\mathcal{W}}{\mathcal{Q} \models \mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}.$$

Proof. For soundness, first note that $\mathcal{I}(\mathcal{S}_\mathcal{P}) \models \mathcal{S}_\mathcal{P}$, and so $\mathcal{I}(\mathcal{S}_\mathcal{P}) \mid\mid \mathcal{Q} \models \mathcal{S}_\mathcal{W}$. Consequently, $\mathcal{AG}(\mathcal{I}(\mathcal{S}_\mathcal{P}) \mid\mid \mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}$, and from the proof of Theorem 5.19 we know that $\mathcal{AG}(\mathcal{I}(\mathcal{S}_\mathcal{P})) \mid\mid \mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}$. Moreover, $\mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{AG}(\mathcal{I}(\mathcal{S}_\mathcal{P})) \sqsubseteq \mathcal{S}_\mathcal{P}$, so by Theorem 5.29 it follows that $\mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ as required.

For completeness, by the interfaces of $\mathcal{P}$ and $\mathcal{S}_\mathcal{P}$, as well as $\mathcal{Q}$ and $\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$, matching, it follows that if $\mathcal{AG}(\mathcal{P}) \sqsubseteq \mathcal{S}_\mathcal{P}$, then $\mathcal{AG}(\mathcal{P}) \mid\mid \mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{P} \mid\mid (\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})$, since the conditions for monotonicity in Theorem 5.18 are satisfied. Now by transitivity (the conditions being trivially satisfied) and Theorem 5.29, we obtain $\mathcal{AG}(\mathcal{P}) \mid\mid \mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}$. Hence $\mathcal{AG}(\mathcal{P} \mid\mid \mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}$ by Lemma 5.20. □

We insist that the components $\mathcal{P}$ and $\mathcal{Q}$ must have the same interfaces as their respective contracts, since parallel composition is only monotonic when restrictions are placed on the interfaces of the contracts to be composed (cf Theorem 5.18). The proof of the rule hints that the universal quantification over all components $\mathcal{P}$ can be replaced by the single component $\mathcal{I}(\mathcal{S}_\mathcal{P})$, meaning that it is not necessary to quantify over an infinite number of components in order to satisfy the premise.

Corollary 5.31. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts such that $\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ is defined, and let $\mathcal{Q}$ be a component having the same interface as $\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ (where the quotient is parameterised on the set $\mathcal{A}^I_\mathcal{Q}$). Then the following AG rule is both sound and complete:

$$\textsc{Safe-Quotient-Revised}\ \frac{\mathcal{I}(\mathcal{S}_\mathcal{P}) \mid\mid \mathcal{Q} \models \mathcal{S}_\mathcal{W}}{\mathcal{Q} \models \mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}.$$

Proof. For soundness, note that $\mathcal{AG}(\mathcal{I}(\mathcal{S}_\mathcal{P}) \mid\mid \mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}$, which by Lemma 5.20 yields $\mathcal{AG}(\mathcal{I}(\mathcal{S}_\mathcal{P})) \mid\mid \mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}$. As $\mathcal{AG}(\mathcal{I}(\mathcal{S}_\mathcal{P})) \sqsubseteq \mathcal{S}_\mathcal{P} \sqsubseteq \mathcal{AG}(\mathcal{I}(\mathcal{S}_\mathcal{P}))$, it follows by Theorem 5.29 that $\mathcal{AG}(\mathcal{Q}) \sqsubseteq \mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ given $\mathcal{AG}(\mathcal{Q})$ and $\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}$ have identical interfaces. Completeness follows by Theorem 5.30. □

Example 5.32. We now derive a Client contract (having an interface as described in Example 5.24) that can interact with Server (Figure 5.1), whilst satisfying the requirements of Spec1 ∧ Spec2 (Figures 5.5 and 5.6). This is obtained as (Spec1 ∧ Spec2)/Server, where the quotient operator is parameterised on the set of inputs $\{\mathsf{login}, \mathsf{ack}\}$. The resulting contract is shown in Figure 5.8. The guarantee is obtained from the assumption by pruning any trace that is in violations(Spec1 ∧ Spec2)

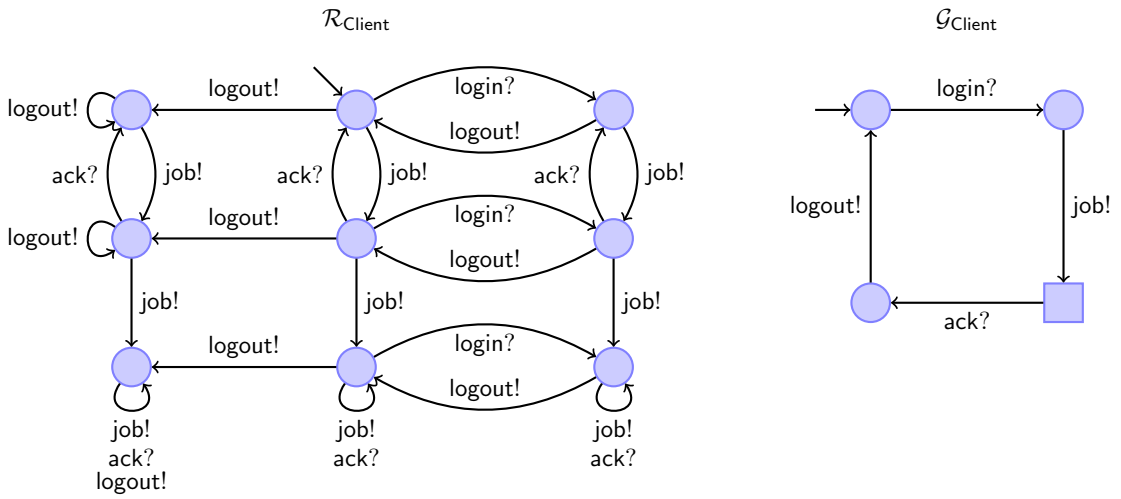

Figure 5.8: Assumption and guarantee of Client

or not in $\mathcal{R}_{\mathsf{Server}}$. The most general implementation of Client has the same pictorial representation as $\mathcal{G}_{\mathsf{Server}}$, although there are implicit inconsistent input transitions from each state in order to ensure input-receptivity. ◇

5.1.6 Decomposing Parallel Composition

The following corollary shows how we can revise the AG rule for parallel composition so that it makes use of quotient on contracts. This is useful for system development, as we will often have the specification of a whole system, rather than the specifications of the subsystems to be composed.

Corollary 5.33. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}_\mathcal{P}$, $\mathcal{S}_\mathcal{Q}$ and $\mathcal{S}$ be contracts such that $\mathcal{A}_\mathcal{P} \cap \mathcal{A}_\mathcal{Q} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. When the quotient is parameterised on $\mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$, the following rule is both sound and complete:

$$\textsc{Safe-Parallel-Decompose}\ \frac{\mathcal{P} \models \mathcal{S}_\mathcal{P} \qquad \mathcal{Q} \models \mathcal{S}_\mathcal{Q} \qquad \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}/\mathcal{S}_\mathcal{P}}{\mathcal{P} \parallel \mathcal{Q} \models \mathcal{S}}.$$

Proof. Follows immediately from Theorems 5.19 and 5.29. □

This rule, based on Theorem 5.19, differs in having the premise $\mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}/\mathcal{S}_\mathcal{P}$ in place of $\mathcal{S}_\mathcal{P} \parallel \mathcal{S}_\mathcal{Q} \sqsubseteq \mathcal{S}$. Note that this substitution requires no change to the constraints on the contracts and components. The rule is useful for scenarios when the contract $\mathcal{S}$ is supplied along with a subcontract $\mathcal{S}_\mathcal{P}$ (or for when a subcontract $\mathcal{S}_\mathcal{P}$ can easily be inferred). In such circumstances, the missing contract $\mathcal{S}_\mathcal{Q}$ can be taken as any refinement of $\mathcal{S}/\mathcal{S}_\mathcal{P}$.

5.2 Assume-Guarantee Framework with Progress

In this section, we introduce an AG framework for reasoning about the safety and liveness properties satisfied by components. Since we continue to work with finite-length traces, our notion of liveness corresponds to progress, which is based on quiescence, as in Section 3.2. A trace is said to be quiescent if it can result in an observable behaviour from which the component cannot produce an output without first having to receive input from the environment. Quiescence has similarities with, although is not equivalent to, deadlock. In the case of the latter, a trace is said to be deadlocked just if there is some behaviour that is both unable to produce any output and is unwilling to accept any inputs.

To instil this notion of progress within our framework, we extend contracts by including a set of liveness traces. These are traces on which any implementing component may not become quiescent.

Definition 5.34. A *progress-sensitive contract* $\mathcal{S}$ is a tuple $\langle \mathcal{A}_\mathcal{S}^I, \mathcal{A}_\mathcal{S}^O, \mathcal{R}_\mathcal{S}, \mathcal{G}_\mathcal{S}, \mathcal{L}_\mathcal{S} \rangle$, in which $\mathcal{A}_\mathcal{S}^I$ and $\mathcal{A}_\mathcal{S}^O$ are disjoint sets (whose union is $\mathcal{A}_\mathcal{S}$), referred to as the inputs and outputs respectively, $\mathcal{R}_\mathcal{S}$ and $\mathcal{G}_\mathcal{S}$ are prefix closed subsets of $\mathcal{A}_\mathcal{S}^*$, referred to as the assumption and guarantee respectively, such that $t \in \mathcal{R}_\mathcal{S}$ and $t' \in (\mathcal{A}_\mathcal{S}^O)^*$ implies $tt' \in \mathcal{R}_\mathcal{S}$, and $\mathcal{L}_\mathcal{S} \subseteq \mathcal{R}_\mathcal{S} \cap \mathcal{G}_\mathcal{S}$ is a (not necessarily prefix-closed) set of liveness traces. ◇

We will often drop the term 'progress-sensitive' and simply refer to $\mathcal{S}$ as a contract. As in the safety framework, the assumption is closed under output-extensions since the environment cannot constrain the output behaviour of a component, while the guarantee is not required to be closed under input-extensions, because the assumption can specify the inputs under which the guarantee is given. Note that, by taking the set of liveness traces to be the empty set, the framework supports reasoning about safety properties as in Section 5.1.

Example 5.35. We now adopt the convention that square nodes within the figure of a guarantee indicates that progress must be made, while a circular node has no such requirement. Considering the contract for the Server, as depicted in Figure 5.1, the assumption leaves process unconstrained, but ensures that error will never be sent providing job and ack alternate in that order (as in the safety setting). The guarantee requires that any job received can only be acknowledged after having been processed, a new job can only arrive after the previous one has been acknowledged, and whenever a job is received it must be processed (the progress condition). ◇

The inclusion of liveness strengthens the definition of satisfaction, meaning that a progress-sensitive contract will, in general, have fewer implementations than a contract disregarding liveness.

Definition 5.36. A progress-sensitive component $\mathcal{P}$ *satisfies* the contract $\mathcal{S}$, written $\mathcal{P} \models_l \mathcal{S}$, iff $\mathcal{P} \models \mathcal{S}$ (as in Definition 5.1) and $\mathcal{L}_\mathcal{S} \cap T_\mathcal{P} \subseteq \overline{K_\mathcal{P}}$. ◇

Therefore, any implementation $\mathcal{P}$ of $\mathcal{S}$ must not be allowed to become inconsistent under its own control when offered inputs in the assumption, and any trace of $\mathcal{P}$ that is contained in $\mathcal{L}_{\mathcal{S}}$ must make observational progress.

Lemma 5.37. Let $\mathcal{P}$ and $\mathcal{Q}$ be progress-sensitive components, and let $\mathcal{S}$ be a contract. If $\mathcal{P} \models_l \mathcal{S}$, $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{P}$, and $\mathcal{A}^I_{\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{S}} = \emptyset$, then $\mathcal{Q} \models_l \mathcal{S}$.

Proof. We need to show that $\mathcal{L}_{\mathcal{S}} \cap T_{\mathcal{Q}} \subseteq \overline{K_{\mathcal{Q}}}$, while the remainder of the result follows from Lemma 5.4. Suppose that $t \in \mathcal{L}_{\mathcal{S}} \cap T_{\mathcal{Q}}$. Then $t \in \mathcal{A}^*_{\mathcal{P}}$, so, from $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{P}$, we have $t \in T_{\mathcal{E}(\mathcal{P})}$. If $t \in T_{\mathcal{P}} \setminus F_{\mathcal{E}(\mathcal{P})}$, then from $\mathcal{P} \models_l \mathcal{S}$ we derive $t \in \overline{K_{\mathcal{P}}}$, thus $t \in \overline{K_{\mathcal{Q}}}$ from $\mathcal{Q} \sqsubseteq^l_{imp} \mathcal{P}$. If instead $t \in F_{\mathcal{E}(\mathcal{P})}$, then, by the same reasoning as in Lemma 5.4, we see that $\mathcal{P} \not\models_l \mathcal{S}$. □

Example 5.38. In the progress-sensitive setting, ServerImpl (Figure 5.2) is still an implementation of Server, as is ServerImpl2 given ServerImpl2 $\sqsubseteq^{\mathsf{l}}_{\mathsf{imp}}$ ServerImpl. NonImpl is still not an implementation, because NonImpl $\not\models$ Server due to $\langle \mathsf{ack} \rangle \in \mathsf{violations}(\mathsf{Server}) \cap T_{\mathsf{NonImpl}}$ as in the safety setting, and secondly because progress is not made after receiving a job. Note that, if the square node in ServerImpl2 was circular, this component would also not be an implementation of Server, since by non-determinism there could be a behaviour of the component that does not perform process after receiving a job. ◇

We now show how to construct the least refined component satisfying a contract. Unlike in the safety case, the progress-sensitive setting is complicated by the requirement of liveness, which can conflict with safety. We therefore define the error traces of a component, which generalises the violations set by including liveness conflicts. Traces contained in this error set cannot be in any satisfying component, because they will violate the guarantee or progress condition.

Definition 5.39. Let $\mathcal{S}$ be a contract. Then $\mathsf{error}(\mathcal{S})$ is defined as the smallest set containing $\mathsf{violations}(\mathcal{S}) \cup \{t \in \mathcal{A}^*_{\mathcal{S}} : \exists t' \in (\mathcal{A}^I_{\mathcal{S}})^* \cdot tt' \in \mathcal{L}_{\mathcal{S}} \text{ and } \forall o \in \mathcal{A}^O_{\mathcal{S}} \cdot tt'o \in \mathsf{error}(\mathcal{S})\} \cdot \mathcal{A}^*_{\mathcal{S}}$. ◇

$\mathsf{error}(\mathcal{S})$ should consist of all traces that are not in any satisfying component of $\mathcal{S}$. Therefore, $\mathsf{error}(\mathcal{S})$ consists of all traces in $\mathsf{violations}(\mathcal{S})$, along with any trace that is required to be live, but cannot be so, due to all output successors violating a safety or progress error. By reducing the allowed behaviours of satisfying components, further progress errors can be introduced, which is why $\mathsf{error}(\mathcal{S})$ is defined recursively. Note that, in the safety setting when $\mathcal{L}_{\mathcal{S}} = \emptyset$, it holds that $\mathsf{error}(\mathcal{S}) = \mathsf{violations}(\mathcal{S})$.

Naturally, $\mathsf{error}(\mathcal{S})$ can be defined as the least fixed point of the defining equation above. Therefore, $\mathsf{error}(\mathcal{S}) = \cup_{i \in \mathbb{N}} X_i$, where $X_0 = \emptyset$ and $X_{i+1} \triangleq \mathsf{violations}(\mathcal{S}) \cup \{t \in \mathcal{A}^*_{\mathcal{S}} : \exists t' \in (\mathcal{A}^I_{\mathcal{S}})^* \cdot tt' \in \mathcal{L}_{\mathcal{S}} \text{ and } \forall o \in \mathcal{A}^O_{\mathcal{S}} \cdot tt'o \in X_i\} \cdot \mathcal{A}^*_{\mathcal{S}}$.

The least refined component satisfying a contract can now be defined in a straightforward manner in terms of the error traces.

Definition 5.40. Let $\mathcal{S}$ be a contract. Then the *least refined component satisfying* $\mathcal{S}$ is the component $\mathcal{I}_l(\mathcal{S}) = \langle \mathcal{A}^I_{\mathcal{S}}, \mathcal{A}^O_{\mathcal{S}}, T_{\mathcal{I}_l(\mathcal{S})}, F_{\mathcal{I}_l(\mathcal{S})}, K_{\mathcal{I}_l(\mathcal{S})} \rangle$, where:

- $T_{\mathcal{I}_l(\mathcal{S})} = \overline{\mathsf{error}(\mathcal{S})}$
- $F_{\mathcal{I}_l(\mathcal{S})} = \overline{\mathsf{error}(\mathcal{S})} \cap \overline{\mathcal{R}_{\mathcal{S}}}$
- $K_{\mathcal{I}_l(\mathcal{S})} = \overline{\mathsf{error}(\mathcal{S})} \cap \overline{\mathcal{L}_{\mathcal{S}}}$. ◇

Since $\mathsf{violations}(\mathcal{S}) \subseteq \mathsf{error}(\mathcal{S})$, it follows that the traces of $\mathcal{I}_l(\mathcal{S})$ cannot violate the safety constraints of $\mathcal{S}$. In addition, if t is a trace of $\mathcal{I}_l(\mathcal{S})$, then no extension of t can be allowed to violate the progress conditions. Therefore, $K_{\mathcal{I}_l(\mathcal{S})}$ allows the component to be quiescent whenever it is not required to be live.

Lemma 5.41. Let $\mathcal{S}$ be a contract, and let $\mathcal{P}$ be a component. Then:

- $\mathcal{I}_l(\mathcal{S})$ is non-realisable implies $\mathcal{S}$ is non-implementable;
- $\mathcal{I}_l(\mathcal{S}) \models_l \mathcal{S}$; and
- $\mathcal{P} \models_l \mathcal{S}$ iff $\mathcal{P} \sqsubseteq^l_{imp} \mathcal{I}_l(\mathcal{S})$.

Proof. For the first claim, we show that $t \in X_i$ implies t is not a trace in any implementation of $\mathcal{S}$ for each $i \in \mathbb{N}$, where X_i is the i-th iteration of defining $\mathsf{error}(\mathcal{S})$ as a least fixed point. For $i = 0$, the result holds trivially as $X_0 = \emptyset$. So suppose that the result holds for $i = k$. Now $t \in X_{k+1}$ implies that $t \in \mathsf{violations}(\mathcal{S})$ or there is $t' \in (\mathcal{A}^I_{\mathcal{S}})^*$ such that $tt' \in \mathcal{L}_{\mathcal{S}}$ and $\forall o \in \mathcal{A}^O_{\mathcal{S}} \cdot tt'o \in X_k$. If $t \in \mathsf{violations}(\mathcal{S})$, then clearly t cannot be a trace of any implementation of $\mathcal{S}$, since condition S4 of Definition 5.3 will not be satisfied. If instead t satisfies the second property, then it follows by the induction hypothesis that tt' is a quiescent trace, which contradicts $tt' \in \mathcal{L}_{\mathcal{S}}$. Therefore, tt' cannot be a trace of any implementation of $\mathcal{S}$, and so t also cannot be a trace, by input receptiveness of components. Taking $t \equiv \epsilon$, it follows that $\mathcal{S}$ is non-implementable.

For the second claim, suppose $t \in \mathcal{R}_{\mathcal{S}} \cap T_{\mathcal{I}_l(\mathcal{S})}$. Then $t \in \mathcal{R}_{\mathcal{S}} \cap \overline{\mathsf{error}(\mathcal{S})}$, which implies $t \in \mathcal{G}_{\mathcal{S}}$. Moreover, as $t \in \mathcal{R}_{\mathcal{S}}$, it follows that $t \in \overline{F_{\mathcal{I}_l(\mathcal{S})}}$. Hence condition S4 of Definition 5.3 is satisfied. Now suppose that $t \in \mathcal{L}_{\mathcal{S}} \cap T_{\mathcal{I}_l(\mathcal{S})}$. Then clearly $t \notin K_{\mathcal{I}_l(\mathcal{S})}$ by definition, so $\mathcal{I}_l(\mathcal{S}) \models_l \mathcal{S}$.

For the third claim, the if direction follows by the previous claim and Lemma 5.37. For the only if direction, we need to show that $T_{\mathcal{E}(\mathcal{P})} \subseteq T_{\mathcal{I}_l(\mathcal{S})} \cup (T_{\mathcal{I}_l(\mathcal{S})} \uparrow \mathcal{A}^I_{\mathcal{P}})$, $F_{\mathcal{E}(\mathcal{P})} \subseteq F_{\mathcal{I}_l(\mathcal{S})} \cup (T_{\mathcal{I}_l(\mathcal{S})} \uparrow \mathcal{A}^I_{\mathcal{P}})$ and $K_{\mathcal{E}(\mathcal{P})} \subseteq K_{\mathcal{I}_l(\mathcal{S})} \cup (T_{\mathcal{I}_l(\mathcal{S})} \uparrow \mathcal{A}^I_{\mathcal{P}})$. If $t \in T_{\mathcal{E}(\mathcal{P})}$ and $t \notin \mathcal{A}^*_{\mathcal{S}}$, then there is a prefix $t'a$ of t such that $t' \in \mathcal{A}^*_{\mathcal{S}}$ and $a \in \mathcal{A}^I_{\mathcal{P}} \setminus \mathcal{A}_{\mathcal{S}}$, which by an inductive argument that assumes the result holds for all strict prefixes allows us to derive $t \in T_{\mathcal{I}_l(\mathcal{S})} \uparrow \mathcal{A}^I_{\mathcal{P}}$. So suppose that $t \in T_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}}$. Then by the first claim, since $\mathcal{P} \models_l \mathcal{S}$, it follows $t \notin \mathsf{error}(\mathcal{S})$. Hence $t \in T_{\mathcal{I}_l(\mathcal{S})}$. Now suppose that $t \in F_{\mathcal{E}(\mathcal{P})} \cap \mathcal{A}^*_{\mathcal{S}}$. Then as $\mathcal{P} \models_l \mathcal{S}$, it follows that $t \notin \mathsf{error}(\mathcal{S})$ and $t \notin \mathcal{R}_{\mathcal{S}}$. Consequently,

$t \in F_{\mathcal{I}_l(\mathcal{S})}$. Finally, suppose that $t \in K_{\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}}$. Then as $t \in T_{\mathcal{P}}$ it follows that $t \notin \mathcal{L}_{\mathcal{S}}$, since $\mathcal{P} \models_l \mathcal{S}$. Hence, $t \in K_{\mathcal{I}_l(\mathcal{S})}$. □

5.2.1 Refinement

The definition of refinement in the progress-sensitive framework is stronger, and so implies safety refinement $\sqsubseteq$. Accordingly, the refinement relation still corresponds to implementation containment.

Definition 5.42. Let $\mathcal{S}$ and $\mathcal{T}$ be contracts. $\mathcal{S}$ is said to be a *progress-sensitive refinement* of $\mathcal{T}$, written $\mathcal{S} \sqsubseteq_l \mathcal{T}$, iff:

RP1. $\mathcal{A}^I_{\mathcal{T}} \subseteq \mathcal{A}^I_{\mathcal{S}}$

RP2. $\mathcal{A}^O_{\mathcal{S}} \subseteq \mathcal{A}^O_{\mathcal{T}}$

RP3. $\mathcal{A}^I_{\mathcal{S}} \cap \mathcal{A}^O_{\mathcal{T}} = \emptyset$

RP4. $\mathsf{error}(\mathcal{T}) \cap \mathcal{A}^*_{\mathcal{S}} \subseteq \mathsf{error}(\mathcal{S})$

RP5. $\mathcal{R}_{\mathcal{T}} \cap \mathcal{A}^*_{\mathcal{S}} \subseteq \mathcal{R}_{\mathcal{S}} \cup \mathsf{error}(\mathcal{S})$

RP6. $\mathcal{L}_{\mathcal{T}} \cap \mathcal{A}^*_{\mathcal{S}} \subseteq \mathcal{L}_{\mathcal{S}} \cup \mathsf{error}(\mathcal{S})$. ◇

Conditions RP1-RP3 are syntactic constraints on the interfaces of the contracts to be compared and so remain unchanged from Definition 5.9. Conditions RP4 and RP5 match conditions R4 and R5 of Definition 5.9, except that references to violations are replaced by error, the latter of which is a generalisation of violations in the progress-sensitive framework. The new condition RP6 forces implementations of $\mathcal{S}$ to be live on a trace t whenever t is required to be live on $\mathcal{T}$, unless a safety or progress violation is inevitable, in which case the implementation would have suppressed an output in the assumption at an earlier stage. This requirement guarantees implementation containment, and also that refinement is a preorder (subject to compatibility).

Lemma 5.43. Refinement captures implementation containment:

$$\mathcal{S} \sqsubseteq_l \mathcal{T} \iff \{\mathcal{P} : \mathcal{P} \models_l \mathcal{S} \text{ and } \mathcal{A}^I_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{T}} = \emptyset\} \subseteq \{\mathcal{P} : \mathcal{P} \models_l \mathcal{T}\}.$$

Proof. For the only if direction, suppose $\mathcal{P} \models_l \mathcal{S}$ and $\mathcal{A}^I_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{T}} = \emptyset$. We first show that $\mathcal{P} \models \mathcal{T}$, so suppose $t \in \mathcal{R}_{\mathcal{T}} \cap T_{\mathcal{P}}$. Then, by the definition of $\sqsubseteq_l$, it follows that $t \in \mathcal{R}_{\mathcal{S}} \cup \mathsf{error}(\mathcal{S})$. If $t \in \mathsf{error}(\mathcal{S})$, then $t \notin T_{\mathcal{P}}$, since $\mathcal{P} \models_l \mathcal{S}$, which is contradictory. Therefore, $t \in \mathcal{R}_{\mathcal{S}}$, which from $\mathcal{P} \models \mathcal{S}$, implies $t \in \mathcal{G}_{\mathcal{S}} \cap \overline{F_{\mathcal{P}}}$. But as $t \notin \mathsf{error}(\mathcal{S})$, it follows that $t \notin \mathsf{error}(\mathcal{T})$. Hence $t \notin \mathsf{violations}(\mathcal{T})$, which implies $t \in \mathcal{G}_{\mathcal{T}}$. Hence $t \in \mathcal{G}_{\mathcal{T}} \cap \overline{F_{\mathcal{P}}}$ as required. Now suppose that $t \in \mathcal{L}_{\mathcal{T}} \cap T_{\mathcal{P}}$. Then from $\mathcal{S} \sqsubseteq_l \mathcal{T}$ it follows that $t \in \mathcal{L}_{\mathcal{S}} \cup \mathsf{error}(\mathcal{S})$. If $t \in \mathcal{L}_{\mathcal{S}}$, then $t \in \overline{K_{\mathcal{P}}}$,

since $\mathcal{P} \models_l \mathcal{S}$. If instead $t \in \mathsf{error}(\mathcal{S})$, then $\mathcal{P} \not\models_l \mathcal{S}$, which is contradictory. Hence, $\mathcal{P} \models_l \mathcal{T}$ as required.

For the if direction, Lemmas 5.37 and 5.41 allow us to conclude that $\mathcal{I}_l(\mathcal{S}) \sqsubseteq^l_{imp} \mathcal{I}_l(\mathcal{T})$. Suppose that $t \in \mathsf{error}(\mathcal{T}) \cap \mathcal{A}^*_{\mathcal{S}}$. Then $t \notin T_{\mathcal{I}_l(\mathcal{T})}$, hence $t \notin T_{\mathcal{I}_l(\mathcal{S})}$, meaning $t \in \mathsf{error}(\mathcal{S})$. Now suppose that $t \in \mathcal{R}_{\mathcal{T}} \cap \mathcal{A}^*_{\mathcal{S}}$. Then $t \notin F_{\mathcal{I}_l(\mathcal{T})}$, which implies $t \notin F_{\mathcal{I}_l(\mathcal{S})}$, hence $t \notin \overline{\mathcal{R}_{\mathcal{S}}} \cap \overline{\mathsf{error}(\mathcal{S})}$ i.e., $t \in \mathcal{R}_{\mathcal{S}} \cup \mathsf{error}(\mathcal{S})$. Finally, suppose that $t \in \mathcal{L}_{\mathcal{T}} \cap \mathcal{A}^*_{\mathcal{S}}$. Then $t \notin K_{\mathcal{I}_l(\mathcal{T})}$, hence $t \notin K_{\mathcal{I}_l(\mathcal{S})}$. Thus $t \notin \overline{\mathcal{L}_{\mathcal{S}}} \cap \overline{\mathsf{error}(\mathcal{S})}$, and so $t \in \mathcal{L}_{\mathcal{S}} \cup \mathsf{error}(\mathcal{S})$ as required. Thus, $\mathcal{S} \sqsubseteq_l \mathcal{T}$. □

As previously remarked in Section 5.1.1, Larsen *et al.* [LNW06] provide a sound and complete characterisation of their refinement relation in terms of conformance tests, which largely corresponds to condition RP4. Condition RP5 is not necessary in their setting because implementations are required to be input-enabled, while condition RP6 is not necessary, since they only consider safety properties, rather than safety and progress.

Refinement is naturally reflexive, and it is also transitive subject to compatibility of interfaces.

Lemma 5.44. For contracts $\mathcal{S}$, $\mathcal{T}$ and $\mathcal{U}$ such that $\mathcal{A}^I_{\mathcal{S}} \cap \mathcal{A}^O_{\mathcal{U}} = \emptyset$, if $\mathcal{S} \sqsubseteq_l \mathcal{T}$ and $\mathcal{T} \sqsubseteq_l \mathcal{U}$, then $\mathcal{S} \sqsubseteq_l \mathcal{U}$.

Proof. Follows from transitivity of $\subseteq$. □

As in the safety framework, we now show how to construct the characteristic contract for a progress-sensitive component.

Definition 5.45. The *characteristic contract* for the component $\mathcal{P}$ is a contract $\mathcal{AG}(\mathcal{P}) = \langle \mathcal{A}^I_{\mathcal{P}}, \mathcal{A}^O_{\mathcal{P}}, \mathcal{R}_{\mathcal{AG}(\mathcal{P})}, \mathcal{G}_{\mathcal{AG}(\mathcal{P})}, \mathcal{L}_{\mathcal{AG}(\mathcal{P})} \rangle$, where:

- $\mathcal{R}_{\mathcal{AG}(\mathcal{P})} = \mathcal{A}^*_{\mathcal{P}} \setminus F_{\mathcal{E}(\mathcal{P})}$
- $\mathcal{G}_{\mathcal{AG}(\mathcal{P})} = T_{\mathcal{P}} \setminus F_{\mathcal{E}(\mathcal{P})}$
- $\mathcal{L}_{\mathcal{AG}(\mathcal{P})} = T_{\mathcal{P}} \setminus K_{\mathcal{E}(\mathcal{P})}$. ◇

The assumption and guarantee are unchanged from the safety setting, while the set of liveness traces $\mathcal{L}_{\mathcal{AG}(\mathcal{P})}$ contains the non-inconsistent traces of $\mathcal{P}$ that are not quiescent.

Lemma 5.46. Let $\mathcal{P}$ be a component and let $\mathcal{S}$ be a contract. Then:

- $\mathcal{P} \models_l \mathcal{AG}(\mathcal{P})$; and
- $\mathcal{P} \models_l \mathcal{S}$ iff $\mathcal{AG}(\mathcal{P}) \sqsubseteq_l \mathcal{S}$.

Proof. For the first claim, by the properties of Lemma 5.14, it follows that $\mathcal{P} \models \mathcal{AG}(\mathcal{P})$. So suppose that $t \in \mathcal{L}_{\mathcal{AG}(\mathcal{P})} \cap T_{\mathcal{P}}$. Then $t \notin K_{\mathcal{E}(\mathcal{P})}$, hence $t \notin K_{\mathcal{P}}$ as required. Thus, $\mathcal{P} \models_l \mathcal{AG}(\mathcal{P})$.

For the second claim, the if direction follows by the previous claim and Lemma 5.43. For the only if direction, suppose that $t \in \mathsf{error}(\mathcal{S}) \cap \mathcal{A}^*_{\mathcal{P}}$. Hence $t \notin T_{\mathcal{P}} \cup F_{\mathcal{E}(\mathcal{P})}$ as $\mathcal{P} \models_l \mathcal{S}$, which implies $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})}}$. Hence, $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$, which implies $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$. Now suppose that $t \in \mathcal{R}_{\mathcal{S}} \cap \mathcal{A}^*_{\mathcal{P}}$. Then $\mathcal{P} \models_l \mathcal{S}$ implies $t \notin T_{\mathcal{P}}$ or $t \notin F_{\mathcal{E}(\mathcal{P})}$. Note that $t \notin T_{\mathcal{P}}$ implies $t \notin F_{\mathcal{E}(\mathcal{P})}$ (consider a prefix in $T_{\mathcal{P}} \cap F_{\mathcal{E}(\mathcal{P})}$). Hence $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$. Finally, suppose that $t \in \mathcal{L}_{\mathcal{S}} \cap \mathcal{A}^*_{\mathcal{P}}$. Then from $\mathcal{P} \models_l \mathcal{S}$, it follows that $t \notin T_{\mathcal{P}}$ or $t \notin K_{\mathcal{P}}$. In the case of the former, $t \notin F_{\mathcal{E}(\mathcal{P})}$ as $\mathcal{P} \models_l \mathcal{S}$, so $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$, which implies $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$. For the latter, if $t \notin \mathcal{L}_{\mathcal{AG}(\mathcal{P})}$, then $t \notin T_{\mathcal{P}}$ or $t \in T_{\mathcal{P}} \cap F_{\mathcal{E}(\mathcal{P})}$, both of which imply $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}))$, and so $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$. □

Based on these results, we define the compositional operators directly on progress-sensitive contracts. As usual, the compositions are only defined when the contracts to be acted upon are composable. The conditions for composability remain unchanged from Section 5.1, with parallel composition requiring disjointness of outputs, conjunction and disjunction insisting that action types are not mixed, and quotient needing the outputs of the subsystem to be contained within those for the whole system.

5.2.2 Parallel Composition

Again, the parallel composition of two contracts is defined as the weakest contract satisfying independent implementability. The assumption and guarantee remain largely unchanged from the safety case, except for the replacement of violations by error trace sets. On the other hand, the addition of liveness requires that a trace in the composition must make progress if at least one of the contracts requires this. This is due to the fact that parallel composition cannot suppress the output behaviour of implementing components.

Definition 5.47. Let $\mathcal{S}_{\mathcal{P}}$ and $\mathcal{S}_{\mathcal{Q}}$ be contracts composable for parallel composition. Then $\mathcal{S}_{\mathcal{P}} \mid\mid_l \mathcal{S}_{\mathcal{Q}}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}, \mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}, \mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}}, \mathcal{G}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}}, \mathcal{L}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}} \rangle$, where:

- $\mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}}$ is the largest prefix closed set such that $\mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}}(\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}})^*$ is contained within the union of:
  - $(\mathcal{R}_{\mathcal{S}_{\mathcal{P}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}) \cap (\mathcal{R}_{\mathcal{S}_{\mathcal{Q}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}})$
  - $\mathsf{error}(\mathcal{S}_{\mathcal{P}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}$
  - $\mathsf{error}(\mathcal{S}_{\mathcal{Q}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}$
- $\mathcal{G}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}} = \mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}} \cap (\overline{\mathsf{error}(\mathcal{S}_{\mathcal{P}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}}) \cap (\overline{\mathsf{error}(\mathcal{S}_{\mathcal{Q}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}})$
- $\mathcal{L}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}} = \mathcal{G}_{\mathcal{S}_{\mathcal{P}}||_l\mathcal{S}_{\mathcal{Q}}} \cap [(\mathcal{L}_{\mathcal{S}_{\mathcal{P}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}) \cup (\mathcal{L}_{\mathcal{S}_{\mathcal{Q}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}})]$. ◇

By the definition of $\mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$, we know that $\mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cap \mathcal{G}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$ as required, and any trace in $\mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$ requires that at least one of $\mathcal{S}_\mathcal{P}$ or $\mathcal{S}_\mathcal{Q}$ is live. Therefore, the parallel composition of any pair of implementations of $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ must be live on this trace. The monotonicity result and the AG rule are the same as in the non-quiescent case, but in order to show this, we first present a lemma on the decomposition of error traces arising in the parallel composition.

Lemma 5.48. $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \,||_l\, \mathcal{S}_\mathcal{Q})$ implies $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ or $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$.

Proof. Show that $t \in X_i$ implies $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ or $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$, where X_i is the i-th iteration of $\mathsf{error}(\mathcal{S}_\mathcal{P} \,||_l\, \mathcal{S}_\mathcal{Q})$ defined as a least fixed point. When $i = 0$, the result hold trivially, since $X_0 = \emptyset$. So suppose that $t \in X_{k+1}$. Then $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \,||_l\, \mathcal{S}_\mathcal{Q})$, or there exists $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})^*$ such that $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$ and $\forall o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \cdot tt'o \in X_k$. If $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \,||_l\, \mathcal{S}_\mathcal{Q})$, then there exists a prefix and input extension $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}}$. So, without loss of generality, $t' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ by the definition of $||_l$, from which it follows $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. For the latter case, without loss of generality suppose that $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$. If $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$, then it follows that there exists $o' \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cdot tt'o' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$. As $tt'o' \in X_k$, it follows that $tt'o' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Moreover, as $o' \notin \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$, it follows that $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$ as required. □

The monotonicity result for parallel composition can now be presented.

Theorem 5.49. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P}$, and $\mathcal{S}_\mathcal{Q}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for parallel composition, such that $\mathcal{A}_{\mathcal{S}'_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = \emptyset$. If $\mathcal{S}'_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$, then $\mathcal{S}'_\mathcal{P} \,||_l\, \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P} \,||_l\, \mathcal{S}_\mathcal{Q}$.

Proof. Note that the alphabet constraints are satisfied, so first show $\mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \subseteq \mathcal{R}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}'_\mathcal{P} \,||_l\, \mathcal{S}'_\mathcal{Q})$. Suppose $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}$, and all strict prefixes of t are in $\mathcal{R}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}} \cap \overline{\mathsf{error}(\mathcal{S}'_\mathcal{P} \,||_l\, \mathcal{S}'_\mathcal{Q})}$. If $t \notin \mathcal{R}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}}$, then there exists $t' \in (\mathcal{A}^O_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}})^*$ such that, without loss of generality, $tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}'_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}'_\mathcal{P})$ and $tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \notin \mathsf{error}(\mathcal{S}'_\mathcal{Q})$. As $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} = tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}}$ and $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = tt' \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{Q}}$, it follows that $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})$ since $\mathcal{S}'_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P}$, and $tt' \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \notin \mathsf{error}(\mathcal{S}_\mathcal{Q})$ since $\mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$. Hence, $tt' \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$, which implies $t \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$ as $t' \in (\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})^*$, but this is contradictory.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \,||_l\, \mathcal{S}_\mathcal{Q}) \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}$, and assume for the difficult case that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$. Then by Lemma 5.48 it follows that, without loss of generality, $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Since $t \upharpoonright \mathcal{A}_{\mathcal{S}_\mathcal{P}} = t \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}}$, it follows from $\mathcal{S}'_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ that $t \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}} \in \mathsf{error}(\mathcal{S}'_\mathcal{P})$. Now from the first part, we know $t \in \mathcal{R}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}'_\mathcal{P} \,||_l\, \mathcal{S}'_\mathcal{Q})$, so it follows that $t \in \mathsf{error}(\mathcal{S}'_\mathcal{P} \,||_l\, \mathcal{S}'_\mathcal{Q})$, since certainly $t \notin \mathcal{G}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}}$.

Finally, show $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}$ implies $t \in \mathcal{L}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}'_\mathcal{P} \,||_l\, \mathcal{S}'_\mathcal{Q})$. Suppose that $t \notin \mathsf{error}(\mathcal{S}'_\mathcal{P} \,||_l\, \mathcal{S}'_\mathcal{Q})$. Then by the first part, as $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}$, it follows that $t \in \mathcal{R}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}}$, and so $t \in \mathcal{G}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}}$. Hence $t \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}'_\mathcal{P})$ and $t \upharpoonright \mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \notin \mathsf{error}(\mathcal{S}'_\mathcal{Q})$. Now,

without loss of generality, $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, so from $\mathcal{S}'_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P}$, it follows that $t \restriction \mathcal{A}_{\mathcal{S}'_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}'_\mathcal{P}}$. Hence $t \in \mathcal{L}_{\mathcal{S}'_\mathcal{P}||_l\mathcal{S}'_\mathcal{Q}}$ as required. □

Based on this result, a sound and complete AG rule can be formulated.

Theorem 5.50. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}_\mathcal{P}$, $\mathcal{S}_\mathcal{Q}$ and $\mathcal{S}$ be contracts such that $\mathcal{A}_\mathcal{P} \cap \mathcal{A}_\mathcal{Q} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Live-Parallel}\ \frac{\mathcal{P} \models_l \mathcal{S}_\mathcal{P} \qquad \mathcal{Q} \models_l \mathcal{S}_\mathcal{Q} \qquad \mathcal{S}_\mathcal{P} \,||_l\, \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}}{\mathcal{P} \,||_l\, \mathcal{Q} \models_l \mathcal{S}}.$$

Proof. The result follows from the same reasoning in Theorem 5.19, when making use of Lemma 5.51 below, which is a generalisation of Lemma 5.20 in the safety setting. □

Lemma 5.51. $\mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q}) \sqsubseteq_l \mathcal{AG}(\mathcal{P}) \,||_l\, \mathcal{AG}(\mathcal{Q}) \sqsubseteq_l \mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q})$.

Proof. First suppose that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})}$ and $t \notin \mathsf{error}(\mathcal{AG}(\mathcal{P}) \,||_l\, \mathcal{AG}(\mathcal{Q}))$. Then $t \restriction \mathcal{A}_\mathcal{P} \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$, which implies that $t \restriction \mathcal{A}_\mathcal{P} \notin F_{\mathcal{E}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \notin F_{\mathcal{E}(\mathcal{Q})}$. Hence, $t \notin F_{\mathcal{E}(\mathcal{P}||_l\mathcal{Q})}$, from which it follows that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})}$. For the other direction, suppose $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})}$ and $t \notin \mathsf{error}(\mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q}))$. Then, $t \in \mathcal{G}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})}$, which implies $t \in T_{\mathcal{P}||_l\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}||_l\mathcal{Q})}$, which means that $t \restriction \mathcal{A}_\mathcal{P} \notin F_{\mathcal{E}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \notin F_{\mathcal{E}(\mathcal{Q})}$ i.e., $t \restriction \mathcal{A}_\mathcal{P} \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$. From this it follows that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})}$, having noticed that no output extension of t can violate this constraint.

For the error set containments, suppose that $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}) \,||_l\, \mathcal{AG}(\mathcal{Q}))$ and $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})} \cap \mathcal{R}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})}$. We demonstrate that $X_i \subseteq \mathsf{error}(\mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q}))$ for each $i \in \mathbb{N}$, where X_i is the i-th iteration of defining the least fixed point characterising $\mathsf{error}(\mathcal{AG}(\mathcal{P}) \,||_l\, \mathcal{AG}(\mathcal{Q}))$. The result holds trivially when $i = 0$, since $X_i = \emptyset$. For the inductive case, suppose $t \in X_{k+1}$. Then $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}) \,||_l\, \mathcal{AG}(\mathcal{Q}))$, or there exists $t' \in (\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}})^*$ such that $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})}$ and for each $o \in \mathcal{A}^O_{\mathcal{P}||\mathcal{Q}}$ it holds that $tt'o \in X_k$. For the former, it follows that there exists $t' \in (\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}})^*$ such that $tt' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})}}$. Consequently, without loss of generality, $tt' \restriction \mathcal{A}_\mathcal{P} \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$, which implies $t \restriction \mathcal{A}_\mathcal{P} \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$. Suppose for a contradiction that $t \in \mathcal{G}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})}$. Then $t \in T_{\mathcal{P}||_l\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}||_l\mathcal{Q})}$, which implies $t \restriction \mathcal{A}_\mathcal{P} \in T_\mathcal{P}$. But, as $t \restriction \mathcal{A}_\mathcal{P} \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$, it follows that $\mathcal{P} \not\models_l \mathcal{AG}(\mathcal{P})$, which is contradictory. Therefore, $t \notin \mathcal{G}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})}$ and so $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q}))$. For the latter case, by the induction hypothesis we have that $tt'o \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q}))$, which implies that $tt' \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q}))$, given that $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})}$ implies $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})}$. To see this last implication, from $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P})||_l\mathcal{AG}(\mathcal{Q})}$ it holds without loss of generality that $tt' \restriction \mathcal{A}_\mathcal{P} \in \mathcal{L}_{\mathcal{AG}(\mathcal{P})}$ and $tt' \restriction \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$, since $tt' \restriction \mathcal{A}_\mathcal{Q} \notin \mathsf{error}(\mathcal{AG}(\mathcal{Q}))$. Hence, $tt' \restriction \mathcal{A}_\mathcal{P} \in T_\mathcal{P} \setminus K_{\mathcal{E}(\mathcal{P})}$ and $tt' \restriction \mathcal{A}_\mathcal{Q} \in T_\mathcal{Q} \cap \overline{F_{\mathcal{E}(\mathcal{Q})}}$. Thus, $tt' \in T_{\mathcal{P}||_l\mathcal{Q}} \setminus K_{\mathcal{E}(\mathcal{P}||_l\mathcal{Q})}$, implying $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P}||_l\mathcal{Q})}$. Consequently, $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \,||_l\, \mathcal{Q}))$ as required.

For the other direction of the containment, suppose that both $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_l \mathcal{Q}))$ and $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}\mid\mid_l\mathcal{Q})} \cap \mathcal{R}_{\mathcal{AG}(\mathcal{P})\mid\mid_l\mathcal{AG}(\mathcal{Q})}$. Using a similar X_i argument it follows $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P} \mid\mid_l \mathcal{Q}))$, or there exists $t' \in (\mathcal{A}^I_{\mathcal{P}\mid\mid\mathcal{Q}})^*$ such that $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P}\mid\mid_l\mathcal{Q})}$ and for each $o \in \mathcal{A}^O_{\mathcal{P}\mid\mid\mathcal{Q}}$, it holds that $tt'o \in \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_l \mathcal{AG}(\mathcal{Q}))$. For the former, suppose that there exists $t' \in (\mathcal{A}^I_{\mathcal{P}\mid\mid\mathcal{Q}})^*$ such that $tt' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}\mid\mid_l\mathcal{Q})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P}\mid\mid_l\mathcal{Q})}}$. Then $tt' \notin T_{\mathcal{P}\mid\mid_l\mathcal{Q}} \cup F_{\mathcal{E}(\mathcal{P}\mid\mid_l\mathcal{Q})}$, which implies without loss of generality that $tt' \upharpoonright \mathcal{A}_\mathcal{P} \notin T_\mathcal{P} \cup F_{\mathcal{E}(\mathcal{P})}$. Hence, $tt' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})}}$, which implies $tt' \upharpoonright \mathcal{A}_\mathcal{P} \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$. Therefore, $t \upharpoonright \mathcal{A}_\mathcal{P} \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$, which implies $t \notin \mathcal{G}_{\mathcal{AG}(\mathcal{P})\mid\mid_l\mathcal{AG}(\mathcal{Q})}$. Consequently, $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_l \mathcal{AG}(\mathcal{Q}))$ as we are assuming that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})\mid\mid_l\mathcal{AG}(\mathcal{Q})}$. For the latter, from $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P}\mid\mid_l\mathcal{Q})}$, it follows that $tt' \in T_{\mathcal{P}\mid\mid_l\mathcal{Q}} \setminus K_{\mathcal{E}(\mathcal{P}\mid\mid_l\mathcal{Q})}$. Consequently, without loss of generality, $tt' \upharpoonright \mathcal{A}_\mathcal{P} \in T_\mathcal{P} \setminus K_{\mathcal{E}(\mathcal{P})}$ and $tt' \upharpoonright \mathcal{A}_\mathcal{Q} \in T_\mathcal{Q} \setminus F_{\mathcal{E}(\mathcal{Q})}$. This means that $tt' \upharpoonright \mathcal{A}_\mathcal{P} \in \mathcal{L}_{\mathcal{AG}(\mathcal{P})}$ and $tt' \upharpoonright \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$. Thus $tt' \in \mathcal{L}_{\mathcal{AG}(\mathcal{P})\mid\mid_l\mathcal{AG}(\mathcal{Q})} \cup \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_l \mathcal{AG}(\mathcal{Q}))$. Either way, we derive $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_l \mathcal{AG}(\mathcal{Q}))$.

The reasoning for the liveness set containments can be extracted from the error set containments mentioned previously. □

5.2.3 Conjunction

We now give an updated definition of conjunction that works in the progress-sensitive setting. The assumption remains unchanged from the safety framework, and the guarantee is modified by making reference to the error traces, rather than the violations traces, of the contracts to be composed. Progress, on the other hand, must be made when at least one of the contracts can make progress, and the other contract has not violated its guarantee.

Definition 5.52. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ be contracts composable for conjunction. Then $\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}, \mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}}, \mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}}, \mathcal{L}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}} \rangle$, where:

- $\mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}} = (\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathcal{R}_{\mathcal{S}_\mathcal{Q}}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}$
- $\mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}}$ is the intersection of the following sets:
  - $\mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}}$
  - $\overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \cup (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \upharpoonright \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}})$
  - $\overline{\mathsf{error}(\mathcal{S}_\mathcal{Q})} \cup (\overline{\mathsf{error}(\mathcal{S}_\mathcal{Q})} \upharpoonright \mathcal{A}^I_{\mathcal{S}_\mathcal{P}})$
- $\mathcal{L}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}} = \mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge_l\mathcal{S}_\mathcal{Q}} \cap (\mathcal{L}_{\mathcal{S}_\mathcal{P}} \cup \mathcal{L}_{\mathcal{S}_\mathcal{Q}})$. ◇

Unlike conjunction on safety contracts, the conjunctive operator in the progress-sensitive setting may not be implementable, even if the two contracts to be conjoined have implementations. This is a consequence of the conflicting nature between safety and progress. It can be shown that this conjunctive operator on contracts is the meet operator for the progress-sensitive refinement preorder, corresponds to intersection of implementations, and is monotonic under refinement.

Theorem 5.53. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for conjunction. Then:

- $\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}_\mathcal{R} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{R} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}_\mathcal{R} \sqsubseteq_l \mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}'_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}'_\mathcal{P} \wedge_l \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}$.

Proof. First show that $\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P}$. Suppose $t \in \mathsf{error}(\mathcal{S}_\mathcal{P}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$. Then there is a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$ and $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Therefore, $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}}$, implying $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q})$. If $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$, then $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$ as required. Finally, suppose $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$. As $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$, it follows that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$. Moreover, if $t \notin \mathsf{error}(\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q})$, then $t \in \mathcal{G}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$. So from $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, it is easy to see that $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$ as required. By similar reasoning $\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$.

For the second claim, we show $\mathsf{error}(\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{R}} \subseteq \mathsf{error}(\mathcal{S}_\mathcal{R})$ by demonstrating that $t \in X_i \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{R}}$ implies $t \in \mathsf{error}(\mathcal{S}_\mathcal{R})$ by induction on i, where X_i is the i-th iteration of defining $\mathsf{error}(\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q})$ as a least fixed point. When $i = 0$ the result holds trivially as $X_i = \emptyset$. Now suppose $i = k$ for $k > 0$. If $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q})$, then there is a prefix t' of t and input extension $t'' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})^*$ such that $t't'' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}}$. So without loss of generality, $t't'' \notin \overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \cup (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}})$. This means that there is a prefix of $t't''$ contained in $\mathsf{error}(\mathcal{S}_\mathcal{P})$, which must also be in $\mathsf{error}(\mathcal{S}_\mathcal{R})$ since $\mathcal{S}_\mathcal{R} \sqsubseteq_l \mathcal{S}_\mathcal{P}$. If instead there exists $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})^*$ such that $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$ and $\forall o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}} \cdot tt'o \in X_{i-1}$, then $\forall o' \in \mathcal{A}^O_{\mathcal{S}_\mathcal{R}}$ it follows that $tt'o' \in \mathsf{error}(\mathcal{S}_\mathcal{R})$ by the induction hypothesis. Moreover, from $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$, it follows that without loss of generality, $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$. So from $\mathcal{S}_\mathcal{R} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ we derive $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{R}} \cup \mathsf{error}(\mathcal{S}_\mathcal{R})$. But $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{R}}$ also implies $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{R})$, hence $t \in \mathsf{error}(\mathcal{S}_\mathcal{R})$ as required. Now suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{R}}$. Then without loss of generality, $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{R}}$, so from $\mathcal{S}_\mathcal{R} \sqsubseteq_l \mathcal{S}_\mathcal{P}$, we derive $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cup \mathsf{error}(\mathcal{S}_\mathcal{R})$. Finally, suppose $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{R}}$. If $t \notin \mathsf{error}(\mathcal{S}_\mathcal{R})$, then we have $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cap \mathcal{G}_{\mathcal{S}_\mathcal{R}}$, since $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$ implies $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}}$, which implies $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}}$. Without loss of generality, $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, so from $\mathcal{S}_\mathcal{R} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ it follows that $t \in \mathcal{L}_{\mathcal{S}_\mathcal{R}}$ as required.

For the third claim, by the first claim we have $\mathcal{S}'_\mathcal{P} \wedge_l \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P} \wedge_l \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}'_\mathcal{Q}$. Now by transitivity, we see that $\mathcal{S}'_\mathcal{P} \wedge_l \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P} \wedge_l \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$ providing $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^I_{\mathcal{S}'_\mathcal{Q}} = \emptyset$ and $\mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^I_{\mathcal{S}'_\mathcal{P}} = \emptyset$, so by the second claim, it follows that $\mathcal{S}'_\mathcal{P} \wedge_l \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}$ as required. If either of the compatibility conditions are not satisfied, we can obtain new contracts $\mathcal{S}''_\mathcal{P}$ for $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}''_\mathcal{Q}$ for $\mathcal{S}_\mathcal{Q}$ that have output set $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$ and contain all traces from the respective contracts, except for those with an output in $(\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \setminus \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}) \cup (\mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \setminus \mathcal{A}^O_{\mathcal{S}_\mathcal{P}})$ that has been removed from the interface. It is straightforward to show that $\mathcal{S}''_\mathcal{P} \wedge_l \mathcal{S}''_\mathcal{Q} = \mathcal{S}_\mathcal{P} \wedge_l \mathcal{S}_\mathcal{Q}$. □

Given these strong algebraic properties, we can formulate a sound and complete AG rule for conjunction, based on the one presented in Section 5.1.3.

Theorem 5.54. Let $\mathcal{P}$ be a component, and let $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}$ be contracts such that $\mathcal{A}^I_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Live-Conjunction}\ \frac{\mathcal{P} \models_l \mathcal{S}_1 \qquad \mathcal{P} \models_l \mathcal{S}_2 \qquad \mathcal{S}_1 \wedge_l \mathcal{S}_2 \sqsubseteq_l \mathcal{S}}{\mathcal{P} \models_l \mathcal{S}}.$$

Proof. Follows from Theorem 5.23, with minimal change. □

Example 5.55. Reverting to Example 5.24, we stipulate that, for the property Spec2, if the observed behaviour over login and logout is always a prefix of $\langle \mathsf{login}, \mathsf{logout} \rangle^*$, then process and logout should alternate, and progress must be made whenever a job has been processed and before a logout request is seen. This can be achieved by making the right-hand state live in $\mathcal{G}_{\mathsf{Spec2}}$ of Figure 5.5. This liveness requirement manifests itself as a liveness requirement after process and before logout in the conjunction, indicated by the square node in Figure 5.6. ◇

5.2.4 Disjunction

In this section, we generalise the disjunctive rule from the safety setting for the progress-sensitive framework. The assumption and guarantee are obtained by replacing references to violations with references to error in the definitions from the safety case. The progress condition, on the other hand, must ensure that progress is made only when each of the contracts to be composed can make progress, unless an error has been encountered, or an output has been seen not belonging to that contract.

Definition 5.56. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ be contracts composable for disjunction. Then $\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}, \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}, \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}, \mathcal{L}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} \rangle$ defined by:

- $\mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$ is the intersection of the following sets:
  - $\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}})$
  - $\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{P}})$
- $\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} = \overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \cup \overline{\mathsf{error}(\mathcal{S}_\mathcal{Q})}$
- $\mathcal{L}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$ is the intersection of the following sets:
  - $\mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} \cap \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$
  - $\mathcal{L}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}})$
  - $\mathcal{L}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{P}})$. ◇

This definition of disjunction satisfies the same algebraic properties as the disjunctive operator in the safety framework, but with respect to the progress-sensitive refinement preorder. Consequently, $\vee_l$ is the join operator for $\sqsubseteq_l$.

Theorem 5.57. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for disjunction. Then:

- $\mathcal{S}_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{R}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{R}$ implies $\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{R}$
- $\mathcal{S}'_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}'_\mathcal{P} \vee_l \mathcal{S}'_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}$.

Proof. For the first claim of $\mathcal{S}_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}$, we first show that $\mathsf{error}(\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}} \subseteq \mathsf{error}(\mathcal{S}_\mathcal{P})$. So let X_i be the i-th iteration of $\mathsf{error}(\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q})$ being defined as a least fixed point. Then by induction on i, we show that $X_i \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}} \subseteq \mathsf{error}(\mathcal{S}_\mathcal{P})$. Suppose that $t \in X_{k+1} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$. If $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q})$, then there is a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}}$. Hence $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ and so $t \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ as required. Otherwise, there is a trace $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}})^*$ such that $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$ and for all $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$ it holds that $tt'o \in X_k$. Consequently, as $t' \in (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}})^*$, it follows that $tt' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$, and so $tt' \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$. As a result, $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ since $tt'o' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ for each $o' \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P}}$ by the induction hypothesis. From this we derive $t \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Now suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$. Then $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})$ by definition. Similarly, if $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$, then $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})$ as required. Showing $\mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}$ is similar.

For the second claim, suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. If $t \equiv \epsilon$, then $\epsilon \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$ trivially, while if $t \equiv t'o$ for $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$, then $t'o \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$ by the induction hypothesis and output extendability of assumptions or extendability of violations/error. Instead, if $t \equiv t'i$ for $i \in \mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$, then by the induction hypothesis in the difficult case we have $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \overline{\mathsf{error}(\mathcal{S}_\mathcal{P})}$ and $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cap \overline{\mathsf{error}(\mathcal{S}_\mathcal{Q})}$. As $i \in \mathcal{A}^I_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$, it follows from $\mathcal{S}_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{R}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{R}$ that $t'i \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{R}_{\mathcal{S}_\mathcal{Q}}$. Hence, $t'i \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{R}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. Then there exists a smallest prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cap \mathsf{error}(\mathcal{S}_\mathcal{R}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. Suppose all strict prefixes of t' are not in $\mathsf{error}(\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q})$. Then by the previous part, it follows that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$. If $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$, then from $\mathcal{S}_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{R}$ it follows that $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$, and if $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$, then from $\mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{R}$ it follows that $t' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Hence $t' \notin \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$ (noting $\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}^*_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$), which implies $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q})$. By extension closure of error, we have $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q})$.

For the progress condition, suppose $t \in \mathcal{L}_{\mathcal{S}_\mathcal{R}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. Assuming $t \notin \mathsf{error}(\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q})$, we can infer that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}} \cap \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$. Suppose for a contradiction that $t \notin \mathcal{L}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$. Then since $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$, it follows that $t \in \mathcal{A}^*_{\mathcal{S}_\mathcal{P}}$ and $t \notin \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, or $t \in \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$ and $t \notin \mathcal{L}_{\mathcal{S}_\mathcal{Q}}$. However, both of these contradict $\mathcal{S}_\mathcal{P} \sqsubseteq_l \mathcal{S}_\mathcal{R}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{P}$. Hence $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P} \vee_l \mathcal{S}_\mathcal{Q}}$ as required.

The third claim follows by the same reasoning as in Theorem 5.26. □

Based on the algebraic properties of disjunction, we can formulate a sound and complete assume-guarantee rule. This demonstrates that a disjunctive contract contains the union of the

implementations of the contracts to be composed, but, as in the safety framework, there may be additional implementations that are not implementations of either of the contracts to be composed.

Theorem 5.58. Let $\mathcal{P}$ be a component, and let $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}$ be contracts such that $\mathcal{S}_1$ and $\mathcal{S}_2$ are composable for disjunction, and $\mathcal{A}^I_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Live-Disjunction}\ \frac{\mathcal{P} \models_l \mathcal{S}_1 \quad \text{or} \quad \mathcal{P} \models_l \mathcal{S}_2 \qquad \mathcal{S}_1 \vee_l \mathcal{S}_2 \sqsubseteq_l \mathcal{S}}{\mathcal{P} \models_l \mathcal{S}}.$$

Proof. The reasoning of Theorem 5.27 applies. □

5.2.5 Quotient

The definition of quotient in the progress-sensitive setting corresponds to the adjoint of the progress-sensitive parallel composition operator, with respect to the $\sqsubseteq_l$ preorder. Consequently, its definition is based on the definitions of $||_l$ and $/$ (Definitions 5.47 and 5.28).

Definition 5.59. Let $\mathcal{S}_{\mathcal{P}}$ and $\mathcal{S}_{\mathcal{W}}$ be contracts. Then the quotient $\mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}, \mathcal{A}^O_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}, \mathcal{R}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}}, \mathcal{G}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}}, \mathcal{L}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}} \rangle$, defined only when $\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}} \subseteq \mathcal{A}^O_{\mathcal{S}_{\mathcal{W}}}$, where:

- $\mathcal{R}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}} = [\mathcal{R}_{\mathcal{S}_{\mathcal{W}}} \cap (\overline{\mathsf{error}(\mathcal{S}_{\mathcal{P}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}})] \upharpoonright \mathcal{A}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}$
- $\mathcal{G}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}}$ is the largest subset of $\mathcal{R}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}}$ disjoint from $[\mathcal{R}_{\mathcal{S}_{\mathcal{W}}} \cap (\overline{\mathsf{error}(\mathcal{S}_{\mathcal{P}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}}) \cap (\mathsf{error}(\mathcal{S}_{\mathcal{W}}) \cup (\overline{\mathcal{R}_{\mathcal{S}_{\mathcal{P}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}}))] \upharpoonright \mathcal{A}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}$
- $\mathcal{L}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}} = \mathcal{G}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}} \cap [\mathcal{L}_{\mathcal{S}_{\mathcal{W}}} \cap (\overline{\mathsf{error}(\mathcal{S}_{\mathcal{P}}) \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}}) \cap (\overline{\mathcal{L}_{\mathcal{S}_{\mathcal{P}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{W}}}})] \upharpoonright \mathcal{A}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}}$. ◇

$\mathcal{L}_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}}$ is defined such that the parallel composition $\mathcal{S}_{\mathcal{P}} \ ||_l \ (\mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}})$ is live whenever $\mathcal{S}_{\mathcal{W}}$ is live. Moreover, to ensure that $\mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}}$ is the least refined solution to $\mathcal{S}_{\mathcal{P}} \ ||_l \ X \sqsubseteq_l \mathcal{S}_{\mathcal{W}}$, $\mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}}$ is only live when $\mathcal{S}_{\mathcal{P}} \ ||_l \ (\mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}})$ needs to be live and $\mathcal{S}_{\mathcal{P}}$ is not live. The next theorem shows that our definition satisfies the characteristic properties of quotient.

Theorem 5.60. Let $\mathcal{S}_{\mathcal{P}}$ and $\mathcal{S}_{\mathcal{W}}$ be contracts. Then there exists a contract $\mathcal{S}_{\mathcal{Q}}$ such that $\mathcal{S}_{\mathcal{P}} \ ||_l \ \mathcal{S}_{\mathcal{Q}} \sqsubseteq_l \mathcal{S}_{\mathcal{W}}$ iff the following properties hold:

- The quotient $\mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}}$ is defined
- $\mathcal{S}_{\mathcal{P}} \ ||_l \ (\mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}}) \sqsubseteq_l \mathcal{S}_{\mathcal{W}}$
- $\mathcal{A}^I_{\mathcal{S}_{\mathcal{Q}}} = \mathcal{A}^I_{\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}}}$ implies $\mathcal{S}_{\mathcal{Q}} \sqsubseteq_l \mathcal{S}_{\mathcal{W}} \ /_l \ \mathcal{S}_{\mathcal{P}}$.

Proof. The first claim follows by the reasoning in Theorem 5.29.

For the second claim, suppose $t \in \mathcal{R}_{\mathcal{S}_{\mathcal{W}}} \cap \mathcal{A}^*_{\mathcal{S}_{\mathcal{P}}||_l(\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}})}$. If $t \notin \mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_l(\mathcal{S}_{\mathcal{W}}/_l\mathcal{S}_{\mathcal{P}})}$, then there exists a prefix t' of t and $t'' \in (\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}||(\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}})})^*$ such that $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_{\mathcal{P}}} \notin \mathcal{R}_{\mathcal{S}_{\mathcal{P}}}$ or $t't'' \upharpoonright \mathcal{A}_{\mathcal{S}_{\mathcal{W}}/\mathcal{S}_{\mathcal{P}}} \notin$

$\mathcal{R}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$, and $t't'' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$ and $t't'' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$. It follows that $t't'' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$, so $t't'' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$, which means $t't'' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$. Therefore, $t't'' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathcal{G}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$, which implies $t't'' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$. But this contradicts $t't'' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$. Hence $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{W}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})}$. Then, there exists a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}} \cap \mathsf{error}(\mathcal{S}_\mathcal{W})$. By the previous part, it follows that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$. Now suppose for a contradiction that $t' \in \mathcal{G}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$. Then $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$ and $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$. But it follows that $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathsf{violations}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$, since $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}}$. This contradicts $t' \in \mathcal{G}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$. Hence $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; (\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}))$ and so $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; (\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}))$.

Finally, suppose that $t \in \mathcal{L}_{\mathcal{S}_\mathcal{W}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$, and $t \notin \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; (\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}))$. Then by the previous part, $t \notin \mathsf{error}(\mathcal{S}_\mathcal{W})$, so $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})} \cap \mathcal{G}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$. Hence $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \overline{\mathsf{error}(\mathcal{S}_\mathcal{P})}$ and $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}} \cap \mathcal{G}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$. If $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, then $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$ as required, since $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$ implies $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathcal{G}_{\mathcal{S}_\mathcal{P}}$. If instead $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, then $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$, which implies $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$ as required.

For the third claim, suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$. Then there exists $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{W}}$ such that $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} = t$ with $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ and $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$. From $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ we derive $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q})$, given that $\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{W}$. If $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$, then it follows that $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q})$. If instead $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q})$, then it follows that $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$ by Lemma 5.48. Note that $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = t$.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}) \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$. Then there exists a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}} \cap \mathsf{error}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$. We show that $X_i \cap \mathcal{R}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}} \subseteq \mathsf{error}(\mathcal{S}_\mathcal{Q})$ by induction on i, where X_i is the i-th iteration of defining $\mathsf{error}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$ as a least fixed point. The case of $i = 0$ is trivial, since $X_0 = \emptyset$. For the difficult case of $t' \in X_{k+1}$, either: (i) $t' \in \mathsf{violations}(\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P})$; or (ii) there exists $t'' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}})^*$ such that $t't'' \in \mathcal{L}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$ and $\forall o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cdot t't''o \in X_k$. For (i), there is a prefix and input extension t'' of t' such that there exists $t_w \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ with $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} = t''$, $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$, and either $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{W})$ or $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$. If $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{W})$, then $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q})$, since $\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{W}$. By Lemma 5.48, it follows that $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Alternatively, if $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$, then if $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \notin \mathsf{error}(\mathcal{S}_\mathcal{Q})$ it follows that $t_w \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$. Since $\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{W}$, it must hold that $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q})$, which again by Lemma 5.48 implies $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Note that $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = t''$, so $t \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. For (ii), by the induction hypothesis we know that $\forall o' \in \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \cdot t't''o' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. To show that $t't'' \in \mathcal{L}_{\mathcal{S}_\mathcal{Q}}$, note from $t't'' \in \mathcal{L}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$ that there exists $t_w \in \mathcal{L}_{\mathcal{S}_\mathcal{W}}$ with $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} = t't''$ such that $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{L}_{\mathcal{S}_\mathcal{P}}$ and $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$. Since $\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S}_\mathcal{W}$, it follows that $t_w \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$ or $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q})$. For the former, it follows that $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathcal{L}_{\mathcal{S}_\mathcal{Q}}$, while for the latter $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$ (Lemma 5.48). Either way, since $t_w \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = t't''$, it follows that $t't'' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$, which in turn yields $t' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$.

Finally, suppose that $t \in \mathcal{L}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$. Then there exists $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{W}}$ with $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} = t$ such that $t' \in \mathcal{L}_{\mathcal{S}_\mathcal{W}}$, $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{L}_{\mathcal{S}_\mathcal{P}}$ and $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$. From $t' \in \mathcal{L}_{\mathcal{S}_\mathcal{W}}$ we derive $t' \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q})$. If $t' \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l\mathcal{S}_\mathcal{Q}}$, then certainly $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathcal{L}_{\mathcal{S}_\mathcal{Q}}$. If instead $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; \mathcal{S}_\mathcal{Q})$, then by Lemma 5.48 $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. It is easy to see that $t' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} = t$. □

The intuition behind the definition remains largely unchanged from the text proceeding Theorem 5.29, having updated references to violations with error. In the case of the liveness set, if $t \in \mathcal{L}_{\mathcal{S}_\mathcal{W}} \setminus \mathsf{error}(\mathcal{S}_\mathcal{W})$ and $t \notin \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; (\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}))$, then we require $t \in \mathcal{L}_{\mathcal{S}_\mathcal{P}||_l(\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P})}$. If $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, then it need not hold that $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$. If instead $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \notin \mathcal{L}_{\mathcal{S}_\mathcal{P}}$, then it must hold that $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$. Moreover, note that if $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{error}(\mathcal{S}_\mathcal{P})$, then $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_l\; (\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}))$, so we do not require $t \restriction \mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \in \mathcal{L}_{\mathcal{S}_\mathcal{W}/_l\mathcal{S}_\mathcal{P}}$.

Parameterisation of the input set for progress-sensitive quotient is applicable just as in the safety setting. Based on these properties of quotient, we can formulate a sound and complete AG rule, closely mirroring the rule of Theorem 5.30.

Theorem 5.61. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts such that $\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}$ is defined, let $\mathcal{P}$ range over components having the same interface as $\mathcal{S}_\mathcal{P}$, and let $\mathcal{Q}$ be a component having the same interface as $\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}$ (where the quotient is parameterised on the set $\mathcal{A}^I_\mathcal{Q}$). Then the following AG rule is both sound and complete:

$$\textsc{Live-Quotient}\frac{\forall \mathcal{P} \cdot \mathcal{P} \models_l \mathcal{S}_\mathcal{P} \text{ implies } \mathcal{P} \;||_l\; \mathcal{Q} \models_l \mathcal{S}_\mathcal{W}}{\mathcal{Q} \models_l \mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}}.$$

Proof. Follows by straightforward modification to Theorem 5.30, having updated concepts to the progress-sensitive equivalents. □

As in Theorem 5.30, we insist that the components $\mathcal{P}$ and $\mathcal{Q}$ must have the same interfaces as their respective contracts, since parallel composition is only monotonic when restrictions are placed on the interfaces of the contracts to be composed (cf Theorem 5.49). Furthermore, the rule can be reformulated so as to avoid the universal quantification by considering the least refined implementation $\mathcal{I}_l(\mathcal{S}_\mathcal{P})$ of $\mathcal{S}_\mathcal{P}$.

Corollary 5.62. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts such that $\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}$ is defined, and let $\mathcal{Q}$ be a component having the same interface as $\mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}$ (where the quotient is parameterised on the set $\mathcal{A}^I_\mathcal{Q}$). Then the following AG rule is both sound and complete:

$$\textsc{Live-Quotient-Revised}\frac{\mathcal{I}_l(\mathcal{S}_\mathcal{P}) \;||_l\; \mathcal{Q} \models_l \mathcal{S}_\mathcal{W}}{\mathcal{Q} \models_l \mathcal{S}_\mathcal{W} \;/_l\; \mathcal{S}_\mathcal{P}}.$$

Proof. Unchanged from Corollary 5.31, having updated references. □

Example 5.63. We now reconsider the Client contract in the progress-sensitive setting, first introduced in Example 5.32. Note that the bottom right node of $\mathcal{G}_{\mathsf{Client}}$ is required to be live in Figure 5.8, since Spec1 $\wedge$ Spec2 requires liveness after the trace $\langle \mathsf{login}, \mathsf{job}, \mathsf{process} \rangle$, whereas Server does not need to be live after this trace (projected on to its own interface). As all output extensions of the trace $\langle \mathsf{login}, \mathsf{job} \rangle$ in Client are contained within error(Client), it follows that $\langle \mathsf{login}, \mathsf{job} \rangle \in \mathsf{error}(\mathsf{Client})$. Consequently, every implementation of the Client contract is unable to issue a job after a successful login, because, if it were to do so, there would be no guarantee that the Server will acknowledge the processing, meaning that a liveness violation can arise. $\diamond$

5.2.6 Decomposing Parallel Composition

As in the case of dealing with safety properties, we present a modification to the progress-sensitive AG rule for parallel composition, by making use of quotient on contracts.

Corollary 5.64. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}_\mathcal{P}$, $\mathcal{S}_\mathcal{Q}$ and $\mathcal{S}$ be contracts such that $\mathcal{A}_\mathcal{P} \cap \mathcal{A}_\mathcal{Q} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P} || \mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. When the quotient is parameterised on $\mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$, the following rule is both sound and complete:

$$\textsc{Live-Parallel-Decompose}\frac{\mathcal{P} \models_l \mathcal{S}_\mathcal{P} \qquad \mathcal{Q} \models_l \mathcal{S}_\mathcal{Q} \qquad \mathcal{S}_\mathcal{Q} \sqsubseteq_l \mathcal{S} \mathbin{/_l} \mathcal{S}_\mathcal{P}}{\mathcal{P} \mathbin{||_l} \mathcal{Q} \models_l \mathcal{S}}.$$

Proof. Follows directly from Theorems 5.50 and 5.60. $\square$

This rule is useful for scenarios when the contract $\mathcal{S}$ is supplied along with a sub-contract $\mathcal{S}_\mathcal{P}$ (or for when a subcontract $\mathcal{S}_\mathcal{P}$ can easily be inferred). In such circumstances, the missing contract $\mathcal{S}_\mathcal{Q}$ can be taken as any refinement of $\mathcal{S} \mathbin{/_l} \mathcal{S}_\mathcal{P}$.

5.3 Case Study

To demonstrate our assume-guarantee framework at work, and to relate it to previously proposed frameworks, we consider a link layer protocol case study drawn from distributed systems, which is a variant of the running example used in [LNW06]. A Client (see Figure 5.9) can communicate with a Server (Figure 5.10) by sending data, and can observe whether the transmission was ok or whether it failed. The Server, on the other hand, is an intermediary between the Client and a Database server. It receives data from the Client via the send interaction, and then transmits it to the Database engine via some communication medium, after which it waits for positive or negative confirmation that the data has been written into the database, in the form of ack and nack signals, respectively. In the case that the transmission is acknowledged, the Server indicates to the Client that all is ok. Otherwise, if nack is received from the Database, the Server attempts to retransmit, and if nack is received for a second time in succession, the Server will signify to the Client that a

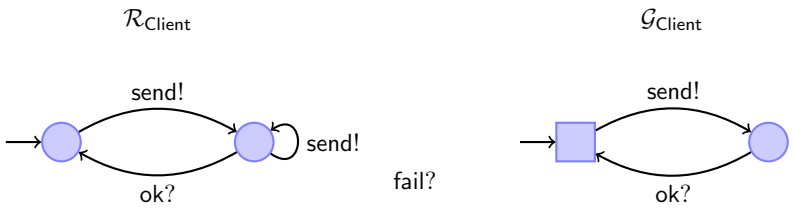

Figure 5.9: Assumption and guarantee of Client (the adrift fail? action indicates that this action appears in the interface of the Client contract)

failure has occurred. The models of the Client and Server are taken from [LNW06] (where they are referred to as *Client* and *TryTwice* respectively), in order to highlight the differentiating features of our work.

Through this case study, we aim to illustrate how to generate automatically the most general contract for the communication medium by applying the AG rules, instead of constructing such a contract working from informal requirements. We will first compute a contract for the combined behaviour of the Client and Server, which we then use to derive the contract for the communication medium by means of the quotient operation.

We begin by considering the parallel composition of the Client with Server, which is shown in Figure 5.11. To understand intuitively how Figure 5.11 is derived, note that fail appears in the static interface of Client, yet Client assumes that fail will never be issued by the environment. It follows that Client || Server can never guarantee that there is a safe behaviour containing fail. Therefore, to prevent such a behaviour arising, the environment must never issue the preceding nack, which will in turn prevent an implementation of Server from issuing fail to an implementation of the Client.

When contrasting Figure 5.11 with the parallel composition of Client and Server in [LNW06] (where our Server corresponds to *TryTwice*), after accounting for the difference in parallel composition (whereby we do not automatically hide the actions that are shared between components, i.e, send, ok, and fail), one observes that our guarantee can be expressed in a simpler manner, given that it need not be input-enabled.

We now wish to construct a contract representing the behaviour of the communication medium that transmits information between the Server and Database. As a first attempt, working from the requirements, we formulate a contract that merely represents an abstract protocol for interaction between the Server and Database, reproduced as LinkLayer1 in Figure 5.12. The protocol awaits a transmission request (transmit), after which it attempts to write the data to the database. Successful writing of the data results in a positive acknowledgment, while a nack occurs if the write request does not complete successfully, or if the write request cannot be performed for some reason. Unfortunately, the parallel composition of LinkLayer1 with Client || Server is a contract for which nothing can be assumed, meaning that no safety or progress proper-

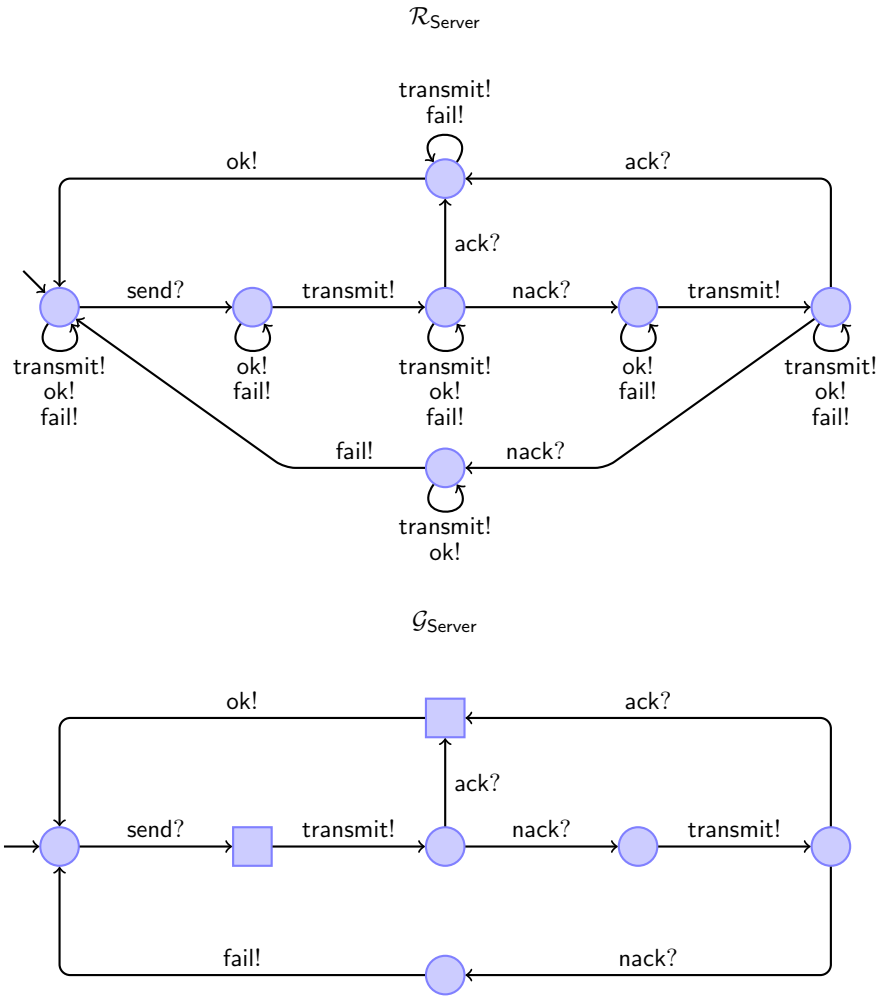

Figure 5.10: Assumption and guarantee of Server

ties can be inferred. To see why, note that the assumption of the composition must be empty because $\langle \mathsf{send}, \mathsf{transmit}, \mathsf{nack}, \mathsf{transmit}, \mathsf{nack} \rangle$ is a trace over outputs whose projections onto Server || Client and LinkLayer1, respectively, are not contained in both assumptions, while they are also not in the respective error sets (cf Definition 5.15 for parallel).

As a second attempt, we therefore use our theory to automate the derivation of the weakest restrictions to the communication medium that allows all three of the Client, Server and Database to communicate. This can be formulated as the quotient ErrorFree/(Client || Server), where ErrorFree is the component having a single chaotic state labelled by all actions, which should be treated as outputs (Figure 5.13). The only significant constraint imposed by ErrorFree is that no communication mismatches will occur. Note that the state is not required to be live, and hence the assumption is the same as the guarantee. The resulting contract, referred to as LinkLayer2, is depicted in Figure 5.14 when the set of input actions is taken to be $\{\mathsf{send}, \mathsf{transmit}, \mathsf{ok}\}$, whereas Figure 5.15 is the corresponding contract synthesised by the quotient operation when the set of inputs is taken to be $\{\mathsf{transmit}\}$.

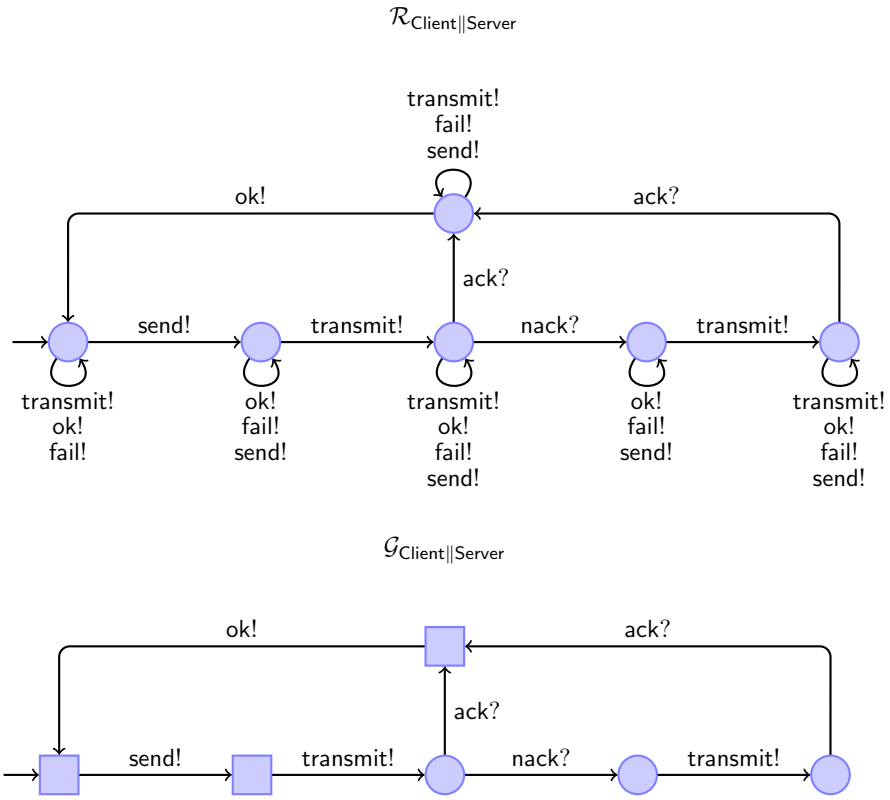

Figure 5.11: Assumption and guarantee of Client || Server

LinkLayer2 (parameterised on {transmit}) is thus a contract that will allow Server and Client to interact with one another, but it may not respect the protocol of LinkLayer1, meaning that it may not meaningfully interact with the Database. Therefore, we define LinkLayer1 ∧ LinkLayer2 as the contract for implementations that should communicate with Database (shown in Figure 5.16). Any implementation of this contract must never nack two transmissions in succession.

We now consider the impact of considering liveness, in addition to safety. Let ErrorFreeLive be the ErrorFree contract, but with the requirement that the sole state must be live (also shown in Figure 5.13). Then ErrorFreeLive/(Client || Server) is the contract in Figure 5.14 (when the quotient is parameterised on {send, transmit, ok}) and Figure 5.15 (when parameterised on {transmit}), but with states containing • treated as though they are live (i.e., they should be squares). Similarly, LinkLayer1 ∧ LinkLayer2 is as depicted in Figure 5.16, but with the • filled nodes converted to squares. In the liveness setting, an implementation of the conjunction must always ack after write and must never nack, because, if the latter were to happen, we would be in a live state from which the implementation cannot safely issue any output, which conflicts with the progress requirements imposed by ErrorFreeLive.

To summarise, this case study demonstrates how our framework adds significant flexibility over previous frameworks, such as the one in [LNW06]. Specifically, we provide a simpler formalism that does not require input-enabledness of guarantees, while supporting compositional

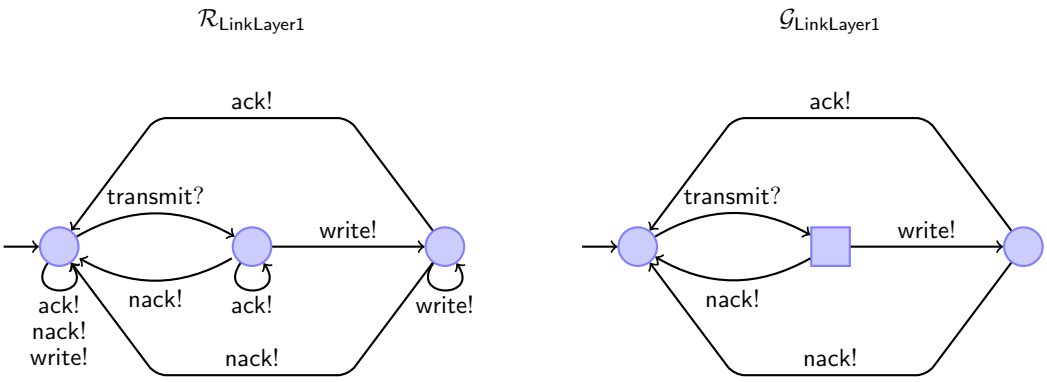

Figure 5.12: Assumption and guarantee of LinkLayer1

Figure 5.13: Assumption and guarantee of ErrorFree and ErrorFreeLive

reasoning not only for safety, but also liveness properties. A rich collection of operators are defined beyond those in [LNW06]. Our quotient operator facilitates the automated incremental construction of contracts for missing components, while conjunction combines independently developed requirements represented by multiple contracts. These features provide a range of additional checks for the validity of derived contracts, and support a truly contract-based design methodology.

5.4 Summary

This chapter has presented a compositional specification theory for reasoning about safety and progress properties of component behaviours, where we explicitly separate the assumptions made on the environment's behaviour from the guarantees provided by the component. The theory supports refinement based on traces, which relates specifications by implementation containment. We define the compositional operations of parallel composition, as well as – for the first time in this setting – conjunction, disjunction and quotient, directly on contracts. Sound and complete AG reasoning rules are provided for the four operators, preserving both safety and progress properties, which facilitates reasoning about, e.g., substitutivity of components synthesised at run-time.

The theory can be extended with hiding, providing a proper treatment of divergence is given for components, as reported in Chapter 3. Allowing divergence necessitates the extension of the

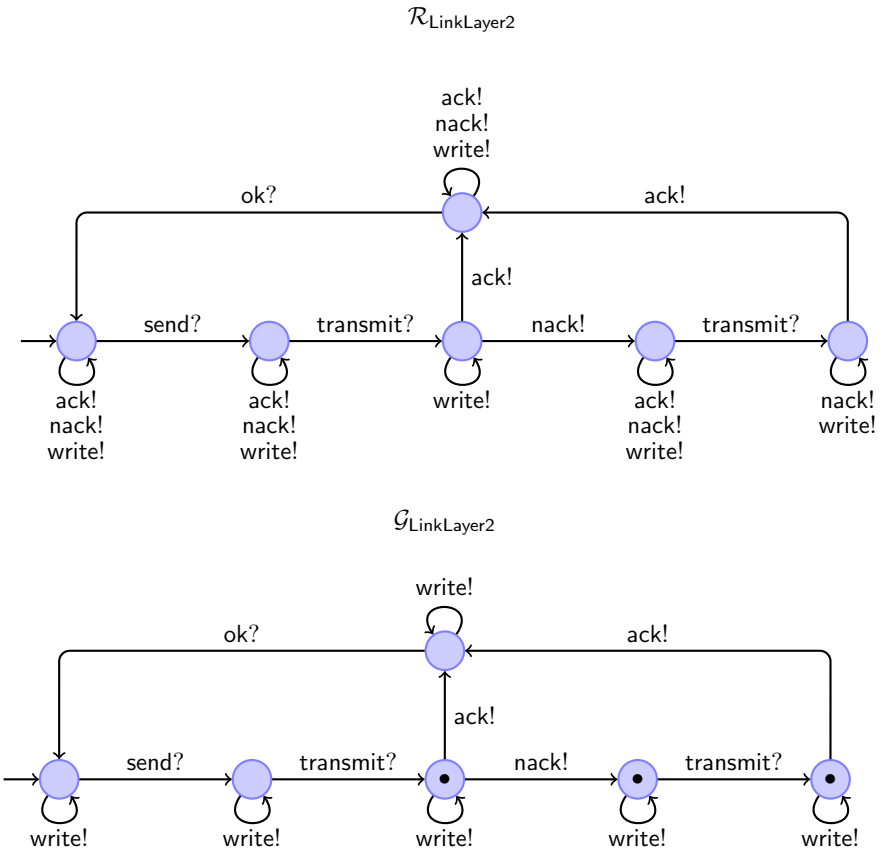

Figure 5.14: Assumption and guarantee of LinkLayer2 with full interface

contract framework (for progress) to include sets of traces that must not diverge, in addition to the traces that must make progress. This is in contrast to works such as [Jon94], which assume that a diverging process makes progress. The work of this chapter takes a more pragmatic view in requiring that progress is observable.

The AG rules can be fully automated, when restricting to regular properties (which can be represented by finite-state automata), as they are based on simple set-theoretic operations and do not require the learning of assumptions. The composition operations are polynomial-time constructions on finite-state automata, while the refinement relation can also be checked in polynomial-time, when the participating specifications are deterministic finite automata. In the general case of non-deterministic automata, refinement checking of individual contracts (i.e. $\mathcal{Q} \sqsubseteq \mathcal{P}$) is PSPACE-hard, as for the FDR model checker of CSP [Ros10], since the specification $\mathcal{P}$ needs to be normalised. However, in practice this tends not to be a limitation.

When considering systems of contracts, checking $\mathcal{Q}_1 \parallel \ldots \parallel \mathcal{Q}_k \sqsubseteq \mathcal{P}_1 \parallel \ldots \parallel \mathcal{P}_l$ is also PSPACE-hard, since the individual contracts can be normalised before performing the parallel composition, which preserves normalisation.

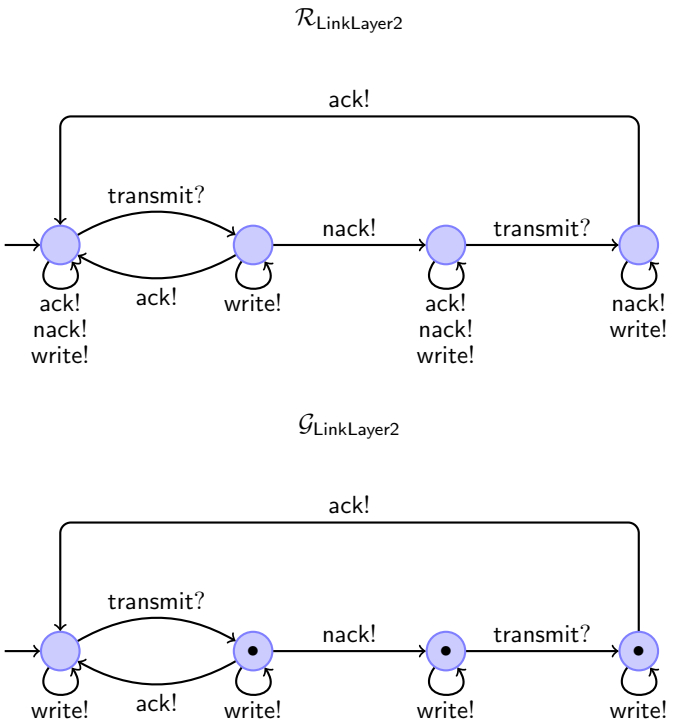

Figure 5.15: Assumption and guarantee of LinkLayer2 with restricted interface

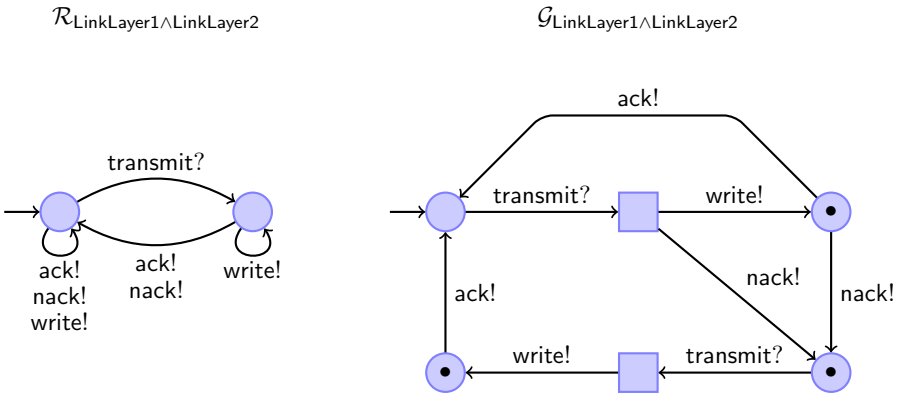

Figure 5.16: Assumption and guarantee of LinkLayer1 $\wedge$ LinkLayer2

CHAPTER

SIX

Theory of Timed Components

The compositional specification theory of Chapter 3 places constraints on the temporal ordering of interactions between components and the environment, but does not specify when these interactions actually occur. In this chapter, we generalise the theory of Chapter 3 to the real-time setting, where critical timing constraints are imposed on the interactions between components and the environment. We begin by constructing a theory of timed systems for which the global system clock can be stopped. This supports reasoning about systems that only have to run over a finite period of time, that is to say, they are able to terminate. Afterwards, we reformulate the theory to the setting where components must interact with the environment indefinitely, without the ability to stop the passage of time. The setting where time can be stopped is closely related to the substitutive framework of Section 3.1, whereas the non-terminating formulation shares similarities with the progress-sensitive framework of Section 3.2.

Both theories are capable of modelling *safety* and *bounded-liveness* errors. A safety error occurs through communication mismatches, or performance of an action that leads to an error state, as in Chapter 3. On the other hand, a bounded-liveness error occurs when a component passes a certain moment of time, potentially signifying that it has failed to perform an interaction earlier. As we will see, safety and bounded-liveness errors are equated in our theory.

6.1 Preliminaries

In this section, we briefly introduce some terminology for dealing with timed traces, and also redefine the lifting operators to the timed setting.

Timed words. Define $\mathbb{R}_0^+$ to be the set of non-negative real numbers and define $\mathbb{R}_0^\infty$ to be $\mathbb{R}_0^+ \cup \{\infty\}$, where ∞ is the special number defined such that $\sup(\mathbb{R}_0^\infty) = \infty$ and $\tau + \infty = \infty$ for each $\tau \in \mathbb{R}_0^+$. For a set of actions $\mathcal{A}$ disjoint from $\mathbb{R}_0^\infty$, the set of *timed words* over $\mathcal{A}$ is the union of:

- the finite timed words:

$$\{\tau_0 a_1 \tau_1 \ldots a_n \tau_n : n \geq 0,\, a_i \in \mathcal{A},\, \tau_i \in \mathbb{R}_0^+ \text{ for } i < n,\, t_n \in \mathbb{R}_0^\infty \text{ and } i < j \implies \tau_i \leq \tau_j\}$$

- and the infinite timed words:

$$\{\tau_0 a_1 \tau_1 a_2 \tau_2 \cdots : a_i \in \mathcal{A},\, \tau_i \in \mathbb{R}_0^+ \text{ and } i < j \implies \tau_i \leq \tau_j\}.$$

An untimed word $a_1 a_2 \ldots a_n \in \mathcal{A}^*$ can be interpreted as the timed word $0a_1 0a_2 0 \ldots 0a_n 0$.

For a finite timed word $t = \tau_0 a_1 \tau_1 \ldots a_m \tau_m$ and (possibly infinite) timed word $t' = \tau'_0 a'_1 \tau'_1 \ldots$, we define the *concatenation* of t and t', written tt', as the timed word t if $\tau_m = \infty$ and as the timed word $\tau_0 a_1 \tau_1 \ldots a_m (\tau_m + \tau'_0) a'_1 (\tau_m + \tau'_1) a'_2 (\tau_m + \tau'_2) \ldots$ otherwise. If t is an infinite timed word, then $tt' = t$. Based on concatenation, the timed word t' is said to be a *prefix* of t if there exists a timed word t'' such that $t't'' \equiv t$.

Zenoness and time convergence. An infinite timed word $t = \tau_0 a_1 \tau_1 a_2 \tau_2 \ldots$ is said to be time convergent if $\lim_{i\to\infty} \tau_i \in \mathbb{R}_0^+$, and is said to be time divergent otherwise (i.e., $\lim_{i\to\infty} \tau_i = \infty$). Moreover, an infinite timed word is said to be Zeno if an infinite number of actions are encountered within a finite amount of time. In our framework, time convergent traces are Zeno and vice versa; this is a consequence of our formulation of timed traces, since consecutive delays are not permitted. As time convergent and Zeno words cannot arise in practice, they will be treated as undesirable and will be removed from our component models. We use $\mathcal{T}(\mathcal{A})$ to denote the set of all non-Zeno timed words over $\mathcal{A}$. Note that concatenation and taking prefixes of non-Zeno timed traces maintains non-Zenoness.

Alphabet manipulations. Let $\mathcal{A}$ and $\mathcal{B}$ be sets of actions. For a timed trace t, write $t \upharpoonright (\mathcal{A} \cup \mathbb{R}_0^\infty)$ for the projection of t onto $\mathcal{A} \cup \mathbb{R}_0^\infty$ obtained by removing all actions not in $\mathcal{A} \cup \mathbb{R}_0^\infty$ and replacing runs of consecutive time values with the supremum of the values. Now, for $X \subseteq \mathcal{T}(\mathcal{A})$, write $X \upharpoonright (\mathcal{B} \cup \mathbb{R}_0^\infty)$ for $\{t \upharpoonright (\mathcal{B} \cup \mathbb{R}_0^\infty) : t \in X\}$, $X \Uparrow \mathcal{B}$ for $\{t \in \mathcal{T}(\mathcal{B}) : t \upharpoonright (\mathcal{A} \cup \mathbb{R}_0^\infty) \in X\}$ and $X \uparrow \mathcal{B}$ for $X \cdot (\mathcal{B} \setminus \mathcal{A}) \cdot \mathcal{T}(\mathcal{A} \cup \mathcal{B})$.

6.2 Terminating Theory of Timed Components

In this section, we present a real-time theory of components for systems that may terminate, meaning that the passage of time can be halted. This has practical applicability in modelling embedded systems and is also useful in circuit design (cf [MTC+00]). Since time cannot be stopped in reality, by termination we mean that we are not interested in the subsequent behaviour of a system of components.

We begin by giving the definition of a timed component, which is based on Definition 3.1 from the untimed setting.

Definition 6.1 (Timed component). A *timed component* $\mathcal{P}$ is a tuple $\langle \mathcal{A}^I_\mathcal{P}, \mathcal{A}^O_\mathcal{P}, T_\mathcal{P}, F_\mathcal{P} \rangle$ in which $\mathcal{A}^I_\mathcal{P}$ and $\mathcal{A}^O_\mathcal{P}$ are disjoint sets referred to as the inputs and outputs respectively (the union of which

is denoted by $\mathcal{A}_\mathcal{P}$), $T_\mathcal{P} \subseteq \mathcal{T}(\mathcal{A}_\mathcal{P})$ is a set of observable traces, and $F_\mathcal{P} \subseteq \mathcal{T}(\mathcal{A}_\mathcal{P})$ is a set of inconsistent traces. The trace sets must satisfy the constraints:

T1. $F_\mathcal{P} \subseteq T_\mathcal{P}$

T2. $T_\mathcal{P}$ is prefix closed

T3. If $t \in T_\mathcal{P}$ and $t' \in (\mathcal{A}^I_\mathcal{P})^*$, then $tt' \in T_\mathcal{P}$

T4. If $t \in F_\mathcal{P}$ and $t' \in \mathcal{T}(\mathcal{A}_\mathcal{P})$, then $tt' \in F_\mathcal{P}$.

If $0 \notin T_\mathcal{P}$, we say that $\mathcal{P}$ is *unrealisable*, and is *realisable* contrariwise. $\diamond$

This formulation of a timed component is near identical to the definition of a substitutive trace-based component as provided in Definition 3.1. The key differences are that the trace-sets are now timed, and inconsistent traces are extension-closed on timed traces. The conditions on the timed trace sets do not conflict with the non-Zenoness requirement on traces, since prefix closure and concatenation maintains non-Zenoness. Condition T3, which ensures input-receptivity, needs only append finite length input sequences to t, since an infinite sequence of inputs $t' \in (\mathcal{A}^I_\mathcal{P})^\omega$ would make tt' Zeno when t is a finite length trace of finite duration. Recall that the trace $a_1 \ldots a_n \in (\mathcal{A}^I_\mathcal{P})^*$ is interpreted as the timed trace $0a_1 0 \ldots 0a_n 0$.

Such a definition hints that the compositional operators of the timed theory are likely to be very similar to the operators defined in the untimed case, which they are.

From hereon let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be timed 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.

6.2.1 Operational Representation of a Timed Component

In order to provide examples demonstrating the features of our timed formalism, we require an operational representation for timed components, as in Chapter 3, so that components may be represented pictorially. Unlike in Chapter 4, we do not provide a complete theory of operational components by defining the compositional operators; that is deferred as future work.

Timed components are modelled using finite-state automata, but with the addition of timing information. The underlying automaton specifies the temporal ordering of interactions, while guards on transitions, and invariants and co-invariants on states, specify the timing constraints. These are given with respect to clock-variables, which can be reset to 0, as for timed automata [AD94]. Figure 6.1 shows the multi-function device of Chapter 3 with timing constraints.

Transitions are labelled with either an input or output interaction. In the case of output labelings, the guard specifies the possible times when the component is willing to fire that transition, while for transitions labelled by inputs the guard specifies the available times when the component is willing to accept such an interaction from the environment. Interactions are assumed to

take place instantaneously, meaning that they have zero duration. Therefore, the component must sojourn in its states. An invariant on a state is used to specify the bound beyond which time is not permitted to pass. A co-invariant, on the other hand, specifies the bound beyond which the component becomes inconsistent, most likely because an interaction did not take place earlier.

In the spirit of timed automata, the timing constraints are specified using Boolean combinations of comparisons between clock variables and rational time values. Invariants are a concept that have already been defined for timed automata, while the notion of a co-invariant, which is particularly relevant to I/O systems, is new. Considering how the two interplay, if the invariant is violated in a state before or at the same time as the co-invariant, time stops there and then. However, if the co-invariant is violated first, then the component becomes inconsistent for all time values beyond that point, so the invariant has no effect.

We now introduce the notation used for representing timing constraints.

Definition 6.2. Let $\mathcal{X}$ be a global set of clock variables. For $X \subseteq \mathcal{X}$, define $\Phi(X)$ to be the set of Boolean combinations of clock constraints φ over X:

$$\varphi := tt \mid x \leq k \mid x < k \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi,$$

where $x \in X$ and $k \in \mathbb{Q}_0^+$. The constraint φ is downward-closed if it does not include $\neg$. $\diamond$

A valuation for a set of clock variables $X \subseteq \mathcal{X}$ is a function $v : X \longrightarrow \mathbb{R}_0^\infty$ that assigns time values to the clock variables. For $Y \subseteq X$, we write $v[Y := 0]$ for the valuation that maps all clocks in Y to 0, and leaves all other clocks unchanged from the valuation v. Similarly, for $d \in \mathbb{R}_0^\infty$, we write $v + d$ for the valuation that adds d to the v-valuation of each clock variable. The notation 0^X is used to denote the valuation that maps all clocks in X to 0.

Given these preliminary definitions, we can define a timed operational component.

Definition 6.3. A *timed operational component* P is a tuple $\langle \mathcal{A}_\mathsf{P}^I, \mathcal{A}_\mathsf{P}^O, X_\mathsf{P}, S_\mathsf{P}, B_\mathsf{P}, \rightarrow_\mathsf{P}, s_\mathsf{P}^0, inv_\mathsf{P}, co_\mathsf{P} \rangle$, where:

- $\mathcal{A}_\mathsf{P}^I$ is a finite set of input actions
- $\mathcal{A}_\mathsf{P}^O$ is a finite set of output actions, disjoint from $\mathcal{A}_\mathsf{P}^I$, where $\mathcal{A}_\mathsf{P} \triangleq \mathcal{A}_\mathsf{P}^I \cup \mathcal{A}_\mathsf{P}^O$
- X_P is a finite collection of clock variables
- S_P is a finite set of states
- $B_\mathsf{P} \subseteq S_\mathsf{P}$ is the set of Büchi accepting states
- $\rightarrow_\mathsf{P} \subseteq S_\mathsf{P} \times \mathcal{A}_\mathsf{P} \times \Phi(X_\mathsf{P}) \times 2^{X_\mathsf{P}} \times S_\mathsf{P}$ is the transition relation
- $s_\mathsf{P}^0 \in S_\mathsf{P}$ is the designated initial state

- $inv_{\mathsf{P}} : S_{\mathsf{P}} \longrightarrow \Phi(X_{\mathsf{P}})$ is the invariant, a mapping to downward-closed clock constraints
- $co_{\mathsf{P}} : S_{\mathsf{P}} \longrightarrow \Phi(X_{\mathsf{P}})$ is the co-invariant, a mapping to downward-closed clock constraints.

The component must satisfy the additional constraint that $(s, a, g, Y, s') \in \rightarrow_{\mathsf{P}}$ when $a \in \mathcal{A}^I_{\mathsf{P}}$ only if for each valuation v it holds that $inv_{\mathsf{P}}(s)(v)$ and $g(v)$ implies $inv_{\mathsf{P}}(s')(v[Y := 0])$. $\diamond$

We use the notation $s \xrightarrow{a,g,Y}_{\mathsf{P}} s'$ to mean that $(s, a, g, Y, s') \in \rightarrow_{\mathsf{P}}$, which can be interpreted as: if the component is in state s, the guard g is satisfied and the interaction a is performed, then the component moves to state s' and resets all clocks in Y to 0.

The final constraint in the definition above ensures that a component cannot avoid being receptive to inputs by terminating. This is essential, because our trace-based formulation of components is required to be input-receptive. In fact, such a restriction can be dropped without affecting the results in Section 6.2, as it leads to a simplification of the theory, and actually brings it inline with the design decision adopted in [CKW12].

Based on the operational definition of a timed component, we formulate the definition of a timed transition system, which can be used to ascribe semantics to the compact automaton model, by making explicit the time delays. The approach is largely based on that adopted for timed automata, but with some subtle modifications to account for non-enabled inputs and inconsistency.

Definition 6.4. The semantics of P is a timed transition system $[\![\mathsf{P}]\!] = \langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, S, s_0, \longrightarrow \rangle$, where:

- $S \triangleq (S_{\mathsf{P}} \times (\mathbb{R}^\infty_0)^{X_{\mathsf{P}}} \times \{act, delay\}) \uplus \{\bot\}$ is the set of configurations, consisting of state-valuation-successor triples (act and $delay$ indicate the next type of transition permitted), plus the special inconsistent state $\bot$
- $s_0 = \begin{cases} (s^0_{\mathsf{P}}, 0^{X_{\mathsf{P}}}, delay) & \text{if } inv_{\mathsf{P}}(s^0_{\mathsf{P}})(0^{X_{\mathsf{P}}}) \text{ and } co_{\mathsf{P}}(s^0_{\mathsf{P}})(0^{X_{\mathsf{P}}}) \\ \bot & \text{if } inv_{\mathsf{P}}(s^0_{\mathsf{P}})(0^{X_{\mathsf{P}}}) \text{ and } \neg co_{\mathsf{P}}(s^0_{\mathsf{P}})(0^{X_{\mathsf{P}}}) \\ \text{undefined} & \text{if } \neg inv_{\mathsf{P}}(s^0_{\mathsf{P}})(0^{X_{\mathsf{P}}}) \end{cases}$
- $\longrightarrow \subseteq S \times (\mathcal{A}_{\mathsf{P}} \cup \mathbb{R}^\infty_0) \times S$ is the smallest set satisfying:
  - $(s, v, delay) \xrightarrow{t} (s, v + t, action)$, if $inv_{\mathsf{P}}(s)(v + t)$ and $co_{\mathsf{P}}(s)(v + t)$
  - $(s, v, delay) \xrightarrow{t} \bot$, if $inv_{\mathsf{P}}(s)(v + t)$ and $\neg co_{\mathsf{P}}(s)(v + t)$
  - If $s \xrightarrow{a,g,Y}_{\mathsf{P}} s'$, $g(v)$ and $inv_{\mathsf{P}}(s')(v[Y := 0])$ then:
    - $(s, v, action) \xrightarrow{a} (s', v[Y := 0], delay)$, providing $co_{\mathsf{P}}(s')(v[Y := 0])$
    - $(s, v, action) \xrightarrow{a} \bot$, providing $\neg co_{\mathsf{P}}(s')(v[Y := 0])$
  - $(s, v, action) \xrightarrow{a} \bot$, if $a \in \mathcal{A}^I_{\mathsf{P}}$ and $\nexists g, Y, s' \cdot g(v)$ and $s \xrightarrow{a,g,Y} s'$. $\diamond$

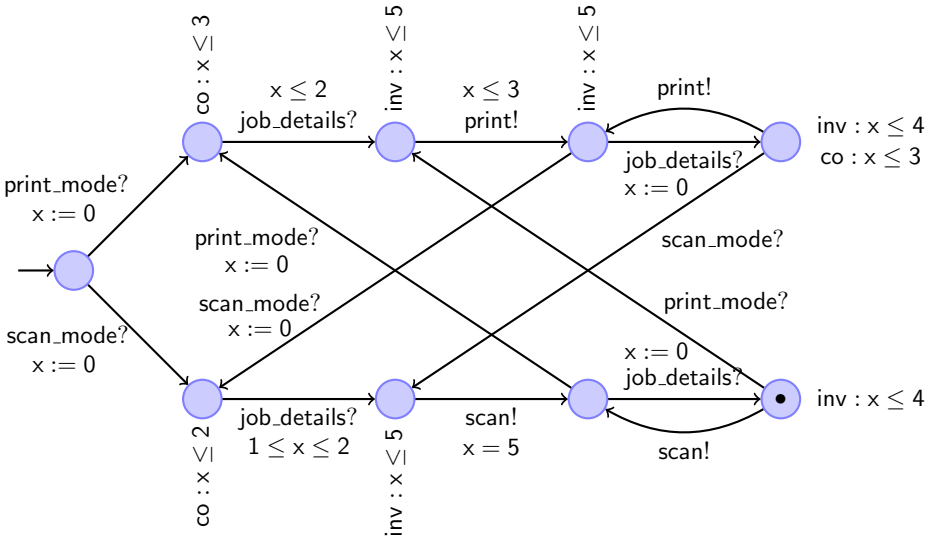

Figure 6.1: Timed printing/scanning device

The semantics of an operational model provides all the information required for generating a trace-based component representing the behaviour of the operational model. Within such a model, the special state $\perp$ is used to encode safety and bounded-liveness errors.

We now explain how the traces can be extracted from the timed transition system, by consideration of its accepted runs. A finite trace $t = \tau_0 a_1 \tau_1 \dots a_n \tau_n$ is accepted by $[\![\mathsf{P}]\!]$ if there exist configurations $s_1, \dots, s_{2n+1}$ such that $s_0 \xrightarrow{\tau_0} s_1 \xrightarrow{a_1} s_2 \dots s_{2n-1} \xrightarrow{a_n} s_{2n} \xrightarrow{\tau_n} s_{2n+1}$. An infinite trace $t = \tau_0 a_1 \tau_1 \dots$ is accepted by $[\![\mathsf{P}]\!]$ if there exist configurations $s_1, s_2, \dots$ such that $s_0 \xrightarrow{\tau_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{\tau_1} \dots$ and there exists a state $s \in B_{\mathsf{P}}$ which occurs infinitely often in the configurations along the run. For a possibly infinite trace $t = \tau_0 a_1 \tau_1 a_2 \tau_2 \dots$, we use $\%t$ to denote the trace $\tau_0 a_1 (\tau_1 - \tau_0) a_2 (\tau_2 - \tau_1) \dots$.

Definition 6.5. The timed component $[\![\mathsf{P}]\!]^*$ represented by P is the structure $\langle \mathcal{A}^I_{\mathsf{P}}, \mathcal{A}^O_{\mathsf{P}}, T_{[\![\mathsf{P}]\!]^*}, F_{[\![\mathsf{P}]\!]^*} \rangle$, where:

- $T_{[\![\mathsf{P}]\!]^*} = \{t \in \mathcal{T}(\mathcal{A}_{\mathsf{P}}) : \%t \text{ is a trace of } [\![\mathsf{P}]\!]\} \cup F_{[\![\mathsf{P}]\!]^*}$
- $F_{[\![\mathsf{P}]\!]^*} = \{t \in \mathcal{T}(\mathcal{A}_{\mathsf{P}}) : \text{there is a finite run over } \%t \text{ in } [\![\mathsf{P}]\!] \text{ ending in } \perp\} \cdot \mathcal{T}(\mathcal{A}_{\mathsf{P}})$. ◇

Note that Definition 6.5 yields a timed component in the sense of Definition 6.1 (i.e., all of the conditions on Definition 6.1 are satisfied). Thus, we can use the operational representation for a timed component when presenting examples throughout this section. The pictorial representation of operational components, as in Figure 6.1, is self-explanatory, although we comment that states containing • are designated as being Büchi accepting, and the invariant and co-invariant are not explicitly mentioned when they are true.

Example 6.6. Figure 6.1 shows a timed component representing the multi-function device first

introduced in Figure 3.1, but with the addition of timing constraints. Explaining the behaviour, the device waits until it is placed in print_mode or scan_mode. When placed in print_mode, the component expects some input from the user within 3 time units (specified by the co-invariant), but the only acceptable input is job_details, which must be received within 2 time units. If job_details is not supplied within 3 time units, the component will generate a bounded-liveness error, while if the input is issued at a time greater than 2, but at most 3, time units later, a communication mismatch (safety error) will occur. After having successfully received job_details, the component can print the document within 3 time units of having being placed in print_mode, and otherwise the device will sit idly for 5 time units, after which it will terminate due to the expiration of the invariant. Once the document has printed, the user must either issue more job_details, or must place the device in scan_mode, both within 5 time units of the device having being placed in print_mode, otherwise the component will terminate. If job_details are received, the component must either receive a request to be placed in scan_mode, or must print the document, within 3 time units, otherwise a bounded-liveness error will occur (represented by the co-invariant). Note that the invariant in the upper right-hand state has no effect, because the co-invariant (which expires earlier) takes precedence. The scan_mode functionality can be explained similarly.

Note that, if the device is placed and left in print_mode, then the device is only permitted to print a finite number of times, since there are no Büchi accepting states in the printing loop. This is in contrast to the scan_mode functionality, which allows the device to scan infinitely often when it is left in scan_mode. ◇

6.2.2 Refinement

Substitutive refinement in the timed setting prevents the introduction of safety and bounded liveness errors. Safety errors encode the inconsistencies mentioned in Chapter 3, which encompass communication mismatches, run-time errors and underspecification. On the other hand, a bounded liveness error occurs when a timed trace becomes inconsistent by allowing time to pass a certain point, rather than encountering a bad action. In essence, this means that the component has failed to engage in some interaction within the specified time bound.

As in the untimed case, we define the safe representation of a component, which propagates inconsistencies backwards over output actions, since the environment cannot influence the behaviour of these interactions. It is not correct to propagate inconsistencies back over timed output words, since the environment is able to prevent the passage of time. As a direct consequence, a refining component is not permitted to become inconsistent earlier than the original component.

Definition 6.7. 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 $F_{\mathcal{E}(\mathcal{P})} = \{t \in T_\mathcal{P} : \exists t' \in (\mathcal{A}^O_\mathcal{P})^* \cdot tt' \in F_\mathcal{P}\} \cdot \mathcal{T}(\mathcal{A}_\mathcal{P})$. ◇

Given the reformulated safe specification of a component, we can give the definition of refine-

ment, which is essentially unchanged from the untimed setting.

Definition 6.8 (Refinement). $\mathcal{Q}$ is said to be a *timed refinement* of $\mathcal{P}$, written $\mathcal{Q} \sqsubseteq^t_{imp} \mathcal{P}$, iff:

TR1. $\mathcal{A}^I_{\mathcal{P}} \subseteq \mathcal{A}^I_{\mathcal{Q}}$

TR2. $\mathcal{A}^O_{\mathcal{Q}} \subseteq \mathcal{A}^O_{\mathcal{P}}$

TR3. $\mathcal{A}^I_{\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$

TR4. $T_{\mathcal{E}(\mathcal{Q})} \subseteq T_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})$

TR5. $F_{\mathcal{E}(\mathcal{Q})} \subseteq F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{Q}})$. ◇

As usual, conditions TR1 and TR2 ensure that the interfaces of $\mathcal{P}$ and $\mathcal{Q}$ are substitutive, while TR3 in conjunction with the previous two conditions guarantees that the components cannot mix action types (that is to say they are compatible). Condition TR4 ensures that the observable behaviour of $\mathcal{Q}$ is also observable in $\mathcal{P}$ (or lies outside of $\mathcal{P}$'s interface), while condition TR5 stipulates that $\mathcal{Q}$'s inconsistent behaviour must also be inconsistent in $\mathcal{P}$ (or lie outside of $\mathcal{P}$'s interface).

Equivalence of timed components corresponds to substitutive equivalence, so is defined in terms of mutual refinement.

Definition 6.9. $\mathcal{P}$ and $\mathcal{Q}$ are *equivalent*, written $\mathcal{P} \equiv^t_{imp} \mathcal{Q}$, iff $\mathcal{P} \sqsubseteq^t_{imp} \mathcal{Q}$ and $\mathcal{Q} \sqsubseteq^t_{imp} \mathcal{P}$. ◇

Example 6.10. Figure 6.2 shows a number of timed components. $(a) \sqsubseteq^t_{imp} (b)$, meaning that a safety or bounded-liveness error arising through the interaction of (a) in an environment $\mathcal{E}$ implies there is a safety or bounded-liveness error in the interaction between (b) and $\mathcal{E}$. $(b) \not\sqsubseteq^t_{imp} (a)$ due to the trace $\langle 3, a, 3\rangle$ being consistent in (a), while it is inconsistent in (b). Even though $\langle 3, a, 3.1\rangle$ is inconsistent in (a), an environment can force (a) to terminate at time $x = 3$, thus avoiding the expiration of the co-invariant and the associated bounded-liveness error.

Similarly, $(a) \sqsubseteq^t_{imp} (c)$ and $(c) \not\sqsubseteq^t_{imp} (a)$, the latter being a consequence of $\langle 2.5, a, 2.5\rangle$ being a consistent trace of (a), while it is inconsistent in (c). For (b) and (c), we have that $(b) \equiv^t_{imp} (c)$, due to the fact that any a action being issued when $2 < x \leq 3$ results in a bounded-liveness error in (b), while it results in a safety error in (c), since the component is not enabled on that action. All other behaviours are the same. ◇

As refinement is defined in terms of set containment, and its formulation is not altered from the untimed setting, it is not surprising that the relation is a preorder (subject to compatibility).

Lemma 6.11. Refinement is reflexive, and is transitive subject to preservation of action types: $\mathcal{R} \sqsubseteq^t_{imp} \mathcal{Q}$, $\mathcal{Q} \sqsubseteq^t_{imp} \mathcal{P}$ and $\mathcal{A}^I_{\mathcal{R}} \cap \mathcal{A}^O_{\mathcal{P}} = \emptyset$ implies $\mathcal{R} \sqsubseteq^t_{imp} \mathcal{P}$.

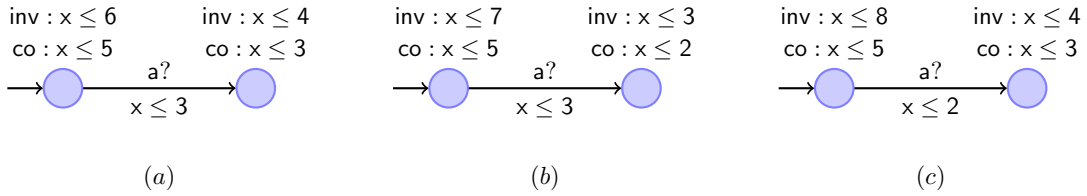

Figure 6.2: Timed refinement

Proof. Follows by transitivity of subset containment and the reasoning in Lemma 3.6. □

We are now in a position to define the compositional operators of the theory, which are in general only defined on composable components, the conditions of which remain unchanged from Chapter 3. Note that the definitions make use of the redefined $\Uparrow$ and $\uparrow$ operators from Section 6.1.

6.2.3 Parallel Composition

The parallel composition of timed components with disjoint output alphabets is a component representing the combined effect of the components running asynchronously.

Definition 6.12. 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 $\mathcal{P} \;||_t\; \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}||\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}||\mathcal{Q}}, T_{\mathcal{P}||_t\mathcal{Q}}, F_{\mathcal{P}||_t\mathcal{Q}} \rangle$, where:

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

This definition of parallel composition differs from Definition 3.7 in that it requires the inconsistent trace set $F_{\mathcal{P}||\mathcal{Q}}$ to be closed under extension of timed words $\mathcal{T}(\mathcal{A}_{\mathcal{P}||\mathcal{Q}})$, rather than $\mathcal{A}^*_{\mathcal{P}||\mathcal{Q}}$. The lifting operation $\Uparrow$ ensures that $\mathcal{P}$ and $\mathcal{Q}$ synchronise on time and common actions, while they interleave on independent actions. Therefore, if one component can stop the passage of time, then their composition will also stop time.

Lemma 6.13. Parallel composition is associative and commutative.

Proof. The result essentially follows from Lemma 3.9. □

It is easy to show that parallel composition is monotonic under timed refinement, subject to the same interface constraints as in the untimed setting.

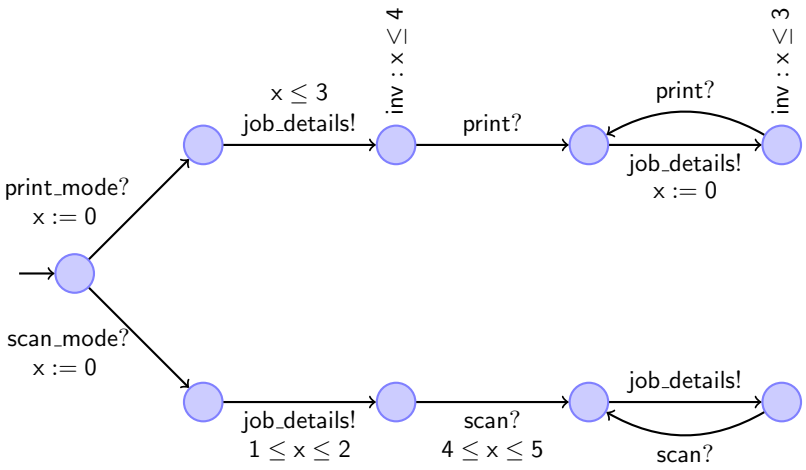

Figure 6.3: Timed spooler

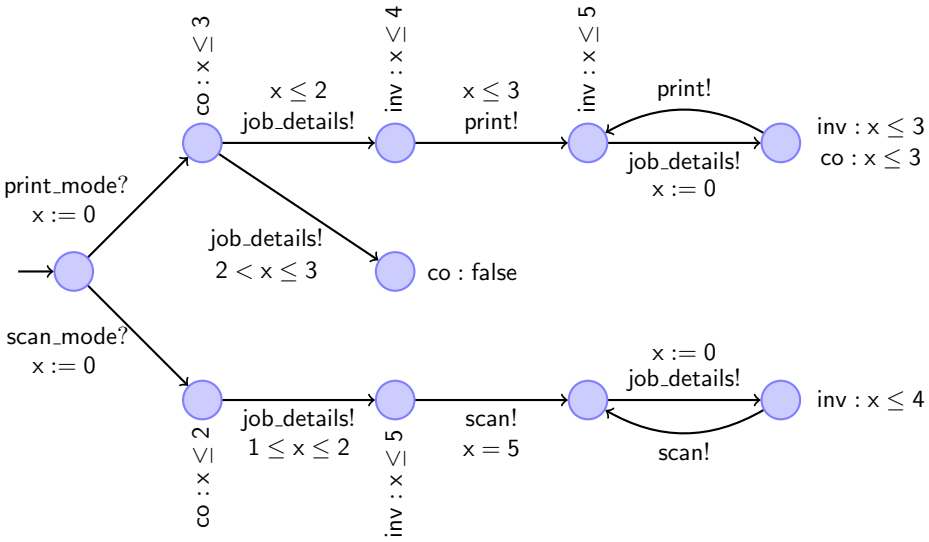

Figure 6.4: Timed parallel composition of the printing/scanning device and the spooler

Theorem 6.14. Let $\mathcal{P}$, $\mathcal{Q}$, $\mathcal{P}'$ and $\mathcal{Q}'$ be timed 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 $\mathcal{P}' \sqsubseteq^t_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^t_{imp} \mathcal{Q}$, then $\mathcal{P}' \,||_t\, \mathcal{Q}' \sqsubseteq^t_{imp} \mathcal{P} \,||_t\, \mathcal{Q}$.

Proof. Straightforward modification to Theorem 3.10. Projections of the form $\restriction \mathcal{A}$ should be replaced with $\restriction (\mathcal{A} \cup \mathbb{R}^\infty_0)$. Note that the techniques used in the proof of Theorem 3.10 are equally applicable to both finite- and infinite-length traces. □

Example 6.15. Figure 6.3 shows a spooler (a device responsible for issuing job_details to the printing/scanning device). The parallel composition of the spooler with the printing/scanning device is shown in Figure 6.4. When placed into print_mode, a communication mismatch will occur if the spooler attempts to send the job_details after 2 time units and at most 3 time units have

passed (this is indicated by the transition to the state having a false co-invariant). Furthermore, it is no longer safe for the device mode to be switched after the initial choice (this is a constraint imposed by the spooler), and when in scan_mode it is no longer possible for scanning to be performed infinitely often, since the spooler does not have any Büchi accepting states. ◇

6.2.4 Conjunction

As conjunction should correspond to the meet operator on the refinement preorder, the behaviour of the conjunction should be contained within the behaviours of its arguments. Therefore, conjunction must stop the passage of time if at least one of its arguments can do so.

Definition 6.16. 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 $\mathcal{P} \wedge_t \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}}, T_{\mathcal{P}\wedge_t\mathcal{Q}}, F_{\mathcal{P}\wedge_t\mathcal{Q}} \rangle$, where:

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

This definition is syntactically identical to Definition 3.12, but note that, as $T_\mathcal{P}$ is a set of timed traces, it follows that $T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q}$ is closed under extensions of timed words over the alphabet $\mathcal{A}_\mathcal{P} \cup \mathcal{A}^I_\mathcal{Q}$ (and similarly for $T_{\mathcal{E}(\mathcal{Q})} \uparrow \mathcal{A}^I_\mathcal{P}$), due to the way that $\uparrow$ is defined. Consequently, on encountering an input that lies outside the interface of one of the components, that component can no longer constrain the behaviour of the other component, and nor can it prevent the passage of time.

Lemma 6.17. Conjunction is associative, commutative and idempotent.

Proof. Obvious, given the set-theoretic definition of conjunction. □

The following theorem shows that timed conjunction enjoys the same strong algebraic properties as in the untimed setting. In particular, conjunction is the meet operator on the timed refinement preorder.

Theorem 6.18. Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$, be components composable for conjunction. Then:

- $\mathcal{P} \wedge_t \mathcal{Q} \sqsubseteq^t_{imp} \mathcal{P}$ and $\mathcal{P} \wedge_t \mathcal{Q} \sqsubseteq^t_{imp} \mathcal{Q}$
- $\mathcal{R} \sqsubseteq^t_{imp} \mathcal{P}$ and $\mathcal{R} \sqsubseteq^t_{imp} \mathcal{Q}$ implies $\mathcal{R} \sqsubseteq^t_{imp} \mathcal{P} \wedge_t \mathcal{Q}$

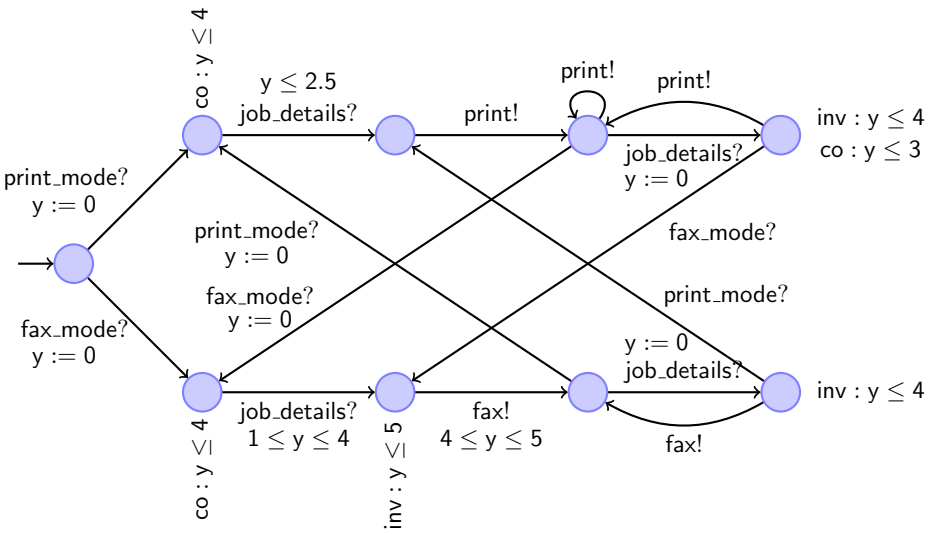

Figure 6.5: Timed printing/faxing device

- $\mathcal{P}' \sqsubseteq^t_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^t_{imp} \mathcal{Q}$ implies $\mathcal{P}' \wedge_t \mathcal{Q}' \sqsubseteq^t_{imp} \mathcal{P} \wedge_t \mathcal{Q}$.

Proof. Follows by elementary modification to Theorem 3.14, since the original theorem makes reference to arbitrary traces, without consideration of their length or structure. □

Example 6.19. Figure 6.5 ascribes timing constraints to the print/fax device represented in Figure 3.2. The conjunction of the print/scan and print/fax devices (in Figures 6.1 and 6.5, respectively) is shown in Figure 6.6, when the interfaces of each device are based on the actions appearing in their respective diagrams. The conjunction arguably looks cumbersome, but its behaviour can be explained clearly in terms of the components that have been conjoined, by appealing to Figure 3.3.

First, the behaviour associated with putting the device in scan_mode or fax_mode matches that in Figure 3.3, with the timing constraints being lifted straight out of Figures 6.1 and 6.5. This is because scan_mode and fax_mode are mutually independent functions of the devices to be conjoined.

Placing the device in print_mode needs more careful examination, as this is a behaviour common to both devices. Note that, after being placed in print_mode, the co-invariant associated with the conjunctive device is the disjunction of the co-invariants in the sub-devices, since the conjunction is only allowed to be inconsistent when both of the sub-devices are inconsistent. Now, the issuing of job_details is partitioned according to the guards in the devices to be conjoined. The upper transition corresponds to where the sub-devices overlap, whereas the lower one is for the print/fax device, since the print/scan device will be inconsistent, due to a communication mismatch arising when job_details are issued at those particular times. Therefore, the upper transition must respect the requirements of both devices, whereas the lower one leaves the print/fax device

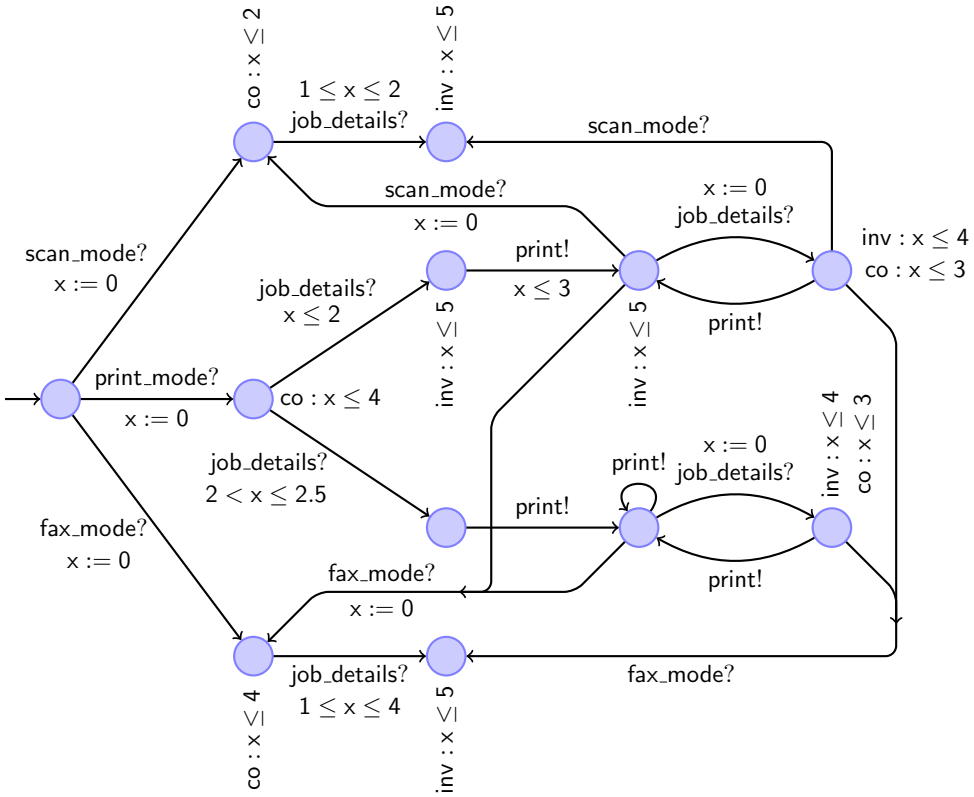

Figure 6.6: Timed conjunction of the printing/scanning and printing/faxing devices

unconstrained. This is why printing can be repeatedly performed on the lower transition, while it is not the case it can be repeatedly performed on the upper transition, due to it not being a common output behaviour of the two devices. ◇

6.2.5 Disjunction

Being the dual of conjunction, each behaviour of the disjunction should be a behaviour in at least one of its arguments. Therefore, time should only be stopped providing it is stopped in both of the components to be composed.

Definition 6.20. 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 $\mathcal{P} \vee_t \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}}, T_{\mathcal{P}\vee_t\mathcal{Q}}, F_{\mathcal{P}\vee_t\mathcal{Q}} \rangle$, where:

- $\mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}} = \mathcal{A}^I_\mathcal{P} \cap \mathcal{A}^I_\mathcal{Q}$
- $\mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}} = \mathcal{A}^O_\mathcal{P} \cup \mathcal{A}^O_\mathcal{Q}$
- $T_{\mathcal{P}\vee_t\mathcal{Q}} = [(T_\mathcal{P} \cup T_\mathcal{Q}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})] \cup F_{\mathcal{P}\vee_t\mathcal{Q}}$
- $F_{\mathcal{P}\vee_t\mathcal{Q}} = [(F_\mathcal{P} \cup F_\mathcal{Q}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})] \cdot \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$. ◇

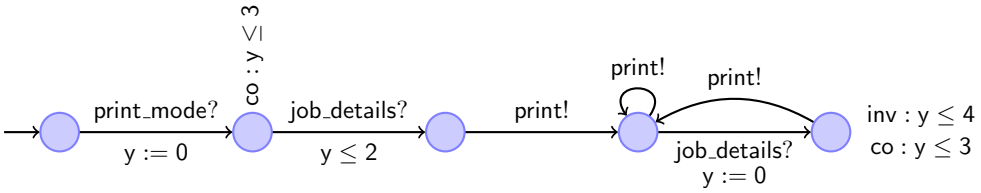

Figure 6.7: Timed disjunction of the printing/scanning and printing/faxing devices

This definition differs from Definition 3.16 in that $T_{\mathcal{P}\vee_t\mathcal{Q}}$ and $F_{\mathcal{P}\vee_t\mathcal{Q}}$ should be restricted to timed words in $\mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$, hence the intersection with this set, rather than $\mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$, and inconsistent traces should be closed under extensions with timed words over $\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}}$.

Lemma 6.21. Disjunction is associative, commutative and idempotent.

Proof. Follows by Lemma 3.17, given that the proof is based on the syntactic constructions of $T_{\mathcal{P}\vee\mathcal{Q}}$ and $F_{\mathcal{P}\vee\mathcal{Q}}$. □

Disjunction enjoys strong algebraic properties in that it is the join operator on the refinement preorder and is monotonic under refinement.

Theorem 6.22. Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$, be components composable for disjunction. Then:

- $\mathcal{P} \sqsubseteq^t_{imp} \mathcal{P} \vee_t \mathcal{Q}$ and $\mathcal{Q} \sqsubseteq^t_{imp} \mathcal{P} \vee_t \mathcal{Q}$
- $\mathcal{P} \sqsubseteq^t_{imp} \mathcal{R}$ and $\mathcal{Q} \sqsubseteq^t_{imp} \mathcal{R}$ implies $\mathcal{P} \vee_t \mathcal{Q} \sqsubseteq^t_{imp} \mathcal{R}$
- $\mathcal{P}' \sqsubseteq^t_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^t_{imp} \mathcal{Q}$ implies $\mathcal{P}' \vee_t \mathcal{Q}' \sqsubseteq^t_{imp} \mathcal{P} \vee_t \mathcal{Q}$.

Proof. Follows from straightforward modification to Theorem 3.18 by replacing references to $\mathcal{A}^*_{\mathcal{P}\vee\mathcal{Q}}$ with $\mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$. □

Example 6.23. Figure 6.7 shows the disjunction of the printing/scanning device (Figure 6.1) and the printing/faxing device (Figure 6.5), when the interfaces of the respective devices consist of the actions arising in each of their models. Therefore, the independent functions of scan_mode and fax_mode are removed, while the union of the print_mode observable and inconsistent behaviours is included. Consequently, the disjunction can become inconsistent through a bounded-liveness error 3 time units after having been placed in print_mode, and job_details need only be safely accepted within 2 time units. The remaining invariants and co-invariants (excluding those in the right-most state) are true, since they are true in the printing/faxing device. ◇

6.2.6 Hiding

Hiding is an operation that contracts the interface of a component, meaning that it can remove an interaction primitive that the component has the potential to issue or can observe. Therefore, it

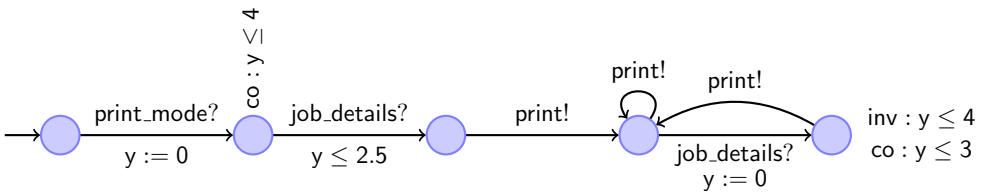

Figure 6.8: Hiding fax_mode in the printing/faxing device

does not make sense to hide a time delay, and so we explicitly rule out this possibility.

Definition 6.24. Let $\mathcal{P}$ be a component and let b be an action. The hiding of b in $\mathcal{P}$ is a component $\mathcal{P} /_t b = \langle \mathcal{A}^I_{\mathcal{P}/b}, \mathcal{A}^O_{\mathcal{P}/b}, T_{\mathcal{P}/_t b}, F_{\mathcal{P}/_t b} \rangle$, where:

- $\mathcal{A}^I_{\mathcal{P}/b} = \mathcal{A}^I_{\mathcal{P}} \setminus \{b\}$
- $\mathcal{A}^O_{\mathcal{P}/b} = \mathcal{A}^O_{\mathcal{P}} \setminus \{b\}$
- $T_{\mathcal{P}/_t b} = \begin{cases} T_{\mathcal{P}} \upharpoonright (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) & \text{if } b \in \mathcal{A}^O_{\mathcal{P}} \\ T_{\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b}) & \text{otherwise} \end{cases}$
- $F_{\mathcal{P}/_t b} = \begin{cases} F_{\mathcal{P}} \upharpoonright (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) & \text{if } b \in \mathcal{A}^O_{\mathcal{P}} \\ F_{\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b}) & \text{otherwise.} \end{cases}$ ◇

Modifications to this definition from the untimed setting (Definition 3.20) include projecting onto timed words over $\mathcal{A}_{\mathcal{P}/b}$ when b is an output, rather than just onto $\mathcal{A}_{\mathcal{P}/b}$, and restricting trace sets to be contained in $\mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$ when b is not an output, rather than $\mathcal{A}^*_{\mathcal{P}/b}$. These changes leave the monotonicity result unaffected.

Theorem 6.25. Let $\mathcal{P}$ and $\mathcal{Q}$ be components and let b be an action. If $\mathcal{Q} \sqsubseteq^t_{imp} \mathcal{P}$, then $\mathcal{Q} /_t b \sqsubseteq^t_{imp} \mathcal{P} /_t b$.

Proof. Follows by the reasoning in Theorem 3.21 when making use of timed projections. □

Example 6.26. Figure 6.8 shows the print/fax device of Figure 6.5, after having hidden the fax_mode functionality. The print_mode functionality remains unchanged. ◇

6.2.7 Quotient

The definition of quotient in the timed framework is a straightforward generalisation from the untimed setting by incorporating time-sensitive projections. This ensures that a timed trace in the

parallel composition of the sub-component $\mathcal{P}$ and the quotient must also be a timed trace in the specification $\mathcal{R}$.

Definition 6.27. 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 $\mathcal{R} /_t \mathcal{P}$ with signature $\langle \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}, \mathcal{A}^O_{\mathcal{R}/\mathcal{P}}, T_{\mathcal{R}/_t\mathcal{P}}, F_{\mathcal{R}/_t\mathcal{P}} \rangle$, where:

- $\mathcal{A}^I_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^I_\mathcal{R} \setminus \mathcal{A}^I_\mathcal{P}$
- $\mathcal{A}^O_{\mathcal{R}/\mathcal{P}} = \mathcal{A}^O_\mathcal{R} \setminus \mathcal{A}^O_\mathcal{P}$
- $T_{\mathcal{R}/_t\mathcal{P}}$ is the largest prefix-closed and input-receptive subset of
  $\{t \in \mathcal{T}(\mathcal{A}_{\mathcal{R}/\mathcal{P}}) : \forall t' \in \mathcal{T}(\mathcal{A}_\mathcal{R}) \cdot t' \restriction (\mathcal{A}_{\mathcal{R}/\mathcal{P}} \cup \mathbb{R}^\infty_0) = t$ implies
  $[t' \restriction (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in T_\mathcal{P} \implies t' \in T_{\mathcal{E}(\mathcal{R})}]$ and
  $[t' \restriction (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in F_\mathcal{P} \implies t' \in F_{\mathcal{E}(\mathcal{R})}]\}$
- $F_{\mathcal{R}/_t\mathcal{P}} = \{t \in T_{\mathcal{R}/_t\mathcal{P}} : \forall t' \in \mathcal{T}(\mathcal{A}_\mathcal{R}) \cdot t' \restriction (\mathcal{A}_{\mathcal{R}/\mathcal{P}} \cup \mathbb{R}^\infty_0) = t$ implies
  $[t' \restriction (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in T_\mathcal{P} \implies t' \in F_{\mathcal{E}(\mathcal{R})}]\}$. ◇

The definition of $T_{\mathcal{R}/_t\mathcal{P}}$ requires that the defining set is both prefix-closed and input-receptive, since it is not necessarily the case that Conditions T2 and T3 of Definition 6.1 hold. Unsurprisingly, the compositionality results for quotient remain unchanged, in that the operator yields the weakest decomposition of the specification $\mathcal{R}$ when presented with the sub-component $\mathcal{P}$.

Theorem 6.28. Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components. Then there exists $\mathcal{Q}$ such that $\mathcal{P} ||_t \mathcal{Q} \sqsubseteq^t_{imp} \mathcal{R}$ iff:

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

Proof. The reasoning is unchanged from Theorem 3.25, which is to be expected given that the definition of parallel composition and refinement are largely unchanged in the timed setting. □

Theorem 6.29. Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components such that $\mathcal{Q} \sqsubseteq^t_{imp} \mathcal{P}$.

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

Proof. Follows by the previous results in this section, which should be substituted in place of the cited results occurring in the proof of Theorem 3.27. □

6.2.8 Full Abstraction

In this section, we demonstrate that our timed refinement relation precisely characterises safe substitutivity of components, meaning that a refining component cannot introduce any safety or bounded-liveness errors in a particular environment, unless the errors can be introduced by the original component in the same environment. This is achieved 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 $\equiv^{t}_{imp}$ is fully abstract for all of the operators in the specification theory.

Definition 6.30. Let $\mathcal{P}$ and $\mathcal{Q}$ be components. Then $\mathcal{Q}$ is inconsistency substitutable for $\mathcal{P}$, denoted by $\mathcal{Q} \sqsubseteq^{F,t}_{imp} \mathcal{P}$, iff for each $t \in \mathcal{T}(\mathcal{A}^{O}_{\mathcal{Q}})$ it holds that $t \in F_{\mathcal{E}(\mathcal{Q})}$ implies $t \in F_{\mathcal{E}(\mathcal{P})}$. $\diamond$

From this definition, we can show that $\sqsubseteq^{t}_{imp}$ is the weakest preorder preserving substitutivity of components by relating the relation with the testing framework.

Theorem 6.31. 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:

$$\mathcal{Q} \sqsubseteq^{t}_{imp} \mathcal{P} \text{ iff } \forall \mathcal{R} \cdot \mathcal{A}^{O}_{\mathcal{R}} = \mathcal{A}^{I}_{\mathcal{P}} \text{ and } \mathcal{A}^{I}_{\mathcal{R}} = \mathcal{A}^{O}_{\mathcal{Q}} \implies \mathcal{Q} \,||_t\, \mathcal{R} \sqsubseteq^{F,t}_{imp} \mathcal{P} \,||_t\, \mathcal{R}.$$

Proof. The only if direction follows by the monotonicity result of Theorem 6.14, given that the considitions on the interface of $\mathcal{R}$ implies that the conditions are satisfied for Theorem 6.14.

For the if direction, we show the contrapositive, so suppose that $\mathcal{Q} \not\sqsubseteq^{t}_{imp} \mathcal{P}$. Then there exists a smallest t such that $t \in T_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}})$ and $t \notin T_{\mathcal{E}(\mathcal{P})}$, or $t \in F_{\mathcal{E}(\mathcal{Q})} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}})$ and $t \notin F_{\mathcal{E}(\mathcal{P})}$. For the former, we construct a component $\mathcal{R} = \langle \mathcal{A}^{O}_{\mathcal{Q}}, \mathcal{A}^{I}_{\mathcal{P}}, T_{\mathcal{R}} \cup F_{\mathcal{R}}, F_{\mathcal{R}} \rangle$, where $T_{\mathcal{R}} = \{t' : t' \text{ is a prefix of } t\} \cdot (\mathcal{A}^{I}_{\mathcal{R}})^{*}$ and $F_{\mathcal{R}} = \{t\} \cdot \mathcal{T}(\mathcal{A}_{\mathcal{R}})$. From this, it follows that $t \in \mathcal{T}(\mathcal{A}^{O}_{\mathcal{Q}||\mathcal{R}}) \cap \mathcal{T}(\mathcal{A}^{O}_{\mathcal{P}||\mathcal{R}})$, $t \in F_{\mathcal{Q}||\mathcal{R}}$ and $t \notin T_{\mathcal{P}||\mathcal{R}}$. Consequently, $t \in F_{\mathcal{E}(\mathcal{Q}||_t\mathcal{R})}$, but $t \notin F_{\mathcal{E}(\mathcal{P}||_t\mathcal{R})}$ because $t \notin T_{\mathcal{E}(\mathcal{P})}$, Hence, $\mathcal{Q} \,||_t\, \mathcal{R} \not\sqsubseteq^{F,t}_{imp} \mathcal{P} \,||_t\, \mathcal{R}$. For the latter, we can construct the component $\mathcal{R}$, where $T_{\mathcal{R}}$ is defined as before and $F_{\mathcal{R}} = \emptyset$. $\square$

As remarked in Section 3.1.7, the conditions on the interfaces of $\mathcal{P}$ and $\mathcal{Q}$ are required for Theorem 6.31 to hold, since $\mathcal{Q} \,||_t\, \mathcal{R} \sqsubseteq^{t}_{imp} \mathcal{P} \,||_t\, \mathcal{R}$ 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 $\mathcal{A}^{I}_{\mathcal{Q}} \cap \mathcal{A}^{O}_{\mathcal{P}} = \emptyset$.

From this characterisation of $\sqsubseteq^{t}_{imp}$, it follows that $\equiv^{t}_{imp}$ is the coarsest congruence for all operators of the specification theory with respect to the inconsistency equivalence $\equiv^{F,t}_{imp}$ (i.e., $\sqsubseteq^{F,t}_{imp} \cap \sqsupseteq^{F,t}_{imp}$). Thus, $\equiv^{t}_{imp}$ is fully abstract for the operators of the theory with respect to observation of safety and bounded-liveness errors.

Corollary 6.32. Substitutive equivalence $\equiv^{t}_{imp}$ is fully abstract for parallel composition, conjunction, disjunction, hiding and quotient with respect to observational equivalence of inconsistency.

Proof. Follows from the fact that $\equiv^t_{imp}$ is a congruence for the compositional operators of the specification theory along with Theorem 6.31, the latter of which demonstrates that $\equiv^t_{imp}$ must be the coarsest such congruence. □

6.3 Non-Terminating Theory of Timed Components

In this section, we consider the subclass of timed components that are not allowed to prevent the passage of time, meaning that they are not permitted to terminate. Despite being a subclass of the components presented in the previous section, the theory is more complicated, because the definitions provided for the compositional operators in the previous section do not maintain closure. For example, the conjunction of two non-terminating components (as defined previously) may terminate. Therefore, we introduce a pruning operator on components that can remove such bad behaviours so that non-termination is guaranteed.

In keeping with the progress-sensitive trace-based theory of components (Section 3.2), the troublesome operators are conjunction and quotient. Consequently, our non-terminating theory of timed components shares similarities with the progress-sensitive framework of Section 3.2, unlike the terminating theory of timed components, which shared similarities with the substitutive (safety) framework of Section 3.1. However, our timed theory does not need to record all of the structure of a progress-sensitive component. For instance, there is no need to record the quiescent traces, since progress can now be stipulated by means of the timing constraints. Furthermore, we do not need to consider the divergent traces, because any divergent behaviour of a component must be time-convergent; such traces are excluded from our models by definition.

Definition 6.33 (Non-terminating timed component). A *non-terminating timed component* $\mathcal{P}$ is a tuple $\langle \mathcal{A}^I_\mathcal{P}, \mathcal{A}^O_\mathcal{P}, T_\mathcal{P}, F_\mathcal{P} \rangle$ in which $\mathcal{A}^I_\mathcal{P}$ and $\mathcal{A}^O_\mathcal{P}$ are disjoint sets referred to as the inputs and outputs respectively (the union of which is denoted by $\mathcal{A}_\mathcal{P}$), $T_\mathcal{P} \subseteq \mathcal{T}(\mathcal{A}_\mathcal{P})$ is a set of observable traces, and $F_\mathcal{P} \subseteq \mathcal{T}(\mathcal{A}_\mathcal{P})$ is a set of inconsistent traces. The trace sets must satisfy the constraints:

NT1. $F_\mathcal{P} \subseteq T_\mathcal{P}$

NT2. $T_\mathcal{P}$ is prefix closed

NT3. If $t \in T_\mathcal{P}$ and $t' \in (\mathcal{A}^I_\mathcal{P})^*$, then $tt' \in T_\mathcal{P}$

NT4. If $t \in F_\mathcal{P}$ and $t' \in \mathcal{T}(\mathcal{A}_\mathcal{P})$, then $tt' \in F_\mathcal{P}$

NT5. If $t \in T_\mathcal{P}$, then there exists $t' \in \mathcal{T}(\mathcal{A}^O_\mathcal{P})$ such that $tt' \in T_\mathcal{P}$ and tt' is of infinite duration.

If $0 \notin T_\mathcal{P}$, we say that $\mathcal{P}$ is *unrealisable*, and is *realisable* contrariwise. ◇

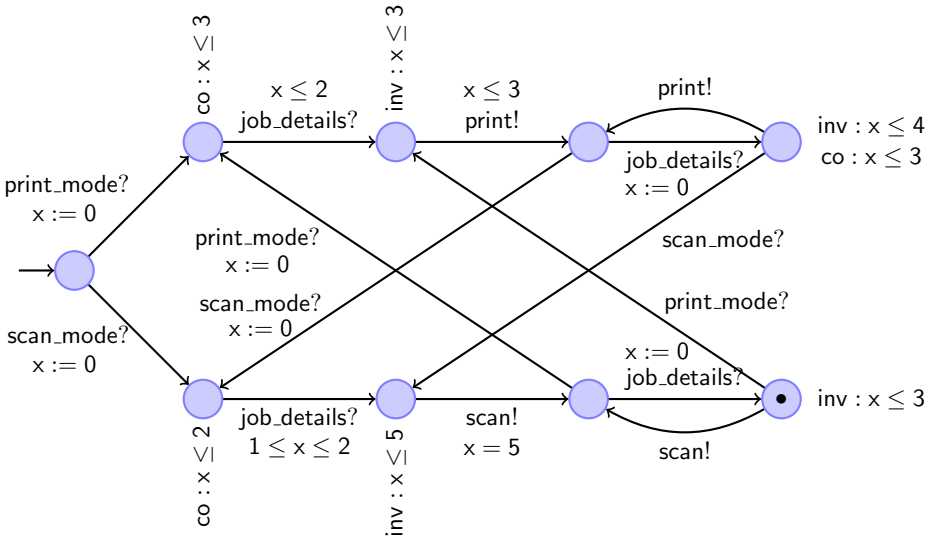

Figure 6.9: Non-terminating timed printing/scanning device

A non-terminating timed component differs from the potentially terminating variant of Section 6.2 by including the additional constraint that any observable trace can be extended by a timed trace of infinite duration over outputs (NT5). The restriction to outputs ensures that the component can always allow time to pass, regardless of how the environment behaves.

Example 6.34. The printing/scanning device in Figure 6.1 is not a well-formed non-terminating component, because it contains a range of terminating traces, such as $\langle 0, \mathsf{print\_mode}, 2, \mathsf{job\_details}, 5\rangle$ and $\langle 0, \mathsf{print\_mode}, 2, \mathsf{job\_details}, 3, \mathsf{print}, 5\rangle$, which cannot be extended by a timed trace over outputs of infinite duration. Figure 6.9 adapts the timing constraints on the printing/scanning device so that it is non-terminating (in particular, the invariants on the upper middle states and the lower right state). The printing/faxing device in Figure 6.5 is a well-formed non-terminating component, because no trace is forced to terminate. ◇

We now redefine $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ to be non-terminating timed components (henceforth referred to as 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.

6.3.1 Refinement

As usual, the refinement relation guarantees substitutivity, meaning that it preserves the absence of safety and bounded liveness errors. As in Section 6.2, the definition of refinement needs to deal with the safe representations $\mathcal{E}(\mathcal{P})$ and $\mathcal{E}(\mathcal{Q})$ of $\mathcal{P}$ and $\mathcal{Q}$, but the definition of the $\mathcal{E}$ operator should be adjusted, since the environment is no longer capable of distinguishing components by stopping time.

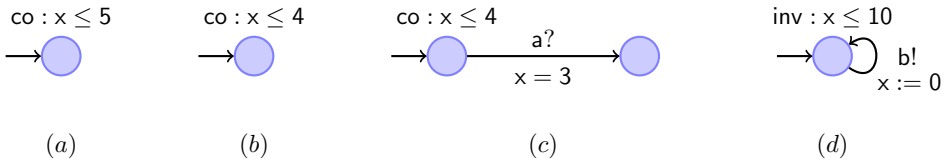

Figure 6.10: Non-terminating timed refinement

Definition 6.35. 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 $F_{\mathcal{E}(\mathcal{P})}$ is defined as the least extension-closed set containing:

$$F_\mathcal{P} \cup \{t \in T_\mathcal{P} : \exists t' \in (\mathcal{A}^O_\mathcal{P})^* \cdot tt' \in F_{\mathcal{E}(\mathcal{P})}\} \cup$$
$$\{t \in T_\mathcal{P} : \exists t' \in \mathbb{R}^\infty_0 \cdot tt' \in F_{\mathcal{E}(\mathcal{P})}, \text{ and } t'' \leq t' \text{ and } i \in \mathcal{A}^I_\mathcal{P} \implies tt''i \in F_{\mathcal{E}(\mathcal{P})}\}. \quad \diamond$$

Essentially, $F_{\mathcal{E}(\mathcal{P})}$ is the set of traces that are either inconsistent (i.e., are contained in $F_\mathcal{P}$) or from which it is inevitable that an inconsistency will be encountered. A trace will inevitably become inconsistent providing it can be extended by a timed trace over outputs to become inconsistent, and there is no input that can be issued by the environment along that path so that an inconsistency is not inevitable. This formulation ensures that the enabled interactions at any instant of time occur non-deterministically with one another, and also that interactions take precedence over the passage of time.

The definition of refinement is syntactically unchanged from the terminating theory, having redefined the $\mathcal{E}$ operator.

Definition 6.36. $\mathcal{Q}$ is said to be a *non-terminating timed refinement* of $\mathcal{P}$, written $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$, iff:

NTR1. $\mathcal{A}^I_\mathcal{P} \subseteq \mathcal{A}^I_\mathcal{Q}$

NTR2. $\mathcal{A}^O_\mathcal{Q} \subseteq \mathcal{A}^O_\mathcal{P}$

NTR3. $\mathcal{A}^I_\mathcal{Q} \cap \mathcal{A}^O_\mathcal{P} = \emptyset$

NTR4. $T_{\mathcal{E}(\mathcal{Q})} \subseteq T_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$

NTR5. $F_{\mathcal{E}(\mathcal{Q})} \subseteq F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_\mathcal{Q})$. $\diamond$

The intuition behind this definition remains unchanged from Definition 6.8. Equivalence of components, indicated using $\equiv^{nt}_{imp}$, can easily be defined by means of mutual refinement, i.e., is equal to $\sqsubseteq^{nt}_{imp} \cap (\sqsubseteq^{nt}_{imp})^{-1}$.

Example 6.37. We now consider refinements among the components shown in Figure 6.10. First note that $(a) \equiv^{nt}_{imp} (b)$, while in the terminating theory we have $(b) \sqsubseteq^{t}_{imp} (a)$, but $(a) \not\sqsubseteq^{t}_{imp} (b)$. This is because, in the terminating theory, an environment could terminate time after, say, 4.5 time units, whereas this is not possible in the non-terminating theory. Therefore, (a) and (b) will inevitably become inconsistent in the non-terminating theory, and hence should be equated.

Looking at (c), we have $(c) \sqsubseteq^{nt}_{imp} (a)$ and $(c) \sqsubseteq^{nt}_{imp} (b)$ (but not equivalence), because the environment that issues a after 3 time units will prevent the inconsistencies in (a) and (b) from being reached. Therefore, inconsistency is not inevitable in (c) from the trace $\langle 0 \rangle$.

Finally, we remark that (d) is not well-formed, because no trace of the component can be extended by a timed output word of infinite duration, due to the sole state not being Büchi accepting.

$\diamond$

Given that the definition of refinement has not changed syntactically from the terminating setting, the reflexivity and transitivity results continue to hold.

Lemma 6.38. Non-terminating timed refinement is reflexive, and is transitive subject to preservation of action types.

Proof. Unchanged from Lemma 6.11. $\square$

6.3.2 Parallel Composition

The parallel composition of two components synchronises on common actions and time, and interleaves on independent actions. Given that the outputs of the components to be composed must be disjoint (for composability) and observable traces are extendable by inputs, it follows that the composition of two non-terminating components must also be non-terminating. Consequently, the definition of parallel composition is unchanged from the terminating timed theory.

Definition 6.39. Let $\mathcal{P}$ and $\mathcal{Q}$ be composable for parallel composition (i.e., $\mathcal{A}^O_\mathcal{P} \cap \mathcal{A}^O_\mathcal{Q} = \emptyset$). The parallel composition of $\mathcal{P}$ and $\mathcal{Q}$, denoted $\mathcal{P} \mid\mid_{nt} \mathcal{Q}$, is the component $\mathcal{P} \mid\mid_t \mathcal{Q}$ as defined in Definition 6.12. $\diamond$

The following lemma demonstrates that $\mid\mid_{nt}$ yields a non-terminating timed component.

Lemma 6.40. $\mid\mid_{nt}$ preserves infinite extendability of traces.

Proof. Suppose $t \in T_{\mathcal{P}\mid\mid_{nt}\mathcal{Q}} \setminus F_{\mathcal{P}\mid\mid_{nt}\mathcal{Q}}$. Then $t \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in T_\mathcal{P}$ and $t \upharpoonright (\mathcal{A}_\mathcal{Q} \cup \mathbb{R}^\infty_0) \in T_\mathcal{Q}$. Now there exists $t_p \in \mathcal{T}(\mathcal{A}^O_\mathcal{P})$ and $t_q \in \mathcal{T}(\mathcal{A}^O_\mathcal{Q})$, both of infinite duration, such that $tt_p \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in T_\mathcal{P}$ and $tt_q \upharpoonright (\mathcal{A}_\mathcal{Q} \cup \mathbb{R}^\infty_0) \in T_\mathcal{Q}$. Thus we can construct $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}\mid\mid\mathcal{Q}})$ such that $t' \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) = t_p$ and $t' \upharpoonright (\mathcal{A}_\mathcal{Q} \cup \mathbb{R}^\infty_0) = t_q$. It is necessarily the case that t' is of infinite duration, and moreover, $tt' \in T_{\mathcal{P}\mid\mid_{nt}\mathcal{Q}}$ as required. $\square$

Theorem 6.41. Let $\mathcal{P}$, $\mathcal{P}'$, $\mathcal{Q}$ and $\mathcal{Q}'$ be components such that $\mathcal{P}$ and $\mathcal{Q}$ are composable, $\mathcal{A}_{\mathcal{P}'} \cap \mathcal{A}_{\mathcal{Q}'} \cap \mathcal{A}_{\mathcal{P}\mid\mid\mathcal{Q}} \subseteq \mathcal{A}_\mathcal{P} \cap \mathcal{A}_\mathcal{Q}$ and $\mathcal{A}^I_{\mathcal{P}'\mid\mid\mathcal{Q}'} \cap \mathcal{A}^O_{\mathcal{P}\mid\mid\mathcal{Q}} = \emptyset$. If $\mathcal{P}' \sqsubseteq^{nt}_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^{nt}_{imp} \mathcal{Q}$, then $\mathcal{P}' \mid\mid_{nt} \mathcal{Q}' \sqsubseteq^{nt}_{imp} \mathcal{P} \mid\mid_{nt} \mathcal{Q}$.

Proof. Despite the fact that $\mid\mid_{nt}$ equals $\mid\mid_t$, the proof of monotonicity for parallel composition does not follow immediately from Theorem 6.14, because the definition of the $\mathcal{E}$ operator has

changed. Therefore, we must show that $F_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')} \subseteq F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'})$ and $T_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')} \subseteq T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'})$. Begin by supposing that $t \in T_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')}$ and $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{P}||\mathcal{Q}})$. Then, by the same reasoning as in Theorem 3.10, it follows that $t \in T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'}$. Now suppose that $t \in F_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}||\mathcal{Q}})$. We demonstrate that $X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}||\mathcal{Q}}) \subseteq F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$ for each $i \in \mathbb{N}$, where X_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')}$ defined as a fixed point. If $i = 0$, then the result holds trivially, because $X_i = \emptyset$. If instead $t \in X_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}||\mathcal{Q}})$, then either (i) $t \in F_{\mathcal{P}'||_{nt}\mathcal{Q}'}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}'||\mathcal{Q}'})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$ and for every prefix t'' of t' and $a \in \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'}$ it holds that $tt''a \in X_k$. For (i), it holds without loss of generality that $t \upharpoonright (\mathcal{A}_{\mathcal{P}'} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{P}'}$ and $t \upharpoonright (\mathcal{A}_{\mathcal{Q}'} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{Q}'}$. By the same reasoning as in Theorem 3.10, it follows that $t \upharpoonright (\mathcal{A}_{\mathcal{P}'} \cup \mathbb{R}^\infty_0) = t \upharpoonright (\mathcal{A}_{\mathcal{P}} \cup \mathbb{R}^\infty_0)$ and $t \upharpoonright (\mathcal{A}_{\mathcal{Q}'} \cup \mathbb{R}^\infty_0) = t \upharpoonright (\mathcal{A}_{\mathcal{Q}} \cup \mathbb{R}^\infty_0)$. Therefore, by $\mathcal{P}' \sqsubseteq^{nt}_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^{nt}_{imp} \mathcal{Q}$ we have $t \upharpoonright (\mathcal{A}_{\mathcal{P}} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{E}(\mathcal{P})}$ and $t \upharpoonright (\mathcal{A}_{\mathcal{Q}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{E}(\mathcal{Q})}$, from which it can be inferred that $t \in F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$. For (ii), by the induction hypothesis it follows $tt' \in F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$, and so $t \in F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$ by the definition of the $\mathcal{E}$ operator. For (iii), by the induction hypothesis we obtain $tt' \in F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$, and $tt''a \in F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$ for each $a \in \mathcal{A}^I_{\mathcal{P}||\mathcal{Q}}$ given that $tt'a \in \mathcal{T}(\mathcal{A}_{\mathcal{P}||\mathcal{Q}})$. Since $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} \subseteq \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'}$, it follows by the definition of the $\mathcal{E}$ operator that $t \in F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$ as required. The result therefore holds for all $i \in \mathbb{N}$, hence $F_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')} \subseteq F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'})$. Now to show that $T_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')} \subseteq T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}'||\mathcal{Q}'})$, suppose that $t \in (T_{\mathcal{P}'||_{nt}\mathcal{Q}'} \setminus F_{\mathcal{E}(\mathcal{P}'||_{nt}\mathcal{Q}')}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}||\mathcal{Q}})$. Then by the same reasoning as in Theorem 6.14, it follows that $t \in T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$, given that $t \upharpoonright (\mathcal{A}_{\mathcal{P}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{E}(\mathcal{P})}$ and $t \upharpoonright (\mathcal{A}_{\mathcal{Q}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{E}(\mathcal{Q})}$ implies $t \in T_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$. □

6.3.3 Conjunction

The definition of conjunction supplied in Definition 6.16 is not adequate for the non-terminating theory, because the conjunction is required to constrain its output behaviour to common behaviours of both components. To see why this is problematic, suppose that t is a trace of finite duration that is common to both $\mathcal{P}$ and $\mathcal{Q}$, while $t_p \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}})$ is the only trace of infinite duration such that $tt_p \in T_{\mathcal{P}}$, and $t_q \in \mathcal{T}(\mathcal{A}^O_{\mathcal{Q}})$ is the only trace of infinite duration such that $tt_q \in T_{\mathcal{Q}}$. If $t_p \neq t_q$, then there is no $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}})$ of infinite duration such that $tt' \in T_{\mathcal{P}\wedge\mathcal{Q}}$. As a result, we must remove all subsequent behaviour from (and including) the last observed output on the trace t. Such a pruning should be applied repeatedly to the conjunction construction supplied in Definition 6.16 until all remaining traces can be extended by a timed output trace of infinite duration. Naturally, this means that the conjunction of two realisable components may not be realisable.

Definition 6.42. Let $\mathcal{P}$ and $\mathcal{Q}$ be composable for conjunction (i.e., $\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}$ is disjoint from $\mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}}$). Then $\mathcal{P} \wedge_{nt} \mathcal{Q}$ is the component $\langle \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}, \mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}}, T_{\mathcal{P}\wedge_t\mathcal{Q}} \setminus Err, F_{\mathcal{P}\wedge_t\mathcal{Q}} \setminus Err, \rangle$, where Err is the smallest set containing:

$$\{t \in T_{\mathcal{P}\wedge_t\mathcal{Q}} : \exists t' \in (\mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})^* \cdot \nexists t'' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}}) \cdot tt't'' \in T_{\mathcal{P}\wedge_t\mathcal{Q}} \setminus Err \text{ and } tt't'' \text{ is of infinite duration}\}.$$

Note that, in our reference to $\mathcal{P} \wedge_t \mathcal{Q}$, we assume that the $\mathcal{E}$ operator corresponds to the one provided in Definition 6.35. ◇

It is obvious by the definition of the Err set that $\mathcal{P} \wedge_{nt} \mathcal{Q}$ is a non-terminating component, since all traces that can lead to termination are pruned. Despite this pruning, the compositionality results continue to hold, as the next theorem shows.

Theorem 6.43. Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$ be components composable for conjunction. Then:

- $\mathcal{P} \wedge_{nt} \mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$ and $\mathcal{P} \wedge_{nt} \mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{Q}$
- $\mathcal{R} \sqsubseteq^{nt}_{imp} \mathcal{P}$ and $\mathcal{R} \sqsubseteq^{nt}_{imp} \mathcal{Q}$ implies $\mathcal{R} \sqsubseteq^{nt}_{imp} \mathcal{P} \wedge_{nt} \mathcal{Q}$
- $\mathcal{P}' \sqsubseteq^{nt}_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^{nt}_{imp} \mathcal{Q}$ implies $\mathcal{P}' \wedge_{nt} \mathcal{Q}' \sqsubseteq^{nt}_{imp} \mathcal{P} \wedge_{nt} \mathcal{Q}$.

Proof. In order to show that $\mathcal{P} \wedge_{nt} \mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$, we need to show that $F_{\mathcal{E}(\mathcal{P}\wedge_{nt}\mathcal{Q})} \subseteq F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})$ and $T_{\mathcal{E}(\mathcal{P}\wedge_{nt}\mathcal{Q})} \subseteq T_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})$. First suppose $t \in T_{\mathcal{E}(\mathcal{P}\wedge_{nt}\mathcal{Q})}$, $t \notin \mathcal{T}(\mathcal{A}_\mathcal{P})$ and assume that the result holds for all strict prefixes. Then there is a prefix $t'i$ of t such that $t' \in \mathcal{T}(\mathcal{A}_\mathcal{P})$ and $i \in \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}} \setminus \mathcal{A}_\mathcal{P}$. Thus, $t' \in T_{\mathcal{E}(\mathcal{P})}$ and $t'i \in T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}$. Hence $t \in T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}$. Now assume that $t \in F_{\mathcal{E}(\mathcal{P}\wedge_{nt}\mathcal{Q})} \cap \mathcal{T}(\mathcal{A}_\mathcal{P})$. We show that $t \in X_i \cap \mathcal{T}(\mathcal{A}_\mathcal{P})$ implies $t \in F_{\mathcal{E}(\mathcal{P})}$ for each $i \in \mathbb{N}$, where X_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{P}\wedge_{nt}\mathcal{Q})}$. In the case that $i = 0$, the result hold trivially as $X_i = \emptyset$. Now suppose that $t \in X_{k+1}$. Then (i) $t \in F_{\mathcal{P}\wedge_t\mathcal{Q}} \setminus Err$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}\wedge\mathcal{Q}})^*$ such that $tt' \in X_k$, or (iii) $t \in T_{\mathcal{P}\wedge_t\mathcal{Q}} \setminus Err$ and there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$, and for all t'' a prefix of t' and $a \in \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}$ it holds that $tt''a \in X_k$. For (i), it holds that $t \in F_{\mathcal{E}(\mathcal{P})}$ by the definition of $\wedge_{nt}$. For (ii), it holds that $tt' \in X_k \cap \mathcal{T}(\mathcal{A}_\mathcal{P})$, so by the induction hypothesis we have $tt' \in F_{\mathcal{E}(\mathcal{P})}$, yielding $t \in F_{\mathcal{E}(\mathcal{P})}$ by the definition of the $\mathcal{E}$ operator. Similarly for (iii), by the induction hypothesis it follows that $tt' \in F_{\mathcal{E}(\mathcal{P})}$ and $tt''a \in F_{\mathcal{E}(\mathcal{P})}$ when $a \in \mathcal{A}^I_\mathcal{P}$. As $\mathcal{A}^I_\mathcal{P} \subseteq \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}}$, it holds that $t \in F_{\mathcal{E}(\mathcal{P})}$ by the definition of the $\mathcal{E}$ operator. Thus, $F_{\mathcal{E}(\mathcal{P}\wedge_{nt}\mathcal{Q})} \subseteq F_{\mathcal{E}(\mathcal{P})} \cup (T_{\mathcal{E}(\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{P}\wedge\mathcal{Q}})$. For the remaining case of $t \in (T_{\mathcal{P}\wedge_{nt}\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}\wedge_{nt}\mathcal{Q})}) \cap \mathcal{T}(\mathcal{A}_\mathcal{P})$, it follows that $t \in T_{\mathcal{P}\wedge_t\mathcal{Q}}$, from which we see that $t \in T_{\mathcal{E}(\mathcal{P})}$, as required.

For the second claim, it is straightforward to show that $F_{\mathcal{E}(\mathcal{R})} \subseteq F_{\mathcal{P}\wedge_t\mathcal{Q}} \cup (T_{\mathcal{P}\wedge_t\mathcal{Q}} \uparrow \mathcal{A}^I_\mathcal{R})$ and $T_{\mathcal{E}(\mathcal{R})} \subseteq T_{\mathcal{P}\wedge_t\mathcal{Q}} \cup (T_{\mathcal{P}\wedge_t\mathcal{Q}} \uparrow \mathcal{A}^I_\mathcal{R})$ by the proof for Theorem 6.18 (based on Theorem 3.14), since the $\mathcal{E}$ operator is only used syntactically. Note that the $\mathcal{E}$ operator does not need to occur on the right hand side of the inclusions. It is therefore sufficient to show that $T_{\mathcal{E}(\mathcal{R})} \cap Err = \emptyset$, from which it follows that $F_{\mathcal{E}(\mathcal{R})} \subseteq F_{\mathcal{P}\wedge_{nt}\mathcal{Q}} \cup (T_{\mathcal{P}\wedge_{nt}\mathcal{Q}} \uparrow \mathcal{A}^I_\mathcal{R})$ and $T_{\mathcal{E}(\mathcal{R})} \subseteq T_{\mathcal{P}\wedge_{nt}\mathcal{Q}} \cup (T_{\mathcal{P}\wedge_{nt}\mathcal{Q}} \uparrow \mathcal{A}^I_\mathcal{R})$. So let X_i be the i-th approximation of Err defined as a fixed point. Then we show that $X_i \cap T_{\mathcal{E}(\mathcal{R})} = \emptyset$ for each $i \in \mathbb{N}$. When $i = 0$ the result holds trivially because $X_i = \emptyset$. So

suppose that $t \in X_{k+1} \cap T_{\mathcal{E}(\mathcal{R})}$. Then $t \in T_{\mathcal{P} \wedge_t \mathcal{Q}}$ and there exists $t' \in (\mathcal{A}^I_{\mathcal{P} \wedge \mathcal{Q}})^*$ for which there is no $t'' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P} \wedge \mathcal{Q}})$ such that $tt't'' \in T_{\mathcal{P} \wedge_t \mathcal{Q}} \setminus X_k$ and $tt't''$ is of infinite duration. Note that $t' \in (\mathcal{A}^I_{\mathcal{R}})^*$, so $tt' \in T_{\mathcal{E}(\mathcal{R})}$. As $\mathcal{R}$ is a non-terminating component, it follows that there is a trace $t_r \in \mathcal{T}(\mathcal{A}^O_{\mathcal{R}})$ such that $tt't_r \in T_{\mathcal{E}(\mathcal{R})}$ and $tt't_r$ is of infinite duration. But $t_r \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P} \wedge \mathcal{Q}})$, and moreover, $tt't_r \in T_{\mathcal{P} \wedge_t \mathcal{Q}}$. Consequently, $tt't_r \in X_k$, but this is contradictory. Therefore, $t \notin Err$ as required, and so $T_{\mathcal{E}(\mathcal{R})} \cap Err = \emptyset$.

The third claim follows by the same reasoning as in Theorem 6.18, which is based on the proof of Theorem 3.14. However, we need to show that $\mathcal{P} \wedge_{nt} \mathcal{Q} = \mathcal{P}'' \wedge_{nt} \mathcal{Q}''$, where $\mathcal{P}''$ and $\mathcal{Q}''$ are obtained from $\mathcal{P}$ and $\mathcal{Q}$ by removing any trace containing actions in $(\mathcal{A}^O_{\mathcal{P}} \setminus \mathcal{A}^O_{\mathcal{Q}}) \cup (\mathcal{A}^O_{\mathcal{Q}} \setminus \mathcal{A}^O_{\mathcal{P}})$. Clearly $\mathcal{P} \wedge_t \mathcal{Q} = \mathcal{P}'' \wedge_t \mathcal{Q}''$, so it is sufficient to show that $Err_{\mathcal{P} \wedge_{nt} \mathcal{Q}} = Err_{\mathcal{P}'' \wedge_{nt} \mathcal{Q}''}$. This can be demonstrated by considering the fixed-point approximations of the Err sets. □

Example 6.44. Under the assumption that the interfaces of the non-terminating print/scan device (Figure 6.9) and the print/fax device in which fax_mode has been hidden (Figure 6.8) consist of the actions in the respective diagrams, the conjunction, prior to removing the Err traces, is as shown in Figure 6.11, but having removed the scan transitions. However, $\langle 0, \mathsf{scan\_mode}, 1, \mathsf{job\_details}, 1 \rangle \in Err$ because the trace is terminating, so $\langle 0 \rangle \in Err$, meaning that the conjunction is undefined.

When the components to be conjoined are assumed to have the same interfaces that include the union of their actions, the conjunction is precisely as depicted in Figure 6.11. No pruning of Err traces is required, because all of the original traces defined according to $\wedge_t$ are non-terminating. ◇

6.3.4 Disjunction

As the disjunction $\mathcal{P} \vee_t \mathcal{Q}$ contains all of the timed traces in $\mathcal{P}$ or $\mathcal{Q}$ whose actions lie in $\mathcal{A}_{\mathcal{P} \vee \mathcal{Q}}$, it follows that the disjunction of two non-terminating components must itself be non-terminating, because $\mathcal{A}^O_{\mathcal{P} \vee \mathcal{Q}} = \mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}}$. Therefore, the definition of disjunction in the non-terminating setting should match Definition 6.20, which is applicable to components that can terminate.

Definition 6.45. Let $\mathcal{P}$ and $\mathcal{Q}$ be composable for disjunction (i.e., $\mathcal{A}^I_{\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{Q}}$ is disjoint from $\mathcal{A}^O_{\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{Q}}$). Then the disjunction of $\mathcal{P}$ and $\mathcal{Q}$, denoted $\mathcal{P} \vee_{nt} \mathcal{Q}$, is the component $\mathcal{P} \vee_t \mathcal{Q}$ as defined in Definition 6.20. ◇

To see that $\mathcal{P} \vee_{nt} \mathcal{Q}$ preserves non-termination, suppose that $t \in T_{\mathcal{P} \vee_{nt} \mathcal{Q}} \setminus F_{\mathcal{P} \vee_{nt} \mathcal{Q}}$. Then, without loss of generality, it follows that $t \in T_{\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P} \vee \mathcal{Q}})$. Since $\mathcal{P}$ is non-terminating, there is $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}})$ such that $tt' \in T_{\mathcal{P}}$ and tt' is of infinite duration. Since $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{P} \vee \mathcal{Q}}$, it follows that $tt' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P} \vee \mathcal{Q}})$ as required, and so $tt' \in T_{\mathcal{P} \vee_{nt} \mathcal{Q}}$. $F_{\mathcal{P} \vee_{nt} \mathcal{Q}}$ is automatically closed under all extensions.

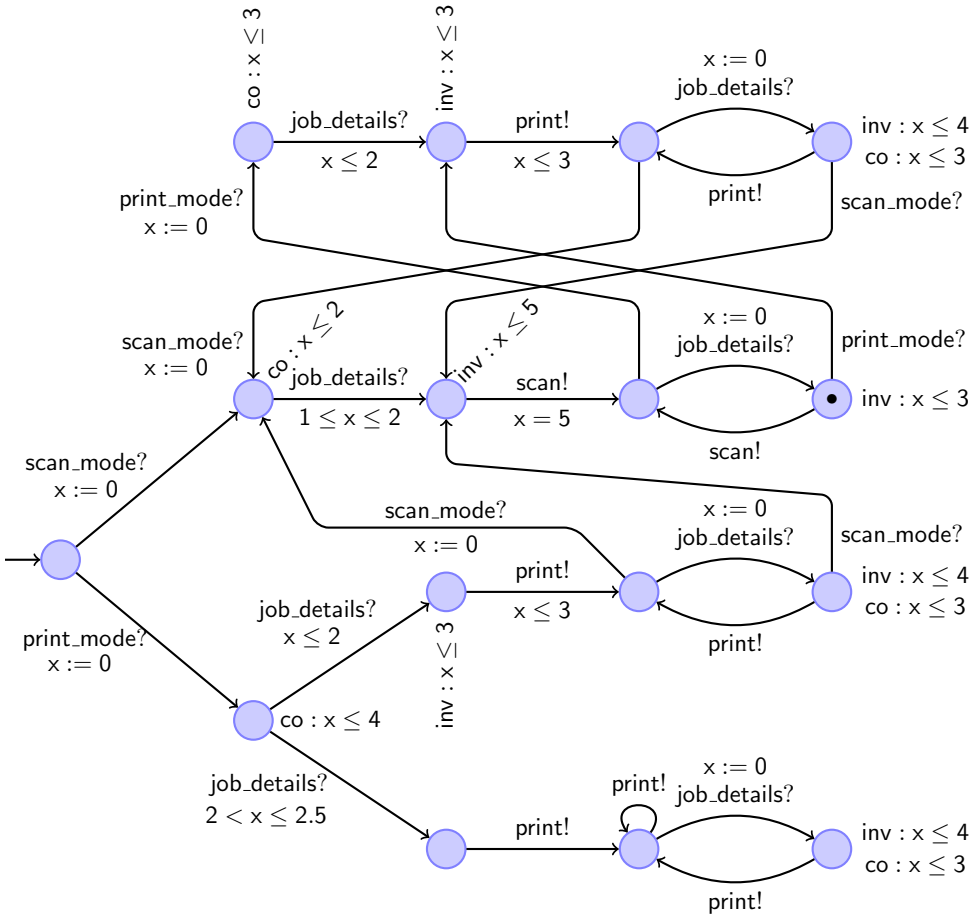

Figure 6.11: Non-terminating timed conjunction of the printing/scanning and printing/faxing devices

As the definition of disjunction on non-terminating components matches that for the terminating theory, it follows that the operator continues to be the join operator for the refinement preorder.

Theorem 6.46. Let $\mathcal{P}$ and $\mathcal{Q}$, and $\mathcal{P}'$ and $\mathcal{Q}'$ be components composable for disjunction. Then:

- $\mathcal{P} \sqsubseteq^{nt}_{imp} \mathcal{P} \vee_{nt} \mathcal{Q}$ and $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P} \vee_{nt} \mathcal{Q}$
- $\mathcal{P} \sqsubseteq^{nt}_{imp} \mathcal{R}$ and $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{R}$ implies $\mathcal{P} \vee_{nt} \mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{R}$
- $\mathcal{P}' \sqsubseteq^{nt}_{imp} \mathcal{P}$ and $\mathcal{Q}' \sqsubseteq^{nt}_{imp} \mathcal{Q}$ implies $\mathcal{P}' \vee_{nt} \mathcal{Q}' \sqsubseteq^{nt}_{imp} \mathcal{P} \vee_{nt} \mathcal{Q}$.

Proof. The results do not follow immediately from Theorem 6.22, which is itself based on Theorem 3.18, because the original proof needs to make use of the actual definition for the $\mathcal{E}$ operator, rather than using it syntactically.

For the first claim, we show that $\mathcal{P} \sqsubseteq^{nt}_{imp} \mathcal{P} \vee_{nt} \mathcal{Q}$ by demonstrating $F_{\mathcal{E}(\mathcal{P})} \subseteq F_{\mathcal{E}(\mathcal{P} \vee_{nt} \mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P} \vee_{nt} \mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}})$ and $T_{\mathcal{E}(\mathcal{P})} \subseteq T_{\mathcal{E}(\mathcal{P} \vee_{nt} \mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P} \vee_{nt} \mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}})$. First suppose $t \in T_{\mathcal{E}(\mathcal{P})}$, $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{P} \vee \mathcal{Q}})$ and the result holds for all strict prefixes. Then there exists t' a prefix of t and $i \in \mathcal{A}^I_{\mathcal{P}}$

such that $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$ and $i \in \mathcal{A}^I_{\mathcal{P}} \setminus \mathcal{A}_{\mathcal{P}\vee\mathcal{Q}}$. Thus, $t' \in T_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ and $t'i \in T_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}}$. Hence $t \in T_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}}$ as required. Now suppose that $t \in F_{\mathcal{E}(\mathcal{P})} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$. We show that $t \in X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$ implies $t \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ for each $i \in \mathbb{N}$, where X_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{P})}$. The result holds trivially when $i = 0$, because $X_i = \emptyset$. For the inductive case of $t \in X_{k+1}$, either (i) $t \in F_{\mathcal{P}}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$, and for every prefix t'' of t' and $a \in \mathcal{A}^I_{\mathcal{P}}$ it holds that $tt''a \in X_k$. For (i), it obviously holds that $t \in F_{\mathcal{P}\vee_{nt}\mathcal{Q}}$ by the definition of $\vee_{nt}$, and so $t \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$. For (ii), by the induction hypothesis it holds that $tt' \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ since $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}}$, implying $tt' \in \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$. Hence $t \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ by the definition of the $\mathcal{E}$ operator. For (iii), by the induction hypothesis it holds that $tt' \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ and $tt''a \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ for each $a \in \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}}$. Since $\mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}} \subseteq \mathcal{A}^I_{\mathcal{P}}$, it holds by the definition of the $\mathcal{E}$ operator that $t \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$. Hence $F_{\mathcal{E}(\mathcal{P})} \subseteq F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \cup (T_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{P}})$. To show T-set containment, suppose that $t \in (T_{\mathcal{P}} \setminus F_{\mathcal{E}(\mathcal{P})}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}\vee\mathcal{Q}})$. Then from the definition of $\vee_{nt}$ it holds that $t \in T_{\mathcal{P}\vee_{nt}\mathcal{Q}}$, and so $t \in T_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ as required. Showing $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P} \vee_{nt} \mathcal{Q}$ is similar.

For the second claim, we need to show that $F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \subseteq F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}})$ and $T_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \subseteq T_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}})$. When $t \in T_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$ and $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{R}})$, it follows by the same reasoning as in the previous case that $t \in T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}}$. Now suppose $t \in F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}})$. We show that $t \in X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}})$ implies $t \in F_{\mathcal{E}(\mathcal{R})}$ by induction on $i \in \mathbb{N}$, where X_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}$. The base case of $i = 0$ is trivial, so suppose $t \in X_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}})$. Then either (i) $t \in F_{\mathcal{P}\vee_{nt}\mathcal{Q}}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$ and for every prefix t'' of t' and $a \in \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}}$ it holds that $tt''a \in X_k$. For (i), there exists a prefix t_f of t such that, without loss of generality, it holds that $t_f \in F_{\mathcal{P}}$. Therefore, from $\mathcal{P} \sqsubseteq^{nt}_{imp} \mathcal{R}$, it follows that $t_f \in F_{\mathcal{E}(\mathcal{R})}$, which given that $t \in \mathcal{T}(\mathcal{A}_{\mathcal{R}})$, implies $t \in F_{\mathcal{E}(\mathcal{R})}$. For (ii), by the same reasoning as in the first claim, it follows that $t \in F_{\mathcal{E}(\mathcal{R})}$ by the induction hypothesis, given that $\mathcal{A}^O_{\mathcal{P}\vee\mathcal{Q}} \subseteq \mathcal{A}^O_{\mathcal{R}}$. Similarly for (iii), by the reasoning in the previous claim, it holds that $t \in F_{\mathcal{E}(\mathcal{R})}$, since $\mathcal{A}^I_{\mathcal{R}} \subseteq \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}}$. Therefore, $F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})} \subseteq F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\vee\mathcal{Q}})$. Now suppose that $t \in (T_{\mathcal{P}\vee_{nt}\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}\vee_{nt}\mathcal{Q})}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}})$. Then by the definition of $T_{\mathcal{P}\vee_{nt}\mathcal{Q}}$, it holds without loss of generality that $t \in T_{\mathcal{P}}$. From $\mathcal{P} \sqsubseteq^{nt}_{imp} \mathcal{R}$, we derive $t \in T_{\mathcal{E}(\mathcal{R})}$, given that $t \in \mathcal{T}(\mathcal{A}_{\mathcal{R}})$, as required.

The third claim follows from Theorem 6.22 because the proof, which is based on Theorem 3.18, would make use of the previous two claims without any reference to the $\mathcal{E}$ operator. □

6.3.5 Hiding

Hiding of an action cannot make a non-terminating component terminate, so the definition of the operator in the non-terminating theory should match that provided in Definition 6.24 for the framework with termination. In fact, hiding has the potential to make a terminating component

non-terminating.

Definition 6.47. Let $\mathcal{P}$ be a component and let b be an action. Then the hiding of b in $\mathcal{P}$, denoted $\mathcal{P} /_{nt} b$, is the component $\mathcal{P} /_t b$ as defined in Definition 6.24. $\diamond$

To see why the hiding operator preserves non-termination, suppose $t \in T_{\mathcal{P}/_{nt}b}$. If $b \in \mathcal{A}^I_{\mathcal{P}}$ or $b \notin \mathcal{A}_{\mathcal{P}}$, then $t \in T_{\mathcal{P}}$. So there exists $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}})$ such that $tt' \in T_{\mathcal{P}}$ and tt' has infinite duration. But then $tt' \in \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$, so $tt' \in T_{\mathcal{P}/_{nt}b}$. If instead $b \in \mathcal{A}^O_{\mathcal{P}}$, then there exists $t' \in T_{\mathcal{P}}$ such that $t' \upharpoonright (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) = t$. As $\mathcal{P}$ is non-terminating, there is $t'' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}})$ such that $t't'' \in T_{\mathcal{P}}$ and $t't''$ is of infinite duration. Therefore, $t't'' \upharpoonright (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{P}/_{nt}b}$. By the way projection is defined, it follows that $t't'' \upharpoonright (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0)$ is of infinite duration. Therefore, we can take $t''' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}/b})$ such that $t''' = t'' \upharpoonright (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0)$. Hence $tt''' \in T_{\mathcal{P}/_{nt}b}$ as required.

We now show that the hiding operator is compositional under refinement.

Theorem 6.48. Let $\mathcal{P}$ and $\mathcal{Q}$ be components and let b be an action such that $b \notin \mathcal{A}^O_{\mathcal{P}}$. If $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$, then $\mathcal{Q} /_{nt} b \sqsubseteq^{nt}_{imp} \mathcal{P} /_{nt} b$.

Proof. We begin by showing that $F_{\mathcal{E}(\mathcal{Q}/_{nt}b)} \subseteq F_{\mathcal{E}(\mathcal{P}/_{nt}b)} \cup (T_{\mathcal{E}(\mathcal{P}/_{nt}b)} \uparrow \mathcal{A}^I_{\mathcal{Q}/b})$ and $T_{\mathcal{E}(\mathcal{Q}/_{nt}b)} \subseteq T_{\mathcal{E}(\mathcal{P}/_{nt}b)} \cup (T_{\mathcal{E}(\mathcal{P}/_{nt}b)} \uparrow \mathcal{A}^I_{\mathcal{Q}/b})$. First suppose that $t \in T_{\mathcal{E}(\mathcal{Q}/_{nt}b)}$ and $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$, and assume that the containment holds for all strict prefixes. Then there is a prefix $t'a$ of t with $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$ and $a \in \mathcal{A}^I_{\mathcal{Q}/b} \setminus \mathcal{A}_{\mathcal{P}/b}$, for which one can conclude that $t' \in T_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ and $t'a \in T_{\mathcal{E}(\mathcal{P}/_{nt}b)} \uparrow \mathcal{A}^I_{\mathcal{Q}/b}$. Consequently, $t \in T_{\mathcal{E}(\mathcal{P}/_{nt}b)} \uparrow \mathcal{A}^I_{\mathcal{Q}/b}$ as required. Now suppose that $t \in F_{\mathcal{E}(\mathcal{Q}/_{nt}b)} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$. We show that $Y_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b}) \subseteq F_{\mathcal{E}(\mathcal{P})/_{nt}b} \subseteq F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$, the latter containment being demonstrated further below, where Y_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{Q}/_{nt}b)}$. When $i = 0$, the result holds trivially because $Y_0 = \emptyset$, so suppose that $t \in Y_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$. Then either (i) $t \in F_{\mathcal{Q}/_{nt}b}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{Q}/b})^*$ such that $tt' \in Y_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in Y_k$, and for all prefixes t'' of t' and $a \in \mathcal{A}^I_{\mathcal{Q}/b}$, it holds that $tt''a \in Y_k$. For (i), begin by supposing that $b \notin \mathcal{A}^O_{\mathcal{P}}$. Then $t \in F_{\mathcal{Q}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{Q}/b})$, so from $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$ we derive $t \in F_{\mathcal{E}(\mathcal{P})}$, as $t \in \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$ by initial assumption. Therefore, $t \in F_{\mathcal{E}(\mathcal{P})/_{nt}b}$. When $b \in \mathcal{A}^O_{\mathcal{P}}$, there exists $t' \in F_{\mathcal{Q}}$ such that $t' \upharpoonright \mathcal{A}_{\mathcal{Q}/b} = t$. Consequently $t' \in F_{\mathcal{E}(\mathcal{P})}$, given that $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{P}})$, itself due to $t \in \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$. From this, it follows that $t \in F_{\mathcal{E}(\mathcal{P})/_{nt}b}$, since $t' \upharpoonright \mathcal{A}_{\mathcal{P}/b} = t$. In the subsequent paragraphs we show that $F_{\mathcal{E}(\mathcal{P})/_{nt}b} \subseteq F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$. For (ii), note that $t' \in (\mathcal{A}^O_{\mathcal{P}/b})^*$, so $tt' \in Y_k \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$, which by the induction hypothesis yields $tt' \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ and so $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ by the definition of the $\mathcal{E}$ operator. For (iii), by the induction hypothesis we derive $tt' \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ and $tt''a \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ when $a \in \mathcal{A}^I_{\mathcal{P}/b}$, given $\mathcal{A}^I_{\mathcal{P}/b} \subseteq \mathcal{A}^I_{\mathcal{Q}/b}$. Hence, by the definition of the $\mathcal{E}$ operator, it holds that $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$.

We now need to show that $F_{\mathcal{E}(\mathcal{P})/_{nt}b} \subseteq F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ (for case (i) previously), by demonstrating that $X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b}) \subseteq F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ when $b \notin \mathcal{A}^O_{\mathcal{P}}$ and $X_i \upharpoonright (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \subseteq F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ when $b \in \mathcal{A}^O_{\mathcal{P}}$, where X_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{P})}$. The base case is trivial, since $X_0 = \emptyset$, so we only consider the inductive cases below.

First suppose that $t \in X_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$ and $b \notin \mathcal{A}^O_{\mathcal{P}}$. Then either (i) $t \in F_{\mathcal{P}}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}})^*$ such that $tt' \in X_k$, or (iii) $t \in T_{\mathcal{P}}$ and there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$ and for each prefix t'' and $a \in \mathcal{A}^I_{\mathcal{P}}$ it holds that $tt''a \in X_k$. For (i), it follows that $t \in F_{\mathcal{P}/_{nt}b}$ and so $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$. For (ii), it holds that $t'' \in (\mathcal{A}^O_{\mathcal{P}/b})^*$, so $tt' \in X_k \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$. By the induction hypothesis, $tt' \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ and so $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ by the definition of the $\mathcal{E}$ operator. For (iii), $tt' \in X_k \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}/b})$ so by the induction hypothesis we derive $tt' \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$. Moreover, $tt''a \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ for each $a \in \mathcal{A}_{\mathcal{P}/b}$, given that $\mathcal{A}^I_{\mathcal{P}/b} \subseteq \mathcal{A}^I_{\mathcal{P}}$, which means $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ by the definition of the $\mathcal{E}$ operator.

Now suppose that $t \in X_{k+1} \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0)$ and $b \in \mathcal{A}^O_{\mathcal{P}}$. Then there exists $t_p \in X_{k+1}$ such that $t_p \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) = t$. So either (i) $t_p \in F_{\mathcal{P}}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}})^*$ such that $t_pt' \in X_k$, or (iii) $t_p \in T_{\mathcal{P}}$ and there exists $t' \in \mathbb{R}^\infty_0$ such that $t_pt' \in X_k$, and for each prefix t'' of t' and $a \in \mathcal{A}^I_{\mathcal{P}}$ it holds that $t_pt''a \in X_k$. For (i), it follows that $t \in F_{\mathcal{P}/_{nt}b}$ and so $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$. For (ii), it follows that $t_pt' \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \in X_k \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0)$, and so, by the induction hypothesis, $t_pt' \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$. As $t' \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \in (\mathcal{A}^O_{\mathcal{P}/b})^*$, it follows that $t_p \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ and so $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$. For (iii), by the induction hypothesis we have $t_pt' \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ and $t_pt''a \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$. As $t''a = t''a \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0)$ and $t' = t' \restriction (\mathcal{A}_{\mathcal{P}/b} \cup \mathbb{R}^\infty_0)$, it follows that $t \in F_{\mathcal{E}(\mathcal{P}/_{nt}b)}$ by the definition of the $\mathcal{E}$ operator.

The T-set containments are unchanged from Theorem 6.25, since it is necessary to only consider the traces in $(T_{\mathcal{Q}/b} \setminus F_{\mathcal{E}(\mathcal{Q}/b)})$, which do not involve the $\mathcal{E}$ operator. □

6.3.6 Quotient

The definition of quotient makes use of the construction in Definition 6.27 for the theory permitting termination; however, this may yield a component that can terminate, even when both $\mathcal{P}$ and $\mathcal{R}$ are non-terminating. It is necessary, therefore, to apply a pruning operation to the construction $\mathcal{R} \;/_t\; \mathcal{P}$ that removes all traces from which the component can terminate under its own control. Note that the $/_t$ operator has the same syntactic definition as in Section 6.2.7, but it does not yield the same component as in that section, because the definition of the $\mathcal{E}$ operator has changed.

Definition 6.49. 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 $\mathcal{R} \;/_{nt}\; \mathcal{P}$ with signature $\langle \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}, \mathcal{A}^O_{\mathcal{R}/\mathcal{P}}, T_{\mathcal{R}/_t\mathcal{P}} \setminus Err, F_{\mathcal{R}/_t\mathcal{P}} \setminus Err \rangle$, where Err is the smallest set containing:

$$\{t \in T_{\mathcal{R}/_t\mathcal{P}} : \exists t' \in (\mathcal{A}^I_{\mathcal{R}/\mathcal{P}})^* \cdot \nexists t'' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{R}/\mathcal{P}}) \cdot tt't'' \in T_{\mathcal{R}/_t\mathcal{P}} \setminus Err \text{ and } tt't'' \text{ is of infinite duration}\}.$$ ◇

It is obvious that $\mathcal{R} \;/_{nt}\; \mathcal{P}$ is a non-terminating component, given the formulation of the Err set. The following theorem shows that the construction satisfies the standard properties for quotient.

Theorem 6.50. Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components. Then $\mathcal{P} \mid\mid_{nt} \mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{R}$ iff:

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

Proof. The first claim is standard. For the second claim, we show that $F_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))} \subseteq F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})})$ and $T_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))} \subseteq T_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})})$. First suppose that $t \in T_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))}$ and $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{R}})$. Then there is a prefix $t'i$ of t such that $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{R}})$ and $i \in \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})} \setminus \mathcal{A}_{\mathcal{R}}$. As t' is a strict prefix of t, it holds that $t' \in T_{\mathcal{E}(\mathcal{R})}$, and so $t'i \in T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})}$. Consequently, $t \in T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})}$ as required. If instead $t \in F_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))} \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}})$, then we demonstrate that $X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}}) \subseteq F_{\mathcal{E}(\mathcal{R})}$ for each $i \in \mathbb{N}$, where X_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))}$ defined as a fixed point. The case of $i = 0$ holds trivially, because $X_i = \emptyset$. For the inductive case, suppose $t \in X_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}})$. Then either (i) $t \in F_{\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P})}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})})^*$ such that $tt' \in X_k$, or (iii) $t \in T_{\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P})}$ and there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$ and for all prefixes t'' of t' and $a \in \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})}$ it holds that $tt''a \in X_k$. For (i), we derive $t \upharpoonright (\mathcal{A}_{\mathcal{P}} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{P}}$ and $t \upharpoonright (\mathcal{A}_{\mathcal{R}/\mathcal{P}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{R}/_{nt}\mathcal{P}}$, or $t \upharpoonright (\mathcal{A}_{\mathcal{P}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{P}}$ and $t \upharpoonright (\mathcal{A}_{\mathcal{R}/\mathcal{P}} \cup \mathbb{R}^\infty_0) \in F_{\mathcal{R}/_{nt}\mathcal{P}}$, both of which imply that $t \in F_{\mathcal{E}(\mathcal{R})}$ by the definition of quotient. For (ii), it holds by the induction hypothesis that $tt' \in F_{\mathcal{E}(\mathcal{R})}$, given that $\mathcal{A}^O_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})} \subseteq \mathcal{A}^O_{\mathcal{R}}$, and so $t \in F_{\mathcal{E}(\mathcal{R})}$ by the definition of the $\mathcal{E}$ operator. For (iii), it follows by the induction hypothesis that $tt' \in F_{\mathcal{E}(\mathcal{R})}$ and $tt''a \in F_{\mathcal{E}(\mathcal{R})}$ for each $a \in \mathcal{A}^I_{\mathcal{R}}$. As $\mathcal{A}^I_{\mathcal{R}} \subseteq \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})}$, it follows that $t \in F_{\mathcal{E}(\mathcal{R})}$ by the definition of the $\mathcal{E}$ operator. Thus, $F_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))} \subseteq F_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})})$. Finally, to show $T_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))} \subseteq T_{\mathcal{E}(\mathcal{R})} \cup (T_{\mathcal{E}(\mathcal{R})} \uparrow \mathcal{A}^I_{\mathcal{P}\mid\mid(\mathcal{R}/\mathcal{P})})$, suppose that $t \in (T_{\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P})} \setminus F_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}(\mathcal{R}/_{nt}\mathcal{P}))}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}})$. By the definition of $\mid\mid_{nt}$, it follows that $t \upharpoonright (\mathcal{A}_{\mathcal{P}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{P}}$ and $t \upharpoonright (\mathcal{A}_{\mathcal{R}/\mathcal{P}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{R}/_{nt}\mathcal{P}}$. Now, by the definition of $T_{\mathcal{R}/_{nt}\mathcal{P}}$, it follows that $t \in T_{\mathcal{E}(\mathcal{R})}$ as required.

For the third claim, we first show that $F_{\mathcal{E}(\mathcal{Q})} \subseteq F_{\mathcal{R}/_t\mathcal{P}} \cup (T_{\mathcal{R}/_t\mathcal{P}} \uparrow \mathcal{A}^I_{\mathcal{Q}})$ and $T_{\mathcal{E}(\mathcal{Q})} \subseteq T_{\mathcal{R}/_t\mathcal{P}} \cup (T_{\mathcal{R}/_t\mathcal{P}} \uparrow \mathcal{A}^I_{\mathcal{Q}})$, and then show that $T_{\mathcal{E}(\mathcal{Q})} \cap Err = \emptyset$. The reasoning for $t \in T_{\mathcal{E}(\mathcal{Q})}$ and $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{R}/\mathcal{P}})$ is the same as in the second claim. So we show that $X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}/\mathcal{P}}) \subseteq F_{\mathcal{R}/_t\mathcal{P}}$ for each $i \in \mathbb{N}$, where X_i is the i-th approximation of $F_{\mathcal{E}(\mathcal{Q})}$. The base case is trivial, so suppose $t \in X_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{R}/\mathcal{P}})$. Then either (i) $t \in F_{\mathcal{Q}}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{Q}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$ and for every prefix t'' of t' and $a \in \mathcal{A}^I_{\mathcal{Q}}$ it holds that $tt''a \in X_k$. For (i), let $t_r \in \mathcal{T}(\mathcal{A}_{\mathcal{R}})$ be an arbitrary trace such that $t_r \upharpoonright (\mathcal{A}_{\mathcal{Q}} \cup \mathbb{R}^\infty_0) = t$. Then if $t \upharpoonright (\mathcal{A}_{\mathcal{P}} \cup \mathbb{R}^\infty_0) \in T_{\mathcal{P}}$, it holds that $t \in F_{\mathcal{P}\mid\mid_{nt}\mathcal{Q}}$, so $t \in F_{\mathcal{E}(\mathcal{R})}$, since $\mathcal{P} \mid\mid_{nt} \mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{R}$. Hence $t \in F_{\mathcal{R}/_t\mathcal{P}}$ by definition, as required, given that $t_r \upharpoonright (\mathcal{A}_{\mathcal{R}/\mathcal{P}} \cup \mathbb{R}^\infty_0) = t$. For (ii) and (iii), the result holds by the usual reasoning on the induction hypothesis. It is easy to show that $F_{\mathcal{E}(\mathcal{R}/_t\mathcal{P})} = F_{\mathcal{R}/_t\mathcal{P}}$ by an inductive argument, so $t \in F_{\mathcal{R}/_t\mathcal{P}}$ as required. When $t \in T_{\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{Q})}$,

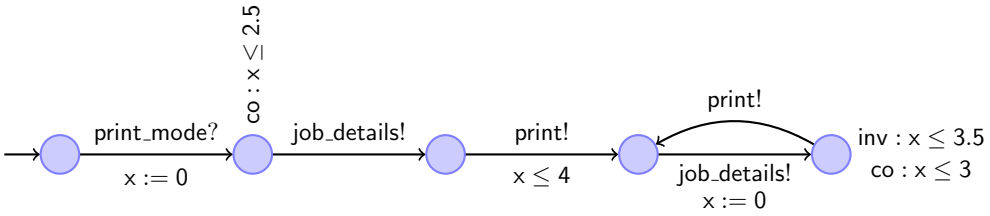

Figure 6.12: Non-terminating timed specification of a print system (PrintSystem)

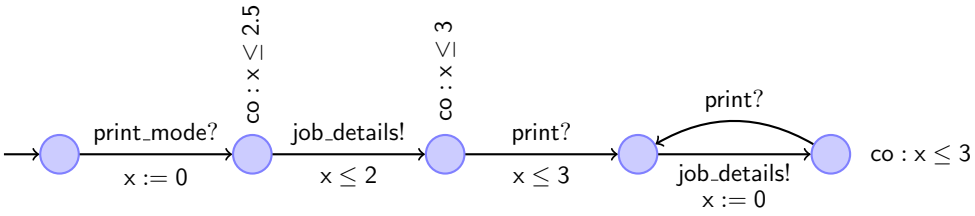

Figure 6.13: Non-terminating timed quotient of the printing/scanning device from PrintSystem

it is straightforward to show that $t \in T_{\mathcal{R}/_t\mathcal{P}}$, the reasoning matching that in Theorem 3.25. Now, to show that $T_{\mathcal{E}(\mathcal{Q})} \cap Err = \emptyset$, we demonstrate that $T_{\mathcal{E}(\mathcal{Q})} \cap Y_i = \emptyset$ for each $i \in \mathbb{N}$, where Y_i is the i-th approximation of Err defined as a fixed point. The base case is trivial, so suppose $t \in T_{\mathcal{E}(\mathcal{Q})} \cap Y_{k+1}$. Then there exists $t' \in (\mathcal{A}^I_{\mathcal{R}/\mathcal{P}})^*$ for which $\nexists t'' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{R}/\mathcal{P}})$ such that $tt't'' \in T_{\mathcal{R}/_t\mathcal{P}} \setminus Y_k$ and $tt't''$ is of infinite duration. Since $\mathcal{A}^I_{\mathcal{Q}} = \mathcal{A}^I_{\mathcal{R}/\mathcal{P}}$, it holds that $tt' \in T_{\mathcal{E}(\mathcal{Q})}$. Now as $\mathcal{Q}$ is a non-terminating component, there exists $t_q \in \mathcal{T}(\mathcal{A}^O_{\mathcal{Q}})$ such that $tt't_q \in T_{\mathcal{E}(\mathcal{Q})}$ and $tt't_q$ is of infinite duration. But as $\mathcal{A}^O_{\mathcal{Q}} \subseteq \mathcal{A}^O_{\mathcal{R}/\mathcal{P}}$, it follows that $t_q \in \mathcal{T}(\mathcal{A}^O_{\mathcal{R}/\mathcal{P}})$, hence $tt't_q \in \mathcal{T}(\mathcal{A}_{\mathcal{R}/\mathcal{P}})$. By the first part of the claim, we derive $tt't_q \in T_{\mathcal{R}/_t\mathcal{P}}$, hence $tt't_q \in Y_k$. But this contradicts the induction hypothesis $T_{\mathcal{E}(\mathcal{Q})} \cap Y_k = \emptyset$. Therefore, $T_{\mathcal{E}(\mathcal{Q})} \cap Err = \emptyset$ as required. Consequently, it is actually the case that $F_{\mathcal{E}(\mathcal{Q})} \subseteq F_{\mathcal{R}/_{nt}\mathcal{P}} \cup (T_{\mathcal{R}/_{nt}\mathcal{P}} \uparrow \mathcal{A}^I_{\mathcal{Q}})$ and $T_{\mathcal{E}(\mathcal{Q})} \subseteq T_{\mathcal{R}/_{nt}\mathcal{P}} \cup (T_{\mathcal{R}/_{nt}\mathcal{P}} \uparrow \mathcal{A}^I_{\mathcal{Q}})$. □

Unsurprisingly, the compositionality result for quotient continues to hold in the non-terminating theory of timed components.

Theorem 6.51. Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be components such that $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$.

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

Proof. Follows by the same reasoning as in Theorem 3.27, having updated the references to the corresponding results from this section. □

Example 6.52. Figure 6.12 presents a specification for a print system (PrintSystem), consisting of a printer and spooler. The quotient of the printing/scanning device in Figure 6.9 from

PrintSystem is the component shown in Figure 6.13, which can be thought of as a specification for the spooler. After being placed in print_mode, the spooler is willing to generate a bounded-liveness error within 2.5 time units, since this is allowed by PrintSystem. However, the spooler may only issue the job_details within 2 time units, as this is the only time range when the printing/scanning device will successfully accept them, without generating a communication mismatch, the latter of which is not permitted by PrintSystem. After issuing the job_details, the spooler may time-out within 3 time units, even though this is not permitted by PrintSystem, since the invariant $x \leq 3$ in the printing/scanning device forces the print action to be taken. The remaining behaviour is self-explanatory. Note that none of the traces arising in the diagrammatic representation of the spooler component are terminating. Therefore, the pruning operation defined as part of the quotient operator does not need to be applied. ◇

6.3.7 Full Abstraction

In this section, we demonstrate that the refinement preorder on components in the non-terminating timed theory preserves the absence of safety and bounded-liveness errors, as has been shown for the relation in the terminating theory. Based on this, component equivalence in the terminating theory is shown to be fully abstract with respect to observation of inconsistency.

Definition 6.53. Let $\mathcal{P}$ and $\mathcal{Q}$ be components. Then $\mathcal{Q}$ is inconsistency substitutable for $\mathcal{P}$, denoted by $\mathcal{Q} \sqsubseteq_{imp}^{F,nt} \mathcal{P}$, iff for each $t \in \mathcal{T}(\mathcal{A}_{\mathcal{Q}}^{O})$ it holds that $t \in F_{\mathcal{E}(\mathcal{Q})}$ implies $t \in F_{\mathcal{E}(\mathcal{P})}$. ◇

Theorem 6.54. Let $\mathcal{P}$ and $\mathcal{Q}$ be components such that $\mathcal{A}_{\mathcal{P}}^{I} \subseteq \mathcal{A}_{\mathcal{Q}}^{I}$, $\mathcal{A}_{\mathcal{Q}}^{O} \subseteq \mathcal{A}_{\mathcal{P}}^{O}$ and $\mathcal{A}_{\mathcal{Q}}^{I} \cap \mathcal{A}_{\mathcal{P}}^{O} = \emptyset$. Then:

$$\mathcal{Q} \sqsubseteq_{imp}^{nt} \mathcal{P} \text{ iff } \forall \mathcal{R} \cdot \mathcal{A}_{\mathcal{R}}^{O} = \mathcal{A}_{\mathcal{P}}^{I} \text{ and } \mathcal{A}_{\mathcal{R}}^{I} = \mathcal{A}_{\mathcal{Q}}^{O} \implies \mathcal{Q} \,||_{nt}\, \mathcal{R} \sqsubseteq_{imp}^{F,nt} \mathcal{P} \,||_{nt}\, \mathcal{R}.$$

Proof. Straightforward modification of Theorem 6.31. □

Consequently, $\sqsubseteq_{imp}^{nt}$ is the weakest preorder preserving substitutivity of components, and so $\equiv_{imp}^{nt}$ is the coarsest equivalence on components with respect to observation of inconsistency. Given that $\equiv_{imp}^{nt}$ is shown to be a congruence for all of the operators, subject to composability, it follows that $\equiv_{imp}^{nt}$ is fully abstract for the specification theory with respect to observational equivalence of safety and bounded-liveness errors.

Corollary 6.55. Substitutive equivalence $\equiv_{imp}^{nt}$ is fully abstract for parallel composition, conjunction, disjunction, hiding and quotient with respect to observational equivalence of inconsistency.

Proof. The same reasoning as in Corollary 6.32. □

6.4 Summary

In this chapter, we have extended the trace-based compositional specification theory of Chapter 3 to the real-time setting, by recording the times at which interactions occur, in addition to their temporal ordering. Two variants of the framework are provided, in order to support the realistic modelling of different types of timed systems. First, we present a framework that allows for the passage of time to be halted, which corresponds to systems that can terminate, while in the subsequent framework we remove the possibility of termination, so that systems must run indefinitely. Linear-time refinements are provided for each framework, which are shown to be the weakest preorders preserving the absence of safety and bounded-liveness errors. Safety errors correspond to the arising of unexpected interactions, while bounded-liveness errors occur when a system passes a certain point in time without having performed a particular interaction. Definitions of parallel composition, conjunction, disjunction, hiding and quotient are provided for each framework, and full abstraction results are supplied.

The conceptual simplicity of the frameworks highlights the essential structure required for reasoning compositionally about safety and bounded-liveness errors. By equating these two types of violations, we have presented a formalism that enjoys strong algebraic properties, and is capable of modelling a range of asynchronous timed systems.

CHAPTER

SEVEN

Assume-Guarantee Reasoning for Timed Components

In this chapter, we present an assume-guarantee reasoning framework for the theory of non-terminating timed components, on the basis that the non-terminating theory has greater practical applicability and is more complicated than the terminating theory. The framework is based largely on the constructs from Chapter 5, both in the safety and progress-sensitive settings. For example, the structure of a timed contract is closely related to the definition of a contract in the safety setting, while the definition of refinement is more akin to the version from the progress-sensitive framework. As in Chapter 5, we define the usual collection of operators directly on contracts (i.e., parallel composition, conjunction, disjunction and quotient), and further present sound and complete assume-guarantee rules that allow for the inference of properties satisfied by compositions of timed components, based on the compositions of their satisfying contracts.

7.1 Timed Contracts

We begin by defining contracts for the timed case.

Definition 7.1 (Contract). A *contract* $\mathcal{S}$ is a tuple $\langle \mathcal{A}^I_{\mathcal{S}}, \mathcal{A}^O_{\mathcal{S}}, \mathcal{R}_{\mathcal{S}}, \mathcal{G}_{\mathcal{S}} \rangle$, in which $\mathcal{A}^I_{\mathcal{S}}$ and $\mathcal{A}^O_{\mathcal{S}}$ are disjoint sets (whose union is $\mathcal{A}_{\mathcal{S}}$), referred to as the inputs and outputs respectively, and $\mathcal{R}_{\mathcal{S}}$ and $\mathcal{G}_{\mathcal{S}}$ are prefix closed subsets of $\mathcal{T}(\mathcal{A}_{\mathcal{S}})$, referred to as the assumption and guarantee respectively, such that $t \in \mathcal{R}_{\mathcal{S}}$ and $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{S}})$ implies $tt' \in \mathcal{R}_{\mathcal{S}}$. $\diamond$

This definition of a contract is a generalisation of the one provided in Definition 5.1, by taking the assumption and guarantee to be timed trace sets. In the untimed setting, the assumption had to be closed under output extensions, because the environment cannot constrain the output behaviour of a satisfying component. However, here in the timed setting, the assumption must be closed under timed output extensions, since the environment also cannot constrain the passage of time.

Definition 7.2 (Satisfaction). A non-terminating timed component $\mathcal{P}$ *satisfies* the contract $\mathcal{S}$, written $\mathcal{P} \models_{nt} \mathcal{S}$, iff:

S1. $\mathcal{A}^I_{\mathcal{S}} \subseteq \mathcal{A}^I_{\mathcal{P}}$

S2. $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{S}}$

S3. $\mathcal{A}^I_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}} = \emptyset$

S4. $\mathcal{R}_{\mathcal{S}} \cap T_{\mathcal{P}} \subseteq \mathcal{G}_{\mathcal{S}} \cap \overline{F_{\mathcal{P}}}$. ◇

The conditions for the satisfaction of a contract by a non-terminating timed component remain unchanged from Definition 5.3, since any common interaction between the environment and component that lies within the assumption must also be contained within the guarantee, and must not allow the component to become inconsistent under its own control.

At this stage, we make an important distinction between satisfaction in the untimed and timed settings. For the untimed setting, assume that Δ (representing an assumption) is a prefix closed set of traces over $\mathcal{A}_{\mathcal{P}}$. Then checking $\Delta \cap T_{\mathcal{P}} \subseteq \overline{F_{\mathcal{E}(\mathcal{P})}}$ is equivalent to verifying $\Delta \cdot (\mathcal{A}^O_{\mathcal{P}})^* \cap T_{\mathcal{P}} \subseteq \overline{F_{\mathcal{P}}}$. That is to say, by making Δ closed under output extensions, it is sufficient to check exclusion of common traces between the environment and component from the set of inconsistent traces $F_{\mathcal{P}}$, rather than checking that they are excluded from the set of traces $F_{\mathcal{E}(\mathcal{P})}$ which have the potential to become inconsistent under $\mathcal{P}$'s own control. However, when Δ is a prefix closed subset of $\mathcal{T}(\mathcal{A}_{\mathcal{P}})$, an analogous result does not hold, even when considering $\Delta \cdot \mathcal{T}(\mathcal{A}^O_{\mathcal{P}})$ in place of $\Delta \cdot (\mathcal{A}^O_{\mathcal{P}})^*$. This is because interactions are assumed to take precedence over the passage of time (cf. Definition 6.35). Considering the phenomenon in more detail, a timed trace t is not necessarily in $F_{\mathcal{E}(\mathcal{P})}$ even if there exists $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}})$ such that $tt' \in F_{\mathcal{P}}$. This is because the environment could issue some input on a prefix of t' that will deflect away from the behaviour tt', due to the issuance of an input taking precedence over the passage of time. But, as the contract has no way of ensuring that such an input will be issued by the environment, it is not possible to guarantee that the behaviour tt' will not materialise. This is why it is essential that the assumption is closed under timed output extensions in the timed setting.

Any timed component that satisfies a contract can be replaced by a refinement such that the new component will automatically satisfy the contract, as the following lemma shows.

Lemma 7.3. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}$ be a contract. If $\mathcal{P} \models_{nt} \mathcal{S}$, $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$ and $\mathcal{A}^I_{\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{S}} = \emptyset$, then $\mathcal{Q} \models_{nt} \mathcal{S}$.

Proof. Suppose that $t \in \mathcal{R}_{\mathcal{S}} \cap T_{\mathcal{Q}}$. Then it is necessarily the case that $t \in \mathcal{T}(\mathcal{A}_{\mathcal{P}})$, so from $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$ we derive $t \in \mathcal{R}_{\mathcal{S}} \cap T_{\mathcal{E}(\mathcal{P})}$. If $t \in T_{\mathcal{P}}$, then $t \in \mathcal{G}_{\mathcal{S}} \cap \overline{F_{\mathcal{P}}}$, from $\mathcal{P} \models_{nt} \mathcal{S}$. If instead $t \in F_{\mathcal{E}(\mathcal{P})}$, then there exists $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{P}})$, such that $tt' \in F_{\mathcal{P}}$. But as $tt' \in \mathcal{R}_{\mathcal{S}}$, it would follow that $\mathcal{P} \not\models_{nt} \mathcal{S}$, which is contradictory. So, from $t \notin F_{\mathcal{E}(\mathcal{P})}$, we derive $t \notin F_{\mathcal{E}(\mathcal{Q})}$ from $\mathcal{Q} \sqsubseteq^{nt}_{imp} \mathcal{P}$, which implies $t \notin F_{\mathcal{Q}}$. Thus, $t \in \mathcal{G}_{\mathcal{S}} \cap \overline{F_{\mathcal{Q}}}$ as required. □

In keeping with the untimed setting, we show how to construct the least refined component satisfying a contract. To do this, we need to determine the set of traces, denoted $\mathsf{error}(\mathcal{S})$, that cannot arise in any implementation of $\mathcal{S}$. This set is obtained by appealing to the definition of the $\mathcal{E}$ operator (Definition 6.35), having transposed inputs with outputs.

Definition 7.4. Let $\mathcal{S}$ be a contract. Then:

- $\mathsf{violations}(\mathcal{S})$ is defined as $\{t \in \mathcal{T}(\mathcal{A}_\mathcal{S}) : \exists t' \in (\mathcal{A}^I_\mathcal{S})^* \cdot tt' \in \mathcal{R}_\mathcal{S} \cap \overline{\mathcal{G}_\mathcal{S}}\} \cdot \mathcal{T}(\mathcal{A}_\mathcal{S})$
- $\mathsf{error}(\mathcal{S})$ is defined as the smallest set containing:
$$\mathsf{violations}(\mathcal{S}) \cup \{t \in \mathcal{T}(\mathcal{A}_\mathcal{S}) : \exists t' \in (\mathcal{A}^I_\mathcal{S})^* \cdot tt' \in \mathsf{error}(\mathcal{S})\} \cup$$
$$\{t \in \mathcal{T}(\mathcal{A}_\mathcal{S}) : \exists t' \in \mathbb{R}^\infty_0 \cdot tt' \in \mathsf{error}(\mathcal{S}), \text{ and}$$
$$t'' \text{ is a prefix of } t' \text{ and } o \in \mathcal{A}^O_\mathcal{S} \implies tt''o \in \mathsf{error}(\mathcal{S})\}. \qquad \diamond$$

Essentially, the formulation of $\mathsf{violations}(\mathcal{S})$ is unchanged from the non-timed setting. A timed trace is a violation if there exists a sequence of inputs that leads to a trace in the assumption while not being in the guarantee. Note that the sequence of inputs is *not* timed, as this is handled by the formulation of the $\mathsf{error}(\mathcal{S})$ set, owing to the difference in precedence afforded between interactions and the passage of time (interactions occur non-deterministically and take precedence over the passage of time (cf. Definition 6.35)). The $\mathsf{error}(\mathcal{S})$ set is designed to capture all traces from which there is no possibility of a component avoiding a violation. Thus, $\mathsf{error}(\mathcal{S})$ is the smallest set containing $\mathsf{violations}(\mathcal{S})$, along with any trace t that can be extended by a sequence of timed inputs t' such that $tt' \in \mathsf{error}(\mathcal{S})$ and, for any prefix t'' of t', at least one of the following holds.

- There exists $i \in \mathcal{A}^I_\mathcal{S}$ such that $t''i$ is a prefix of t'. Since the component cannot prevent the environment from issuing the input i after t'', under some resolution of non-determinism $t''i$ must be a behaviour of the component, even if the component can issue outputs immediately after t''.
- If t'' can delay such that it is still a prefix of t', then for each $o \in \mathcal{A}^O_\mathcal{S}$ we require that $tt''o \in \mathsf{error}(\mathcal{S})$. If instead there was some o' such that $tt''o' \notin \mathsf{error}(\mathcal{S})$, then some component could always issue this o' after t'', which would automatically take precedence over the delay, always allowing the component to steer away from the trace tt'.

As usual, $\mathsf{error}(\mathcal{S})$ can be defined iteratively as a least fixed point. More precisely, $\mathsf{error}(\mathcal{S}) = \sum_{i \in \mathbb{N}} X_i$, where $X_0 = \emptyset$ and

$$X_{k+1} = \mathsf{violations}(\mathcal{S}) \cup \{t \in \mathcal{T}(\mathcal{A}_\mathcal{S}) : \exists t' \in (\mathcal{A}^I_\mathcal{S})^* \cdot tt' \in X_k\} \cup$$
$$\{t \in \mathcal{T}(\mathcal{A}_\mathcal{S}) : \exists t' \in \mathbb{R}^\infty_0 \cdot tt' \in X_k, \text{ and}$$
$$t'' \text{ is a prefix of } t' \text{ and } o \in \mathcal{A}^O_\mathcal{S} \implies tt''o \in X_k\}.$$

Note that $\overline{\mathsf{error}(\mathcal{S})}$ is a collection of non-terminating traces, meaning that, for each $t \in \overline{\mathsf{error}(\mathcal{S})}$, there exists $t' \in \mathcal{T}(\mathcal{A}^O_\mathcal{S})$ such that $tt' \in \overline{\mathsf{error}(\mathcal{S})}$ and tt' has infinite duration. Thus, $\overline{\mathsf{error}(\mathcal{S})}$ can be used to define the traces of the least refined component satisfying $\mathcal{S}$, as per Definition 5.6 in the untimed setting.

Definition 7.5. Let $\mathcal{S}$ be a contract. Then the *least refined component satisfying* $\mathcal{S}$ is the component $\mathcal{I}_{nt}(\mathcal{S}) = \langle \mathcal{A}^I_{\mathcal{S}}, \mathcal{A}^O_{\mathcal{S}}, T_{\mathcal{I}_{nt}(\mathcal{S})}, F_{\mathcal{I}_{nt}(\mathcal{S})} \rangle$, where:

- $T_{\mathcal{I}_{nt}(\mathcal{S})} = \overline{\mathsf{error}(\mathcal{S})}$
- $F_{\mathcal{I}_{nt}(\mathcal{S})} = \overline{\mathsf{error}(\mathcal{S})} \cap \overline{\mathcal{R}_{\mathcal{S}}}$. ◇

Based on the construction of $\mathcal{I}_{nt}(\mathcal{S})$, it is fairly obvious that the following properties hold.

Lemma 7.6. Let $\mathcal{S}$ be a contract and $\mathcal{P}$ be a component. Then:

- $\mathcal{I}_{nt}(\mathcal{S})$ is non-realisable implies $\mathcal{S}$ is non-implementable;
- $\mathcal{I}_{nt}(\mathcal{S}) \models_{nt} \mathcal{S}$; and
- $\mathcal{P} \models_{nt} \mathcal{S}$ iff $\mathcal{P} \sqsubseteq^{nt}_{imp} \mathcal{I}_{nt}(\mathcal{S})$.

Proof. For the first claim, we begin by showing that $t \in \mathsf{error}(\mathcal{S})$ implies t is not a trace of any implementation of $\mathcal{S}$, by demonstrating that $t \in X_i$ implies t is not a trace in an implementation of $\mathcal{S}$ for each $i \in \mathbb{N}$, where X_i is the i-th approximation of $\mathsf{error}(\mathcal{S})$. The base case is trivial, so suppose that $t \in X_{k+1}$. Then (i) $t \in \mathsf{violations}(\mathcal{S})$, (ii) there exists $t' \in (\mathcal{A}^I_{\mathcal{S}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$, and for each t'' a prefix of t' and $o \in \mathcal{A}^O_{\mathcal{S}}$ it holds that $tt''o \in X_k$. For (i), if $t \in \mathsf{violations}(\mathcal{S})$, then there exists $t' \in (\mathcal{A}^I_{\mathcal{S}})^*$ such that $tt' \in \mathcal{R}_{\mathcal{S}} \cap \overline{\mathcal{G}_{\mathcal{S}}}$. Hence tt' cannot be in any implementation of $\mathcal{S}$, which implies t is not in any implementation by input-receptivity. For (ii), tt' is not in any implementation by the induction hypothesis, which implies t is not in any implementation by input-receptivity. For (iii), by the induction hypothesis, we know that tt' and each of the $tt''o$ cannot be in any implementation, hence tt'' is a terminating trace, which implies that it cannot be in any implementation. Hence t cannot be in any implementation. Now if $\mathcal{I}_{nt}(\mathcal{S})$ is non-realisable, then $0 \notin T_{\mathcal{I}_{nt}(\mathcal{S})}$, which implies $0 \in \mathsf{error}(\mathcal{S})$. Thus 0 cannot be a trace in any implementation of $\mathcal{S}$, which implies that $\mathcal{S}$ has no implementations.

For the second claim, suppose that $t \in \mathcal{R}_{\mathcal{S}} \cap T_{\mathcal{I}_{nt}(\mathcal{S})}$. Then $t \notin \mathsf{error}(\mathcal{S})$, which implies $t \notin F_{\mathcal{I}_{nt}(\mathcal{S})}$ and $t \notin \mathsf{violations}(\mathcal{S})$. From the latter, it follows that $t \in \mathcal{G}_{\mathcal{S}}$.

For the third claim, the if direction follows by the previous claim and Lemma 7.3. For the only if direction, we need to show that $T_{\mathcal{E}(\mathcal{P})} \subseteq T_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))} \cup (T_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))} \uparrow \mathcal{A}^I_{\mathcal{P}})$ and $F_{\mathcal{E}(\mathcal{P})} \subseteq F_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))} \cup (T_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))} \uparrow \mathcal{A}^I_{\mathcal{P}})$. So suppose that $t \in T_{\mathcal{E}(\mathcal{P})}$ and assume that the result holds for all strict prefixes. If $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{S}})$, then there is a prefix $t'i$ of t such that $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{S}})$ and $i \in \mathcal{A}^I_{\mathcal{P}} \setminus \mathcal{A}_{\mathcal{S}}$. Hence, $t' \in T_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$ by the induction hypothesis and $t'i \in T_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))} \uparrow \mathcal{A}^I_{\mathcal{P}}$, which implies $t \in T_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))} \uparrow \mathcal{A}^I_{\mathcal{P}}$. Now suppose that $t \in F_{\mathcal{E}(\mathcal{P})} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}})$. In the usual manner, we approximate $F_{\mathcal{E}(\mathcal{P})}$ with Y_i. So for the inductive case of $t \in Y_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{P}})$, either (i) $t \in F_{\mathcal{P}}$, (ii) there exists $t' \in (\mathcal{A}^O_{\mathcal{P}})^*$ such that $tt' \in Y_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in Y_k$ and for each t'' a

prefix of t' and $a \in \mathcal{A}^I_{\mathcal{P}}$ it holds that $tt''a \in Y_k$. For (i), it follows that $t \in T_{\mathcal{P}}$, which implies (by the first claim) $t \notin \mathsf{error}(\mathcal{S})$. Moreover, since $\mathcal{P} \models_{nt} \mathcal{S}$, it follows that $t \notin \mathcal{R}_{\mathcal{S}}$. Hence $t \in F_{\mathcal{I}_{nt}(\mathcal{S})}$, meaning $t \in F_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$ as required. For (ii), since $\mathcal{A}^O_{\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{S}}$, it follows that $tt' \in \mathcal{T}(\mathcal{A}_{\mathcal{S}})$. By the induction hypothesis we derive $tt' \in F_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$, which implies $t \in F_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$ by the definition of the $\mathcal{E}$ operator. For (iii), again by the induction hypothesis we see that $tt' \in F_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$ and $tt''a \in F_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$ when $a \in \mathcal{A}^I_{\mathcal{S}}$, since $\mathcal{A}^I_{\mathcal{S}} \subseteq \mathcal{A}^I_{\mathcal{P}}$. Thus, by the definition of the $\mathcal{E}$ operator, we derive $t \in F_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$. Finally, when $t \in T_{\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}})$, it follows by the first claim that $t \notin \mathsf{error}(\mathcal{S})$. Hence $t \in T_{\mathcal{I}_{nt}(\mathcal{S})}$, implying $t \in T_{\mathcal{E}(\mathcal{I}_{nt}(\mathcal{S}))}$. □

7.2 Refinement

The principles behind refinement are largely unchanged in the timed setting, and indeed the definition is syntactically similar to Definitions 5.9 and 5.42. As was the case in the untimed frameworks of Chapter 5, refinement continues to correspond to implementation containment.

Definition 7.7 (Refinement). Let $\mathcal{S}$ and $\mathcal{T}$ be contracts. $\mathcal{S}$ is said to be a *non-terminating timed refinement* of $\mathcal{T}$, written $\mathcal{S} \sqsubseteq_{nt} \mathcal{T}$, iff:

TCR1. $\mathcal{A}^I_{\mathcal{T}} \subseteq \mathcal{A}^I_{\mathcal{S}}$

TCR2. $\mathcal{A}^O_{\mathcal{S}} \subseteq \mathcal{A}^O_{\mathcal{T}}$

TCR3. $\mathcal{A}^I_{\mathcal{S}} \cap \mathcal{A}^O_{\mathcal{T}} = \emptyset$

TCR4. $\mathsf{error}(\mathcal{T}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}}) \subseteq \mathsf{error}(\mathcal{S})$

TCR5. $\mathcal{R}_{\mathcal{T}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}}) \subseteq \mathcal{R}_{\mathcal{S}} \cup \mathsf{error}(\mathcal{S})$. ◇

The intuition behind this definition is unchanged from Chapter 5. The following lemma makes clear the intended understanding of refinement in terms of implementation containment.

Lemma 7.8. Refinement captures implementation containment:

$$\mathcal{S} \sqsubseteq_{nt} \mathcal{T} \iff \{\mathcal{P} : \mathcal{P} \models_{nt} \mathcal{S} \text{ and } \mathcal{A}^I_{\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{T}} = \emptyset\} \subseteq \{\mathcal{P} : \mathcal{P} \models_{nt} \mathcal{T}\}.$$

Proof. The proof follows by Lemma 5.43, omitting the liveness trace containment. Note that the proof treats the error sets syntactically, so there is no need to recourse to the actual definition. □

Given that the definition of refinement is syntactically unchanged from the untimed setting, it is not surprising that $\sqsubseteq_{nt}$ is reflexive and transitive subject to compatibility.

Lemma 7.9 (Weak transitivity). Let $\mathcal{S}$, $\mathcal{T}$ and $\mathcal{U}$ be contracts such that $\mathcal{A}^I_{\mathcal{S}} \cap \mathcal{A}^O_{\mathcal{U}} = \emptyset$. If $\mathcal{S} \sqsubseteq_{nt} \mathcal{T}$ and $\mathcal{T} \sqsubseteq_{nt} \mathcal{U}$, then $\mathcal{S} \sqsubseteq_{nt} \mathcal{U}$.

Proof. Follows from the transitivity of subset inclusion. □

Having established the key properties of refinement, we can now show how to construct the characteristic contract for a component, which is the contract having the component as the least refined implementation.

Definition 7.10. The *characteristic contract* for component $\mathcal{P}$ is a contract $\mathcal{AG}_{nt}(\mathcal{P}) = \langle \mathcal{A}^I_{\mathcal{P}}, \mathcal{A}^O_{\mathcal{P}}, \mathcal{R}_{\mathcal{AG}_{nt}(\mathcal{P})}, \mathcal{G}_{\mathcal{AG}_{nt}(\mathcal{P})} \rangle$, where $\mathcal{R}_{\mathcal{AG}_{nt}(\mathcal{P})} = \mathcal{T}(\mathcal{A}_{\mathcal{P}}) \setminus F_{\mathcal{E}(\mathcal{P})}$ and $\mathcal{G}_{\mathcal{AG}_{nt}(\mathcal{P})} = T_{\mathcal{P}} \setminus F_{\mathcal{E}(\mathcal{P})}$. ◇

Given that the definition of the characteristic contract is hardly changed from the untimed setting, it is straightforward to show that the following properties continue to hold.

Lemma 7.11. Let $\mathcal{P}$ be a component and let $\mathcal{S}$ be a contract. Then:

- $\mathcal{P} \models_{nt} \mathcal{AG}_{nt}(\mathcal{P})$; and
- $\mathcal{P} \models_{nt} \mathcal{S}$ iff $\mathcal{AG}_{nt}(\mathcal{P}) \sqsubseteq_{nt} \mathcal{S}$.

Proof. The first claim in unchanged from Lemma 5.14. For the second claim, the if direction follows by the previous claim and Lemma 7.8. For the only if direction of the second claim, the reasoning follows by the same reasoning as in Lemma 7.8, having disregarded the liveness set containment. □

7.3 Parallel Composition

In the timed setting, the parallel composition of two contracts continues to be a contract having the least number of implementations that satisfies independent implementability. The definition is syntactically equivalent to that in the progress-sensitive theory without time, having updated the definition of the error traces to the timed setting.

Definition 7.12. Let $\mathcal{S}_{\mathcal{P}}$ and $\mathcal{S}_{\mathcal{Q}}$ be contracts composable for parallel composition (i.e., $\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}} \cap \mathcal{A}^O_{\mathcal{S}_{\mathcal{Q}}} = \emptyset$). Then $\mathcal{S}_{\mathcal{P}} \,||_{nt}\, \mathcal{S}_{\mathcal{Q}}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}, \mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}, \mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_{nt}\mathcal{S}_{\mathcal{Q}}}, \mathcal{G}_{\mathcal{S}_{\mathcal{P}}||_{nt}\mathcal{S}_{\mathcal{Q}}} \rangle$, where:

- $\mathcal{A}^I_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}} = (\mathcal{A}^I_{\mathcal{S}_{\mathcal{P}}} \cup \mathcal{A}^I_{\mathcal{S}_{\mathcal{Q}}}) \setminus (\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}} \cup \mathcal{A}^O_{\mathcal{S}_{\mathcal{Q}}})$
- $\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}} = \mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}} \cup \mathcal{A}^O_{\mathcal{S}_{\mathcal{Q}}}$
- $\mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_{nt}\mathcal{S}_{\mathcal{Q}}}$ is the largest prefix closed set such that $\mathcal{R}_{\mathcal{S}_{\mathcal{P}}||_{nt}\mathcal{S}_{\mathcal{Q}}} \cdot \mathcal{T}(\mathcal{A}^O_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}})$ is contained within the union of:
  - $(\mathcal{R}_{\mathcal{S}_{\mathcal{P}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}}) \cap (\mathcal{R}_{\mathcal{S}_{\mathcal{Q}}} \Uparrow \mathcal{A}_{\mathcal{S}_{\mathcal{P}}||\mathcal{S}_{\mathcal{Q}}})$

- $\mathsf{error}(\mathcal{S}_\mathcal{P}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$
- $\mathsf{error}(\mathcal{S}_\mathcal{Q}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$

- $\mathcal{G}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}} = \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}} \cap (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}}) \cap (\overline{\mathsf{error}(\mathcal{S}_\mathcal{Q}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}})$. ◇

Given that the definition of parallel composition is syntactically unchanged from the progress-sensitive setting (cf Definition 5.47) it is not surprising that the monotonicity result continues to hold, subject to the usual constraints on interfaces. First, however, we present a decomposition result on traces in the error set for the parallel composition, which is a timed extension of Lemma 5.48. This result is useful for proving that parallel composition is monotonic.

Lemma 7.13. $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} ||_{nt} \mathcal{S}_\mathcal{Q})$ implies $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ or $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$.

Proof. Show that $t \in X_i$ implies $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ or $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$, where X_i is the i-th iteration of defining $\mathsf{error}(\mathcal{S}_\mathcal{P} ||_{nt} \mathcal{S}_\mathcal{Q})$ as a least fixed point. When $i = 0$, the result hold trivially, since $X_0 = \emptyset$. So suppose that $t \in X_{k+1}$. Then (i) $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} ||_{nt} \mathcal{S}_\mathcal{Q})$, (ii) there exists $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}_0^\infty$ such that $tt' \in X_k$ and for each t'' a prefix of t' and $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$ it holds that $tt''o \in X_k$. For (i), there exists a prefix and input extension $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}}}$. So, without loss of generality, $t' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ by the definition of $||_{nt}$, from which it follows $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. For (ii), $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})^*$ implies $t' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}})^*$ and $t' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{Q}})^*$. By the induction hypothesis we have, without loss of generality, $tt' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$, hence $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. For (iii), by the induction hypothesis, we have without loss of generality $tt' \restriction \mathcal{A}_{\mathcal{S}_\mathcal{P}} \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Now for each prefix t'' and $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P}}$, we have that $tt'o \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ or $tt'o \restriction \mathcal{A}_{\mathcal{S}_\mathcal{Q}} \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$, given that $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}}$. If the latter holds, then $o \notin \mathcal{A}_\mathcal{Q}$ or $o \in \mathcal{A}^I_\mathcal{Q}$, which implies $tt'' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Therefore, by the definition of $\mathsf{error}(\mathcal{S}_\mathcal{P})$, it holds that $tt'' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ or $tt'' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. □

We can now present the monotonicity result for parallel composition.

Theorem 7.14. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for parallel composition, such that $\mathcal{A}_{\mathcal{S}'_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} = \emptyset$. If $\mathcal{S}'_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$, then $\mathcal{S}'_\mathcal{P} ||_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} ||_{nt} \mathcal{S}_\mathcal{Q}$.

Proof. Note that the alphabet constraints are satisfied, so we need to show that $\mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}) \subseteq \mathcal{R}_{\mathcal{S}'_\mathcal{P}||_{nt}\mathcal{S}'_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}'_\mathcal{P} ||_{nt} \mathcal{S}'_\mathcal{Q})$ and $\mathsf{error}(\mathcal{S}_\mathcal{P} ||_{nt} \mathcal{S}_\mathcal{Q}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}}) \subseteq \mathsf{error}(\mathcal{S}'_\mathcal{P} ||_{nt} \mathcal{S}'_\mathcal{Q})$.

First suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}})$, and assume that all strict prefixes of t are in $\mathcal{R}_{\mathcal{S}'_\mathcal{P}||_{nt}\mathcal{S}'_\mathcal{Q}} \cap \overline{\mathsf{error}(\mathcal{S}'_\mathcal{P} ||_{nt} \mathcal{S}'_\mathcal{Q})}$. If $t \notin \mathcal{R}_{\mathcal{S}'_\mathcal{P}||_{nt}\mathcal{S}'_\mathcal{Q}}$, then there exists $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}})$ such that,

without loss of generality, $tt' \restriction (\mathcal{A}_{\mathcal{S}'_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathcal{R}_{\mathcal{S}'_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}'_\mathcal{P})$ and $tt' \restriction (\mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \notin \mathsf{error}(\mathcal{S}'_\mathcal{Q})$. As $tt' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) = tt' \restriction (\mathcal{A}_{\mathcal{S}'_\mathcal{P}} \cup \mathbb{R}_0^\infty)$ and $tt' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) = tt' \restriction (\mathcal{A}_{\mathcal{S}'_\mathcal{Q}} \cup \mathbb{R}_0^\infty)$, it follows that $tt' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})$ since $\mathcal{S}'_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$, and $tt' \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \notin \mathsf{error}(\mathcal{S}_\mathcal{Q})$ since $\mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$. Hence, $tt' \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}}$, which implies $t \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}}$ as $t' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}})$, but this is contradictory.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \mid\mid_{nt} \mathcal{S}_\mathcal{Q}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}'_\mathcal{P}||\mathcal{S}'_\mathcal{Q}})$, and assume for the difficult case that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}}$. Then by Lemma 7.13 it follows that, without loss of generality, $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Since $t \restriction (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) = t \restriction (\mathcal{A}_{\mathcal{S}'_\mathcal{P}} \cup \mathbb{R}_0^\infty)$, it follows from $\mathcal{S}'_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$ that $t \restriction (\mathcal{A}_{\mathcal{S}'_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}'_\mathcal{P})$. Now from the first part, we know $t \in \mathcal{R}_{\mathcal{S}'_\mathcal{P}||_{nt}\mathcal{S}'_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}'_\mathcal{P} \mid\mid_{nt} \mathcal{S}'_\mathcal{Q})$, so it follows that $t \in \mathsf{error}(\mathcal{S}'_\mathcal{P} \mid\mid_{nt} \mathcal{S}'_\mathcal{Q})$, since certainly $t \notin \mathcal{G}_{\mathcal{S}'_\mathcal{P}||_{nt}\mathcal{S}'_\mathcal{Q}}$. □

From the monotonicity of parallel composition under refinement, it is possible to formulate a sound and complete assume-guarantee rule, akin to the one presented in Theorem 5.50.

Theorem 7.15. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}_\mathcal{P}$, $\mathcal{S}_\mathcal{Q}$ and $\mathcal{S}$ be contracts such that $\mathcal{A}_\mathcal{P} \cap \mathcal{A}_\mathcal{Q} \cap \mathcal{A}_{\mathcal{S}_\mathcal{P}||\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Timed-Parallel}\ \frac{\mathcal{P} \models_{nt} \mathcal{S}_\mathcal{P} \qquad \mathcal{Q} \models_{nt} \mathcal{S}_\mathcal{Q} \qquad \mathcal{S}_\mathcal{P} \mid\mid_{nt} \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}}{\mathcal{P} \mid\mid_{nt} \mathcal{Q} \models_{nt} \mathcal{S}}.$$

Proof. The result follows from the reasoning in Theorem 5.19, having shown that $\mathcal{AG}_{nt}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}) \sqsubseteq_{nt} \mathcal{AG}_{nt}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}_{nt}(\mathcal{Q}) \sqsubseteq_{nt} \mathcal{AG}_{nt}(\mathcal{P} \mid\mid_{nt} \mathcal{Q})$ (i.e., Lemma 7.16, below), and updating the references to the corresponding results from this chapter. □

The ancillary result required to prove soundness and completeness of the parallel composition AG rule is duly presented below.

Lemma 7.16. $\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}) \sqsubseteq_{nt} \mathcal{AG}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}(\mathcal{Q}) \sqsubseteq_{nt} \mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q})$.

Proof. First suppose that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||_{nt}\mathcal{AG}(\mathcal{Q})}$ and $t \notin \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}(\mathcal{Q}))$. Then $t \restriction \mathcal{A}_\mathcal{P} \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$, which implies that $t \restriction \mathcal{A}_\mathcal{P} \notin F_{\mathcal{E}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \notin F_{\mathcal{E}(\mathcal{Q})}$. Hence, $t \notin F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$, from which it follows that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||_{nt}\mathcal{Q})}$. For the other direction, suppose $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||_{nt}\mathcal{Q})}$ and $t \notin \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$. Then, $t \in \mathcal{G}_{\mathcal{AG}(\mathcal{P}||_{nt}\mathcal{Q})}$, which implies $t \in T_{\mathcal{P}||_{nt}\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}||_{nt}\mathcal{Q})}$, which means that $t \restriction \mathcal{A}_\mathcal{P} \notin F_{\mathcal{E}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \notin F_{\mathcal{E}(\mathcal{Q})}$ i.e., $t \restriction \mathcal{A}_\mathcal{P} \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})}$ and $t \restriction \mathcal{A}_\mathcal{Q} \in \mathcal{R}_{\mathcal{AG}(\mathcal{Q})}$. From this it follows that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})||_{nt}\mathcal{AG}(\mathcal{Q})}$, having noticed that no output extension of t can violate this constraint.

For the error set containments, suppose that $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}(\mathcal{Q}))$ and $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}||_{nt}\mathcal{Q})} \cap \mathcal{R}_{\mathcal{AG}(\mathcal{P})||_{nt}\mathcal{AG}(\mathcal{Q})}$. We demonstrate that $X_i \subseteq \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$ for each $i \in \mathbb{N}$, where X_i is the i-th iteration of defining the least fixed point characterising $\mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}(\mathcal{Q}))$. The result holds trivially when $i = 0$, since $X_i = \emptyset$. For the inductive case, suppose $t \in X_{k+1}$. Then

(i) $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}(\mathcal{Q}))$, (ii) there exists $t' \in (\mathcal{A}^I_{\mathcal{P}\mid\mid\mathcal{Q}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$ and for each t'' a prefix of t' and $o \in \mathcal{A}^O_{\mathcal{P}\mid\mid\mathcal{Q}}$, it holds that $tt''o \in X_k$. For (i), it follows that there exists t' a prefix and input extension over $(\mathcal{A}^I_{\mathcal{P}\mid\mid\mathcal{Q}})^*$ of t such that $t' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})\mid\mid_{nt}\mathcal{AG}(\mathcal{Q})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})\mid\mid_{nt}\mathcal{AG}(\mathcal{Q})}}$. Consequently, without loss of generality, $t' \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$, which implies $t \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$. Suppose for a contradiction that $t \in \mathcal{G}_{\mathcal{AG}(\mathcal{P}\mid\mid_{nt}\mathcal{Q})}$. Then $t \in T_{\mathcal{P}\mid\mid_{nt}\mathcal{Q}} \setminus F_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}\mathcal{Q})}$, which implies $t \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in T_\mathcal{P}$. But, as $t \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$, it follows that $\mathcal{P} \not\models_{nt} \mathcal{AG}(\mathcal{P})$, which is contradictory. Therefore, $t \notin \mathcal{G}_{\mathcal{AG}(\mathcal{P}\mid\mid_{nt}\mathcal{Q})}$ and so $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$. For (ii), by the induction hypothesis we have $tt' \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$, which implies $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$ by the formulation of the error set. For (iii), by the induction hypothesis we know that $tt' \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$, and moreover $tt''o \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$.Therefore, $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$ by the formulation of the error set, as required.

For the other direction of the error containment, suppose $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$ and $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}\mid\mid_{nt}\mathcal{Q})} \cap \mathcal{R}_{\mathcal{AG}(\mathcal{P})\mid\mid_{nt}\mathcal{AG}(\mathcal{Q})}$. Using a similar X_i argument for defining the least fixed point, it follows that (i) $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P} \mid\mid_{nt} \mathcal{Q}))$, (ii) there exists $t' \in (\mathcal{A}^I_{\mathcal{P}\mid\mid\mathcal{Q}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$ and for each prefix t'' of t' and $o \in \mathcal{A}^O_{\mathcal{P}\mid\mid\mathcal{Q}}$, it holds that $tt''o \in X_k$. For (i), there exists t' a prefix and input extension over $(\mathcal{A}^I_{\mathcal{P}\mid\mid\mathcal{Q}})^*$ of t such that $t' \in \mathcal{R}_{\mathcal{AG}(\mathcal{P}\mid\mid_{nt}\mathcal{Q})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P}\mid\mid_{nt}\mathcal{Q})}}$. Then $t' \notin T_{\mathcal{P}\mid\mid_{nt}\mathcal{Q}} \cup F_{\mathcal{E}(\mathcal{P}\mid\mid_{nt}\mathcal{Q})}$, which implies without loss of generality that $t' \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \notin T_\mathcal{P} \cup F_{\mathcal{E}(\mathcal{P})}$. Hence, $t' \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})} \cap \overline{\mathcal{G}_{\mathcal{AG}(\mathcal{P})}}$, which implies $t' \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$. Therefore, $t \upharpoonright (\mathcal{A}_\mathcal{P} \cup \mathbb{R}^\infty_0) \in \mathsf{error}(\mathcal{AG}(\mathcal{P}))$, which implies $t \notin \mathcal{G}_{\mathcal{AG}(\mathcal{P})\mid\mid_{nt}\mathcal{AG}(\mathcal{Q})}$. Consequently, $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}(\mathcal{Q}))$ as we are assuming that $t \in \mathcal{R}_{\mathcal{AG}(\mathcal{P})\mid\mid_{nt}\mathcal{AG}(\mathcal{Q})}$. For (ii) and (iii), the reasoning is the same as in the other direction. We conclude that $t \in \mathsf{error}(\mathcal{AG}(\mathcal{P}) \mid\mid_{nt} \mathcal{AG}(\mathcal{Q}))$. □

7.4 Conjunction

We now introduce the timed conjunction operator, which corresponds to the meet operation on the refinement preorder. This means that the conjunction of two contracts contains the intersection of the respective contracts' implementations. Consequently, an implementation in the conjunction must not be allowed to violate either of the contracts to be composed.

Definition 7.17. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ be contracts composable for conjunction (i.e., $\mathcal{A}^I_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$ and $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$ are disjoint). Then $\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}}, \mathcal{R}_{\mathcal{S}_\mathcal{P}\wedge_{nt}\mathcal{S}_\mathcal{Q}}, \mathcal{G}_{\mathcal{S}_\mathcal{P}\wedge_{nt}\mathcal{S}_\mathcal{Q}} \rangle$ defined by:

- $\mathcal{A}^I_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} = \mathcal{A}^I_{\mathcal{S}_\mathcal{P}} \cup \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}}$
- $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}\wedge\mathcal{S}_\mathcal{Q}} = \mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$

- $\mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}} = \left(\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathcal{R}_{\mathcal{S}_\mathcal{Q}}\right) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})$
- $\mathcal{G}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}}$ is the intersection of the following sets:
  - $\mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}}$
  - $\overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \cup (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}})$
  - $\overline{\mathsf{error}(\mathcal{S}_\mathcal{Q})} \cup (\overline{\mathsf{error}(\mathcal{S}_\mathcal{Q})} \uparrow \mathcal{A}^I_{\mathcal{S}_\mathcal{P}})$. ◇

The definition is syntactically unchanged from the progress-sensitive setting without time, excepting the restriction to timed traces over $\mathcal{A}_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$. Consequently, the usual compositionality results hold, as the following theorem demonstrates.

Theorem 7.18. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for conjunction. Then:

- $\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}_\mathcal{R} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{R} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}_\mathcal{R} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}'_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}'_\mathcal{P} \wedge_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}$.

Proof. First show that $\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$. Suppose $t \in \mathsf{error}(\mathcal{S}_\mathcal{P}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})$. Then there is a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})$ and $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Therefore, $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}}}$, implying $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q})$. If $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})$, then $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}}$ as required. By similar reasoning, it can be shown that $\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$.

For the second claim, we show $\mathsf{error}(\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{R}}) \subseteq \mathsf{error}(\mathcal{S}_\mathcal{R})$ by demonstrating that $t \in X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{R}})$ implies $t \in \mathsf{error}(\mathcal{S}_\mathcal{R})$ by induction on i, where X_i is the i-th iteration of defining $\mathsf{error}(\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q})$ as a least fixed point. When $i = 0$, the result holds trivially, as $X_i = \emptyset$. Now suppose $i = k$ for $k > 0$. If $t \in \mathsf{violations}(\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q})$, then there is a prefix t' of t and input extension $t'' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})^*$ such that $t't'' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}}}$. So, without loss of generality, $t't'' \notin \overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \cup (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \uparrow \mathcal{A}^I_{\mathcal{S}_\mathcal{Q}})$. This means that there is a prefix of $t't''$ contained in $\mathsf{error}(\mathcal{S}_\mathcal{P})$, which must also be in $\mathsf{error}(\mathcal{S}_\mathcal{R})$ since $\mathcal{S}_\mathcal{R} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$. Therefore, $t' \in \mathsf{error}(\mathcal{S}_\mathcal{R})$, which implies $t \in \mathsf{error}(\mathcal{S}_\mathcal{R})$. For the case when there exists $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}})^*$ such that $tt' \in X_{k-1}$, it follows that $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{R})$, given that $\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}^I_{\mathcal{S}_\mathcal{R}}$. Therefore, $t \in \mathsf{error}(\mathcal{S}_\mathcal{R})$, by the definition of the error construction. Finally, when there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_{k-1}$ and for each t'' a prefix of t' and $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$ it holds that $tt''o \in X_{k-1}$, then by the induction hypothesis we derive that $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{R})$ and $tt''o \in \mathsf{error}(\mathcal{S}_\mathcal{R})$, when $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{R}}$. As $\mathcal{A}^O_{\mathcal{S}_\mathcal{R}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \wedge \mathcal{S}_\mathcal{Q}}$, it holds by the error construction that $t \in \mathsf{error}(\mathcal{S}_\mathcal{R})$. To show the assumption containment, suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{R}})$. Then, without loss of generality, $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{R}})$, so from $\mathcal{S}_\mathcal{R} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$, we derive $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cup \mathsf{error}(\mathcal{S}_\mathcal{R})$ as required.

For the third claim, by the first claim we have $\mathcal{S}'_\mathcal{P} \wedge_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P} \wedge_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}'_\mathcal{Q}$. Now by transitivity, we see that $\mathcal{S}'_\mathcal{P} \wedge_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{P} \wedge_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$ providing $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^I_{\mathcal{S}'_\mathcal{Q}} = \emptyset$

and $\mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \cap \mathcal{A}^I_{\mathcal{S}'_\mathcal{P}} = \emptyset$, so by the second claim, it follows that $\mathcal{S}'_\mathcal{P} \wedge_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}$ as required. If either of the compatibility conditions are not satisfied, we can obtain new contracts $\mathcal{S}''_\mathcal{P}$ for $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}''_\mathcal{Q}$ for $\mathcal{S}_\mathcal{Q}$ that have output set $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \cap \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$ and contain all of the traces from the respective contracts, except for those with an output in $(\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \setminus \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}) \cup (\mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \setminus \mathcal{A}^O_{\mathcal{S}_\mathcal{P}})$ that has been removed from the interface. It is straightforward to show that $\mathcal{S}''_\mathcal{P} \wedge_{nt} \mathcal{S}''_\mathcal{Q} = \mathcal{S}_\mathcal{P} \wedge_{nt} \mathcal{S}_\mathcal{Q}$. □

From these properties of conjunction, we can formulate an AG rule that finds a contract satisfiable by a component, whenever the component satisfies multiple independently developed requirements. Naturally, such a contract must be a refinement of each of the contracts representing the requirements, so it can be taken as their conjunction.

Theorem 7.19. Let $\mathcal{P}$ be a component, and let $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}$ be contracts such that $\mathcal{A}^I_\mathcal{P} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Timed-Conjunction}\ \frac{\mathcal{P} \models_{nt} \mathcal{S}_1 \qquad \mathcal{P} \models_{nt} \mathcal{S}_2 \qquad \mathcal{S}_1 \wedge_{nt} \mathcal{S}_2 \sqsubseteq_{nt} \mathcal{S}}{\mathcal{P} \models_{nt} \mathcal{S}}.$$

Proof. The reasoning is unchanged from Theorem 5.23, when making use of Theorem 7.18 in place of Theorem 5.22. □

7.5 Disjunction

We now introduce the dual operator of conjunction, namely disjunction, which works by strengthening assumptions and weakening guarantees. As expected, the disjunctive operator corresponds to the join operation on the refinement preorder, meaning that the disjunction of two contracts contains the union of their implementations.

Definition 7.20. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$ be contracts composable for disjunction (i.e., the same conditions as for conjunction). Then $\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}, \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}}, \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}} \rangle$, where:

- $\mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}}$ is the intersection of the following sets:
  - $\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}})$
  - $\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q}) \cup ((\mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{P}})$
- $\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}} = \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}} \cap (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P})} \cup \overline{\mathsf{error}(\mathcal{S}_\mathcal{Q})})$. ◇

The usual compositionality results can now be presented, which show that disjunction really is the join operator for the refinement preorder in the timed assume-guarantee framework.

Theorem 7.21. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{Q}$, and $\mathcal{S}'_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q}$ be contracts composable for disjunction. Then:

- $\mathcal{S}_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}$
- $\mathcal{S}_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{R}$ and $\mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{R}$ implies $\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{R}$
- $\mathcal{S}'_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P}$ and $\mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{Q}$ implies $\mathcal{S}'_\mathcal{P} \vee_{nt} \mathcal{S}'_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}$.

Proof. For the first claim of $\mathcal{S}_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}$, we first show that $\mathsf{error}(\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}}) \subseteq \mathsf{error}(\mathcal{S}_\mathcal{P})$. So let X_i be the i-th iteration of $\mathsf{error}(\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q})$ being defined as a least fixed point. Then, by induction on i, we show that $X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}}) \subseteq \mathsf{error}(\mathcal{S}_\mathcal{P})$. When $i = 0$, the result holds trivially, since $X_i = \emptyset$. So for the inductive case, suppose that $t \in X_{k+1} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}})$. Then either (i) $t \in \mathsf{violations}(\mathcal{AG}(\mathcal{P} \vee_{nt} \mathcal{Q}))$, (ii) there exists $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}^\infty_0$ such that $tt' \in X_k$, and for each $t'' \leq t'$ and $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$ it holds that $tt''o \in X_k$. If (i) holds, then there is a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}}}$. Hence $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ and so $t \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ as required. If instead (ii) holds, then since $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{P}})^*$, it follows by the induction hypothesis that $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$, and so $t \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Finally, if (iii) holds, then by the induction hypothesis we derive $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ and $tt''o \in \mathsf{error}(\mathcal{S}_\mathcal{P})$ for each $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{P}}$, given that $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}}$. Consequently, $t \in \mathsf{error}(\mathcal{S}_\mathcal{P})$. Now suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}})$. Then $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})$ by definition. Showing $\mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}$ is similar.

For the second claim, suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}})$. If $t \in \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}})$, then from $\mathcal{S}_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{R}$ it follows that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})$. If $t \notin \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}})$, then there exists a prefix $t'o$ of t with $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}})$ and $o \in \mathcal{A}^O_\mathcal{Q} \setminus \mathcal{A}_\mathcal{P}$. By the induction hypothesis on the strictly shorter trace t', we derive $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})$ and $t'o \in (\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$, itself implying $t \in (\mathcal{R}_{\mathcal{S}_\mathcal{P}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P})) \uparrow \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$. A similar result holds on $\mathcal{S}_\mathcal{Q}$, and so $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}}$ as required.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{R}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}})$. Then there exists a smallest prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{R}} \cap \mathsf{error}(\mathcal{S}_\mathcal{R}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P} \vee \mathcal{S}_\mathcal{Q}})$. Suppose all strict prefixes of t' are not in $\mathsf{error}(\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q})$. Then by the previous part, it follows that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}}$. If $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}})$, then from $\mathcal{S}_\mathcal{P} \sqsubseteq_{nt} \mathcal{S}_\mathcal{R}$ it follows that $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P})$, and if $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{Q}})$, then from $\mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{R}$ it follows that $t' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Hence $t' \notin \mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}}$ (noting $\mathcal{G}_{\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q}} \subseteq \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}}) \cup \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{Q}})$), which implies $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q})$. By extension closure of error, we have $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \vee_{nt} \mathcal{S}_\mathcal{Q})$.

The third claim follows by the same reasoning as in Theorem 5.26. □

Theorem 7.22. Let $\mathcal{P}$ be a component, and let $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}$ be contracts such that $\mathcal{S}_1$ and $\mathcal{S}_2$ are composable for disjunction, and $\mathcal{A}^I_\mathcal{P} \cap \mathcal{A}^O_\mathcal{S} = \emptyset$. Then the following AG rule is both sound and complete:

$$\textsc{Timed-Disjunction}\ \frac{\mathcal{P} \models_{nt} \mathcal{S}_1 \quad \text{or} \quad \mathcal{P} \models_{nt} \mathcal{S}_2 \qquad \mathcal{S}_1 \vee_{nt} \mathcal{S}_2 \sqsubseteq_{nt} \mathcal{S}}{\mathcal{P} \models_{nt} \mathcal{S}}.$$

Proof. Follows by the argument in Theorem 5.27, having replaced the reference to Theorem 5.26 with Theorem 7.21. □

7.6 Quotient

In this section, we formulate the definition of quotient on timed contracts, which can be used to decompose a system-wide contract, given a subcontract fulfilling part of that system. The contract produced by the quotient thus stipulates the remaining portion of the system that needs to be implemented. More formally, the quotient is the adjoint of the parallel operator under contract-based refinement. The definition is largely unchanged from Definition 5.28, although note that the lifting operations, as well as the error sets, have been redefined.

Definition 7.23. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts. Then the quotient $\mathcal{S}_\mathcal{W} \ /_{nt} \ \mathcal{S}_\mathcal{P}$ is a contract $\langle \mathcal{A}^I_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}, \mathcal{A}^O_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}, \mathcal{R}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}}, \mathcal{G}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}} \rangle$, defined only when $\mathcal{A}^O_{\mathcal{S}_\mathcal{P}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{W}}$, where:

- $\mathcal{R}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}} = [\mathcal{R}_{\mathcal{S}_\mathcal{W}} \cap (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{W}}})] \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0)$
- $\mathcal{G}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}}$ is the largest subset of $\mathcal{R}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}}$ disjoint from $[\mathcal{R}_{\mathcal{S}_\mathcal{W}} \cap (\overline{\mathsf{error}(\mathcal{S}_\mathcal{P}) \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{W}}}) \cap (\mathsf{error}(\mathcal{S}_\mathcal{W}) \cup (\overline{\mathcal{R}_{\mathcal{S}_\mathcal{P}} \Uparrow \mathcal{A}_{\mathcal{S}_\mathcal{W}}}))] \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0)$. ◇

The intuition behind this definition is completely unchanged from that for Definition 5.28, having updated references to violations with error. The next theorem shows that our definition satisfies the characteristic properties of quotient, in that it yields the weakest decomposition of a contract.

Theorem 7.24. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts. Then there exists a contract $\mathcal{S}_\mathcal{Q}$ such that $\mathcal{S}_\mathcal{P} \ ||_{nt} \ \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{W}$ iff the following properties hold:

- The quotient $\mathcal{S}_\mathcal{W} \ /_{nt} \ \mathcal{S}_\mathcal{P}$ is defined
- $\mathcal{S}_\mathcal{P} \ ||_{nt} \ (\mathcal{S}_\mathcal{W} \ /_{nt} \ \mathcal{S}_\mathcal{P}) \sqsubseteq_{nt} \mathcal{S}_\mathcal{W}$
- $\mathcal{A}^I_{\mathcal{S}_\mathcal{Q}} = \mathcal{A}^I_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$ implies $\mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{W} \ /_{nt} \ \mathcal{S}_\mathcal{P}$.

Proof. The first claim follows by the reasoning in Theorem 5.29. The remaining proofs, which we reproduce below, are largely unchanged from the proofs contained in Theorem 5.60, excepting the revised definition of error.

For the second claim, suppose $t \in \mathcal{R}_{\mathcal{S}_\mathcal{W}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})})$. If $t \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}(\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P})}$, then there exists a prefix t' of t and $t'' \in \mathcal{T}(\mathcal{A}^O_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})})$ such that $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$ or $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \notin \mathcal{R}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}}$, and $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$ and $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \notin \mathsf{error}(\mathcal{S}_\mathcal{W} \ /_{nt} \ \mathcal{S}_\mathcal{P})$. It follows that $t't'' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$, so $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}}$, which means $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$. Therefore, $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \notin \mathcal{G}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}}$, which implies $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \in \mathsf{violations}(\mathcal{S}_\mathcal{W} \ /_{nt} \ \mathcal{S}_\mathcal{P})$. But this contradicts $t't'' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}^\infty_0) \notin \mathsf{error}(\mathcal{S}_\mathcal{W} \ /_{nt} \ \mathcal{S}_\mathcal{P})$. Hence $t \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}(\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P})}$.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{W}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{P}||(\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P})})$. Then, there exists a prefix t' of t such that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}} \cap \mathsf{error}(\mathcal{S}_\mathcal{W})$. By the previous part, it follows that $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}(\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P})}$. Now suppose for a contradiction that $t' \in \mathcal{G}_{\mathcal{S}_\mathcal{P}||_{nt}(\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P})}$. Then $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$ and $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathsf{error}(\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P})$. But it follows that $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathsf{violations}(\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P})$, since $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}} \cap \overline{\mathcal{G}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}}}$. This contradicts $t' \in \mathcal{G}_{\mathcal{S}_\mathcal{P}||_{nt}(\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P})}$. Hence $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_{nt}\; (\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P}))$ and so $t \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_{nt}\; (\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P}))$.

For the third claim, suppose that $t \in \mathcal{R}_{\mathcal{S}_\mathcal{W}/_{nt}\mathcal{S}_\mathcal{P}} \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{Q}})$. Then there exists $t' \in \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{W}})$ such that $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) = t$ with $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ and $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$. From $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ we derive $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_{nt}\; \mathcal{S}_\mathcal{Q})$, given that $\mathcal{S}_\mathcal{P} \;||_{nt}\; \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{W}$. If $t' \in \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}}$, then it follows that $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathcal{R}_{\mathcal{S}_\mathcal{Q}} \cup \mathsf{error}(\mathcal{S}_\mathcal{Q})$. If instead $t' \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_{nt}\; \mathcal{S}_\mathcal{Q})$, then it follows that $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$ by Lemma 7.13. Note that $t' \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) = t$.

Now suppose that $t \in \mathsf{error}(\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P}) \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{Q}})$. We show that $X_i \cap \mathcal{T}(\mathcal{A}_{\mathcal{S}_\mathcal{Q}}) \subseteq \mathsf{error}(\mathcal{S}_\mathcal{Q})$ by induction on i, where X_i is the i-th iteration of defining $\mathsf{error}(\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P})$ as a least fixed point. The case of $i = 0$ is trivial, since $X_0 = \emptyset$. For the difficult case of $t \in X_{k+1}$, either: (i) $t \in \mathsf{violations}(\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P})$, (ii) there exists $t' \in (\mathcal{A}^I_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}})^*$ such that $tt' \in X_k$, or (iii) there exists $t' \in \mathbb{R}_0^\infty$ such that $tt' \in X_k$ and for each $t'' \leq t'$ and $o \in \mathcal{A}^O_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$ it holds that $tt''o \in X_k$. For (i), there is a prefix and input extension t' of t such that there exists $t_w \in \mathcal{R}_{\mathcal{S}_\mathcal{W}}$ with $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) = t'$, $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathsf{error}(\mathcal{S}_\mathcal{P})$, and either $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{W})$ or $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$. If $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{W})$, then $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_{nt}\; \mathcal{S}_\mathcal{Q})$, since $\mathcal{S}_\mathcal{P} \;||_{nt}\; \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{W}$. By Lemma 7.13, it follows that $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Alternatively, if $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{P}} \cup \mathbb{R}_0^\infty) \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}}$, then if $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \notin \mathsf{error}(\mathcal{S}_\mathcal{Q})$ it follows that $t_w \notin \mathcal{R}_{\mathcal{S}_\mathcal{P}||_{nt}\mathcal{S}_\mathcal{Q}}$. Since $\mathcal{S}_\mathcal{P} \;||_{nt}\; \mathcal{S}_\mathcal{Q} \sqsubseteq_{nt} \mathcal{S}_\mathcal{W}$, it must hold that $t_w \in \mathsf{error}(\mathcal{S}_\mathcal{P} \;||_{nt}\; \mathcal{S}_\mathcal{Q})$, which again by Lemma 7.13 implies $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Note that $t_w \upharpoonright (\mathcal{A}_{\mathcal{S}_\mathcal{Q}} \cup \mathbb{R}_0^\infty) = t'$, so $t \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. For (ii), by the induction hypothesis we know that $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$, given that $t' \in \mathcal{A}^*_{\mathcal{S}_\mathcal{Q}}$. Hence $t \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$. Finally for (iii), by the induction hypothesis we derive $tt' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$, and $tt''o' \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$ for each $o' \in \mathcal{A}^O_{\mathcal{S}_\mathcal{Q}}$, given that $\mathcal{A}^O_{\mathcal{S}_\mathcal{Q}} \subseteq \mathcal{A}^O_{\mathcal{S}_\mathcal{W}/\mathcal{S}_\mathcal{P}}$. Hence, $t \in \mathsf{error}(\mathcal{S}_\mathcal{Q})$ as required. □

Parameterisation of the input set for the timed quotient is applicable just as in the safety setting. Based on these properties of quotient, we can formulate a sound and complete AG rule, closely mirroring the rule of Theorem 5.30.

Theorem 7.25. Let $\mathcal{S}_\mathcal{P}$ and $\mathcal{S}_\mathcal{W}$ be contracts such that $\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P}$ is defined, let $\mathcal{P}$ range over components having the same interface as $\mathcal{S}_\mathcal{P}$, and let $\mathcal{Q}$ be a component having the same interface as $\mathcal{S}_\mathcal{W} \;/_{nt}\; \mathcal{S}_\mathcal{P}$ (where the quotient is parameterised on the set $\mathcal{A}^I_\mathcal{Q}$). Then the following AG rule is

both sound and complete:

$$\textsc{Timed-Quotient}\frac{\forall \mathcal{P} \cdot \mathcal{P} \models_{nt} \mathcal{S}_{\mathcal{P}} \text{ implies } \mathcal{P} \mid\mid_{nt} \mathcal{Q} \models_{nt} \mathcal{S}_{\mathcal{W}}}{\mathcal{Q} \models_{nt} \mathcal{S}_{\mathcal{W}} \mathbin{/_{nt}} \mathcal{S}_{\mathcal{P}}}.$$

Proof. Follows by straightforward modification to Theorem 5.30, having updated concepts to the timed equivalents. □

As in Theorem 5.30, we insist that the components $\mathcal{P}$ and $\mathcal{Q}$ must have the same interfaces as their respective contracts, since parallel composition is only monotonic when restrictions are placed on the interfaces of the contracts to be composed (cf Theorem 7.14). Furthermore, the rule can be reformulated so as to avoid the universal quantification by considering the least refined implementation $\mathcal{I}_{nt}(\mathcal{S}_{\mathcal{P}})$ of $\mathcal{S}_{\mathcal{P}}$.

Corollary 7.26. Let $\mathcal{S}_{\mathcal{P}}$ and $\mathcal{S}_{\mathcal{W}}$ be contracts such that $\mathcal{S}_{\mathcal{W}} \mathbin{/_{nt}} \mathcal{S}_{\mathcal{P}}$ is defined, and let $\mathcal{Q}$ be a component having the same interface as $\mathcal{S}_{\mathcal{W}} \mathbin{/_{nt}} \mathcal{S}_{\mathcal{P}}$ (where the quotient is parameterised on the set $\mathcal{A}^I_{\mathcal{Q}}$). Then the following AG rule is both sound and complete:

$$\textsc{Timed-Quotient-Revised}\frac{\mathcal{I}_{nt}(\mathcal{S}_{\mathcal{P}}) \mid\mid_{nt} \mathcal{Q} \models_{nt} \mathcal{S}_{\mathcal{W}}}{\mathcal{Q} \models_{nt} \mathcal{S}_{\mathcal{W}} \mathbin{/_{nt}} \mathcal{S}_{\mathcal{P}}}.$$

Proof. Unchanged from Corollary 5.31, having updated references. □

7.7 Decomposing Parallel Composition

The following corollary shows how we can revise the AG rule for parallel composition so that it makes use of quotient on contracts. This is useful for system development, as we will often have the specification of a whole system, rather than the specifications of the subsystems to be composed.

Corollary 7.27. Let $\mathcal{P}$ and $\mathcal{Q}$ be components, and let $\mathcal{S}_{\mathcal{P}}$, $\mathcal{S}_{\mathcal{Q}}$ and $\mathcal{S}$ be contracts such that $\mathcal{A}_{\mathcal{P}} \cap \mathcal{A}_{\mathcal{Q}} \cap \mathcal{A}_{\mathcal{S}_{\mathcal{P}} || \mathcal{S}_{\mathcal{Q}}} \subseteq \mathcal{A}_{\mathcal{S}_{\mathcal{P}}} \cap \mathcal{A}_{\mathcal{S}_{\mathcal{Q}}}$ and $\mathcal{A}^I_{\mathcal{P}||\mathcal{Q}} \cap \mathcal{A}^O_{\mathcal{S}} = \emptyset$. When the quotient is parameterised on $\mathcal{A}^I_{\mathcal{S}_{\mathcal{Q}}}$, the following rule is both sound and complete:

$$\textsc{Timed-Parallel-Decompose}\frac{\mathcal{P} \models_{nt} \mathcal{S}_{\mathcal{P}} \qquad \mathcal{Q} \models_{nt} \mathcal{S}_{\mathcal{Q}} \qquad \mathcal{S}_{\mathcal{Q}} \sqsubseteq_{nt} \mathcal{S} \mathbin{/_{nt}} \mathcal{S}_{\mathcal{P}}}{\mathcal{P} \mid\mid_{nt} \mathcal{Q} \models_{nt} \mathcal{S}}.$$

Proof. Follows immediately from Theorems 7.15 and 7.24. □

This rule, based on Theorem 7.15, differs in having the premise $\mathcal{S}_{\mathcal{Q}} \sqsubseteq_{nt} \mathcal{S} \mathbin{/_{nt}} \mathcal{S}_{\mathcal{P}}$ in place of $\mathcal{S}_{\mathcal{P}} \mid\mid_{nt} \mathcal{S}_{\mathcal{Q}} \sqsubseteq_{nt} \mathcal{S}$. Note that this substitution requires no change to the constraints on the contracts and components. The rule is useful for scenarios when the contract $\mathcal{S}$ is supplied along with a subcontract $\mathcal{S}_{\mathcal{P}}$ (or for when a subcontract $\mathcal{S}_{\mathcal{P}}$ can easily be inferred). In such circumstances, the missing contract $\mathcal{S}_{\mathcal{Q}}$ can be taken as any refinement of $\mathcal{S} \mathbin{/_{nt}} \mathcal{S}_{\mathcal{P}}$.

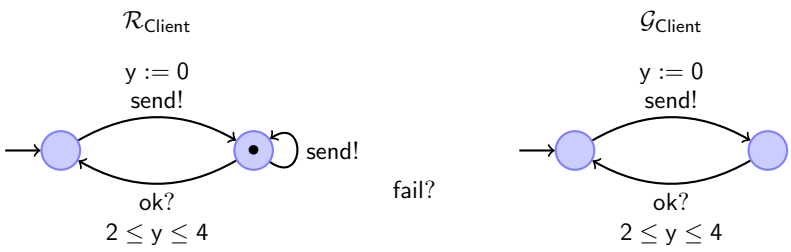

Figure 7.1: Assumption and guarantee of Client

7.8 Case Study

Building on the case study introduced in Section 5.3, we introduce timing constraints to demonstrate the features of our timed assume-guarantee framework in practice. In the pictorial representation, assumptions and guarantees are modelled by timed operational components (Definition 6.3) that are not required to satisfy the well-formedness condition at the end of Definition 6.3, and whose co-invariants are always true in the guarantee.

The semantics of an assumption R and a guarantee G are given in terms of the timed transition systems $[\![\mathsf{R}]\!]$ and $[\![\mathsf{G}]\!]$ respectively, as presented in Definition 6.4. Based on these operational (infinite-state) models, we can extract the assumption and guarantee traces in accordance with the definition below. Note that the initial states of $[\![\mathsf{R}]\!]$ and $[\![\mathsf{G}]\!]$ cannot be $\perp$ by definition.

Definition 7.28. The assumption $[\![\mathsf{R}]\!]^*$ and guarantee $[\![\mathsf{G}]\!]^*$ represented by R and G are:

- $[\![\mathsf{R}]\!]^* = \{t \in \mathcal{T}(\mathcal{A}_\mathsf{P}) : \text{there is an accepting run over } \%t \text{ in } [\![\mathsf{R}]\!] \text{ not encountering } \perp\} \cdot \mathcal{T}(\mathcal{A}_\mathsf{P}^O)$
- $[\![\mathsf{G}]\!]^* = \{t \in \mathcal{T}(\mathcal{A}_\mathsf{P}) : \text{there is an accepting run over } \%t \text{ in } [\![\mathsf{G}]\!] \text{ not encountering } \perp\}.$ ◇

Recall that the case study in Section 5.3 is concerned with a link layer protocol drawn from distributed systems, which is a variant of the running example used in [LNW06]. A Client (see Figure 7.1) can communicate with a Server (Figure 7.2) by sending data, and can observe whether the transmission was ok or whether it failed. The Server, on the other hand, is an intermediary between the Client and a Database server. It receives data from the Client via the send interaction, and then transmits it to the Database engine via some communication medium, after which it waits for positive or negative confirmation that the data has been written into the database, in the form of ack and nack signals, respectively. In the case that the transmission is acknowledged, the Server indicates to the Client that all is ok. Otherwise, if nack is received from the Database, the Server attempts to retransmit, and if nack is received for a second time in succession, the Server will signify to the Client that a failure has occurred. The model of the Database itself is irrelevant.

The contract for the Client in Figure 7.1 assumes that a transmission will only be confirmed as ok between two and four time units of the send action being sent after the last ok. Based on

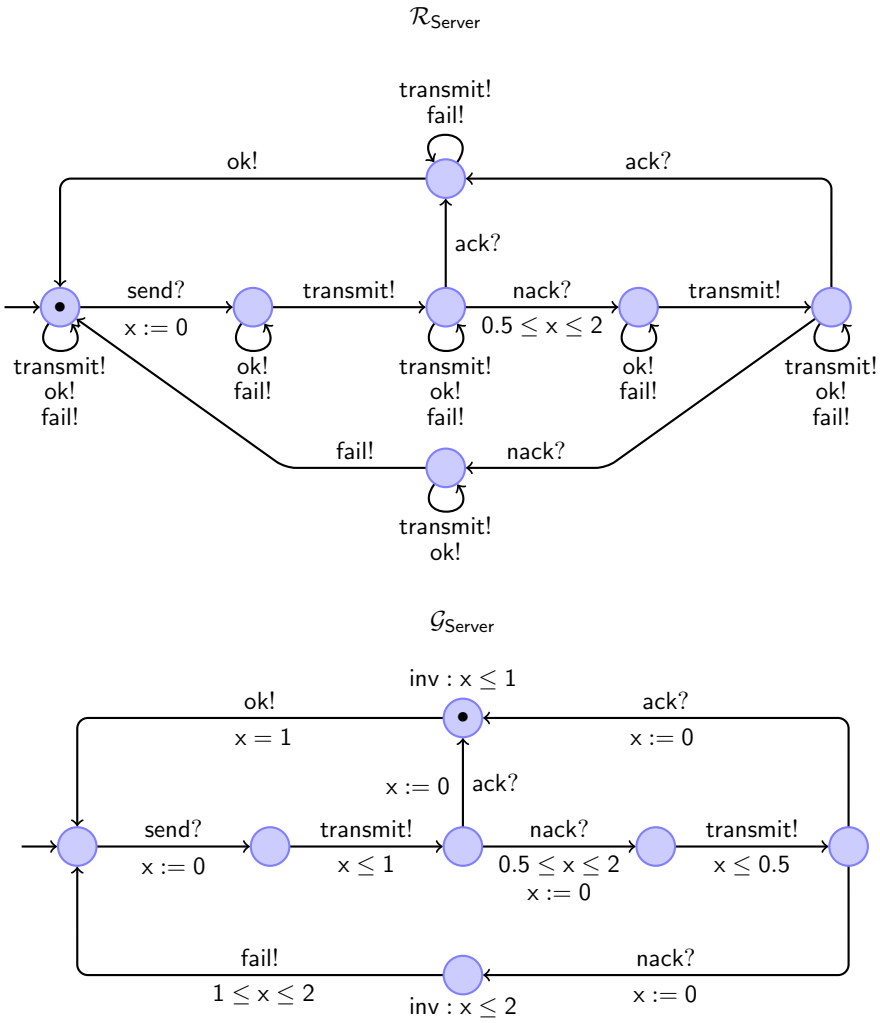

Figure 7.2: Assumption and guarantee of Server

the guarantee, an implementation of Client is obliged not to send multiple times without receiving confirmation that all is ok between successive attempts (since such behaviour is included in the assumption by the timed output extensions, while it is not in the guarantee, thus it is in error(Client)). Furthermore, note that an implementation is prohibited from sending an infinite number of times, since this is an allowable behaviour in the assumption, but not in the guarantee, due to the lack of a Büchi accepting state in the latter. Consequently, the most general implementation of Client is a timed component whose pictorial representation is the same as $\mathcal{G}_{\mathsf{Client}}$. Note that this is a well-formed timed component in the sense of Definitions 6.3 and 6.33.

We now consider the contract for the Server, shown in Figure 7.2. The assumption assumes that the temporal ordering of any sequence of interactions with the environment ending in one of send, ack and nack must be an explicit trace in $\mathcal{R}_{\mathsf{Server}}$, and any infinite trace through $\mathcal{R}_{\mathsf{Server}}$ must also be accepting. Note that the assumption places no restrictions on the timing constraints associated with these interactions, except that the first nack following a send must occur between

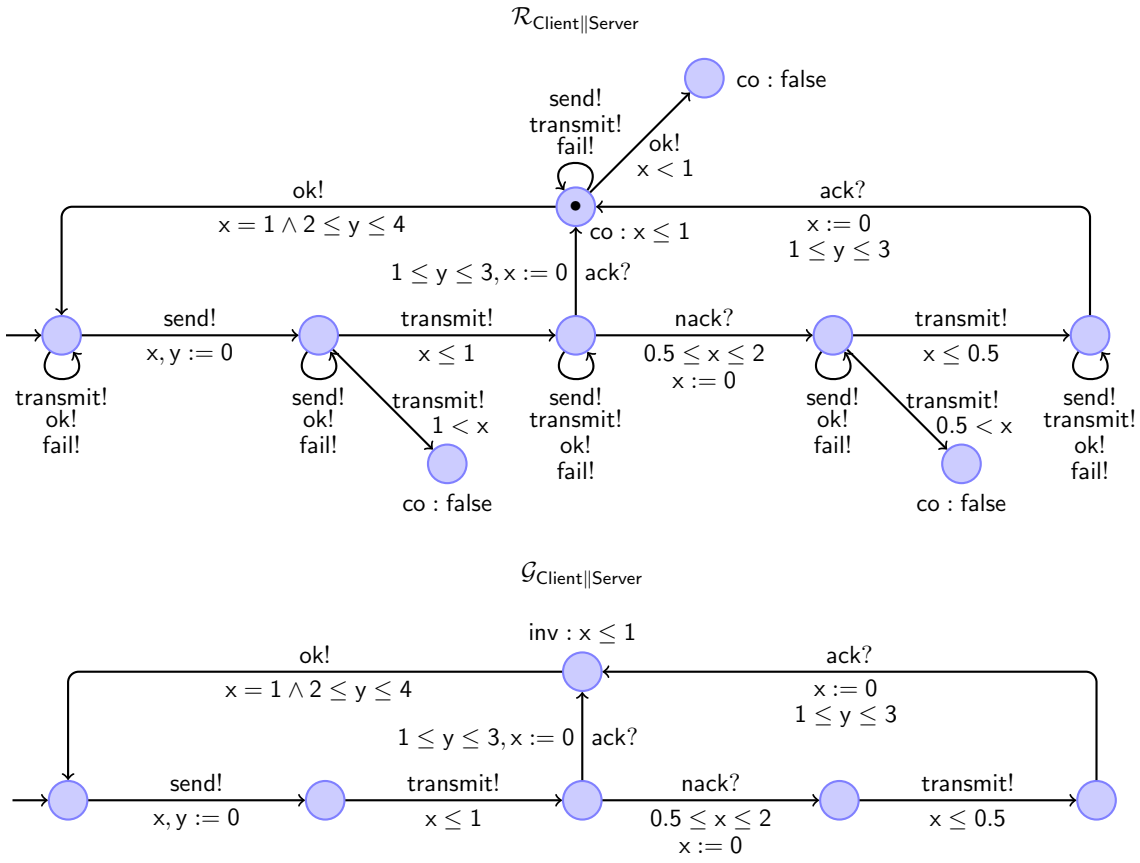

Figure 7.3: Assumption and guarantee of Client || Server

half and 2 time units after the send. Naturally, the guarantee is more permissive on the allowable behaviours. After receiving send, an implementation is permitted to transmit within 1 time unit, although it is not obliged to do so, since the invariant is true on the state between the two interactions, meaning that an implementation can sojourn there for an unbounded amount of time. After transmission, an implementation can wait an unbounded amount of time for an ack, or can receive a nack within half and 2 time units. If an ack is received, then an implementation is required to confirm that all is ok precisely 1 time unit later (imposed by the invariant). The other behaviours can be understood similarly. Note that a transmission is not permitted to fail successively an infinite number of times. As for the Client, the most general implementation of Server is a timed component represented by $\mathcal{G}_{\mathsf{Server}}$, since the behaviour of the guarantee is strictly contained within the assumption.

The contract representing the combined effect of Client and Server, that is Client || Server, is shown in Figure 7.3. After a nack following a transmit, the following transmit may not be followed by nack, otherwise unpredictable behaviour could ensue, due to the fact that Client does not specify the behaviour of fail. Similarly, Client does not specify the behaviour of ok when the clock y does not satisfy $2 \leq y \leq 4$. Consequently, these timing constraints must be propagated

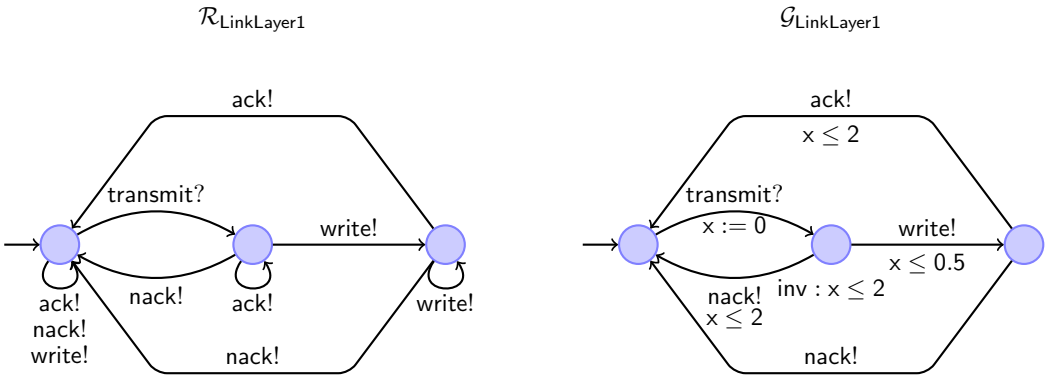

Figure 7.4: Assumption and guarantee of LinkLayer1

backwards in the parallel composition. Any implementation that reaches the state containing $\bullet$ must remain in that state for exactly 1 time unit, hence why the timing constraints in the preceding transitions are appended with $1 \leq y \leq 3$. Note that the states marked with co : false are chaotic, so time may continue unbounded and any interaction may be performed. Any trace reaching such a chaotic state is automatically contained within error(Client || Server), since such traces are not in the guarantee. The invariant $x \leq 1$ in the guarantee becomes a co-invariant in the assumption, because no implementation of the contract can allow time to pass 1 time unit in the state labelled by $\bullet$, hence subsequent behaviours are unspecified and should be in the error set.

We now wish to consider the behaviour of the communication link between the Server and Database. An abstract protocol is provided in Figure 7.4, which ensures that after each transmit, a nack will be provided within 2 time units, or a data write response will be performed within half a time unit (both imposed by the invariant $x \leq 2$). After writing of the data, a nack can be sent at any time, an ack can be provided within 2 time units of the transmit request, or the implementation can hang. Such a contract is not compatible with the system composed of the Client and Server, because the protocol allows multiple successive nack responses. To circumvent this problem, we find the most general restrictions that need to be applied to the protocol, by first computing ErrorFree/(Client || Server), which is the most general specification of a contract that can interact with Client || Server without introducing safety or bounded-liveness errors. The conjunction of this contract can then be taken with the link layer protocol.

The contract ErrorFree that ensures no safety or bounded-liveness errors occur remains as depicted in Figure 5.13. Note that the invariant on the sole state is true, and the co-invariant is false. Figure 7.5 shows the quotient ErrorFree/(Client || Server) when the input set for the quotient operation is taken to be $\{\mathsf{send}, \mathsf{transmit}, \mathsf{ok}\}$. Certainly, the guarantee is unchanged from the guarantee of the parallel composition, excepting the addition of write self-loops since these correspond with an independent action, and interchanging of some action types. The assumption for the quotient is also similar to that for the parallel composition, after accounting for the inclusion of

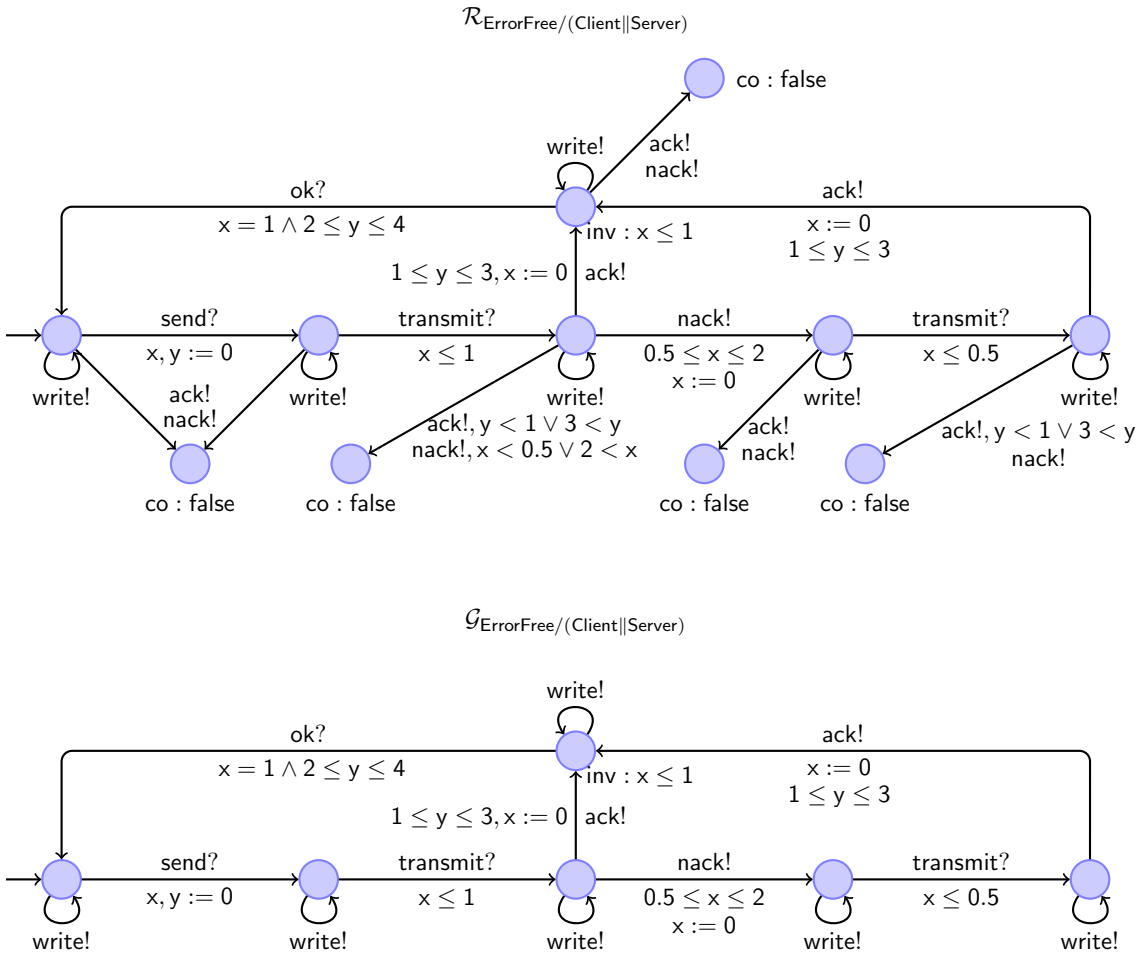

Figure 7.5: Assumption and guarantee of ErrorFree/(Client || Server) with full interface

unspecified traces of Client || Server that are not in error(Client || Server). However, note that the co-invariant in $\mathcal{R}_{\mathsf{Client}||\mathsf{Server}}$ has become an invariant in $\mathcal{R}_{\mathsf{ErrorFree}/(\mathsf{Client}||\mathsf{Server})}$. Since the invariant takes precedence over the co-invariant in the composition, this prevents the bounded-liveness error in Client || Server manifesting itself in (ErrorFree/(Client || Server)) || (Client || Server).

The contract ErrorFree/(Client || Server), when the input set for the quotient operation is taken to be {transmit} (referred to as LinkLayer2), is depicted in Figure 7.6, and is obtained from Figure 7.5 by performing timed projections. Note how, in the assumption, non-determinism can occur on the actions. For example, in the right-most state the upper ack transition overlaps with the lower one for certain time values. This is a consequence of losing track of when the newly hidden interactions occur. Such overlaps are eradicated in the guarantee, by restricting to transitions that are guaranteed to be safe under all resolutions of non-determinism.

As a final step, we compute the conjunction of the abstract protocol LinkLayer1 along with the most general restrictions needing to be applied (i.e., LinkLayer2) so that the Client and Server can communicate with the Database in a safe and responsive manner. The assumption of the

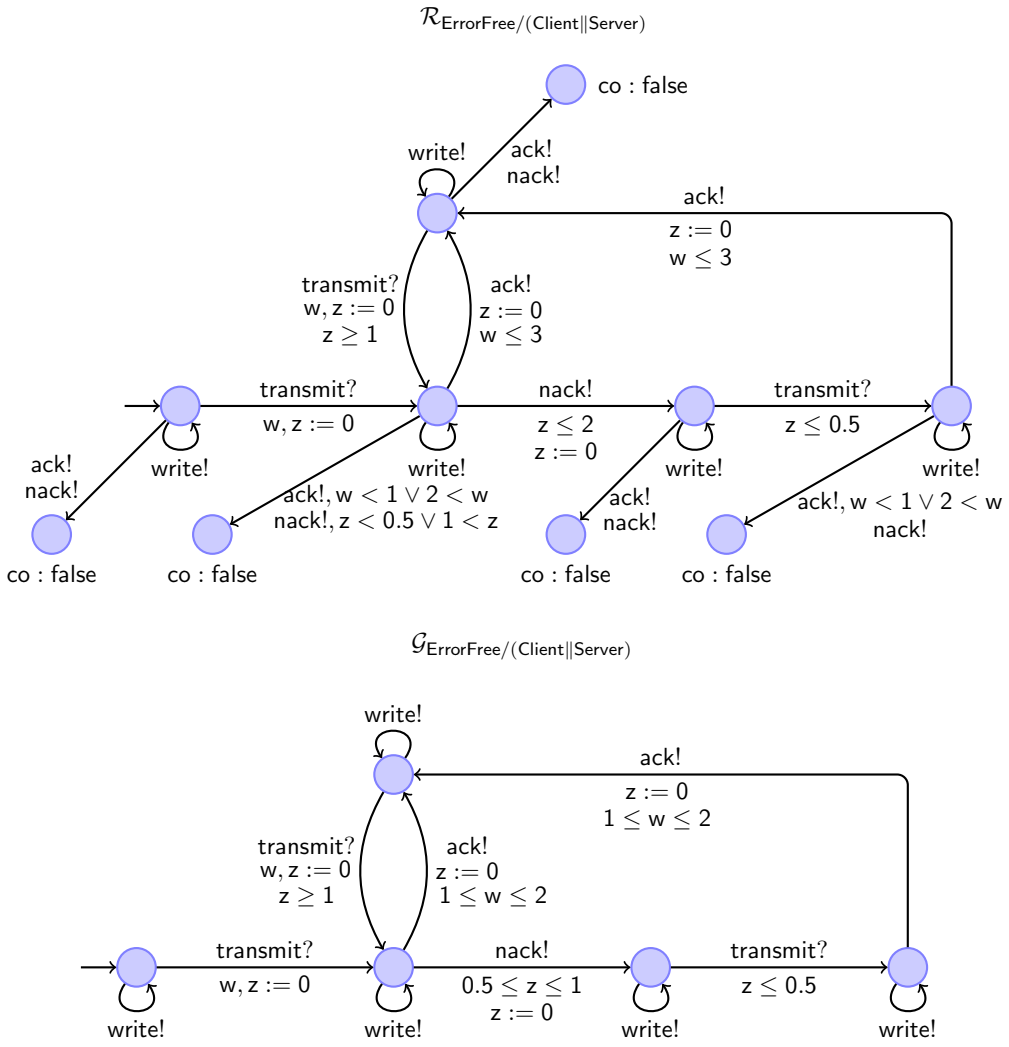

Figure 7.6: Assumption and guarantee of ErrorFree/(Client || Server) with restricted interface

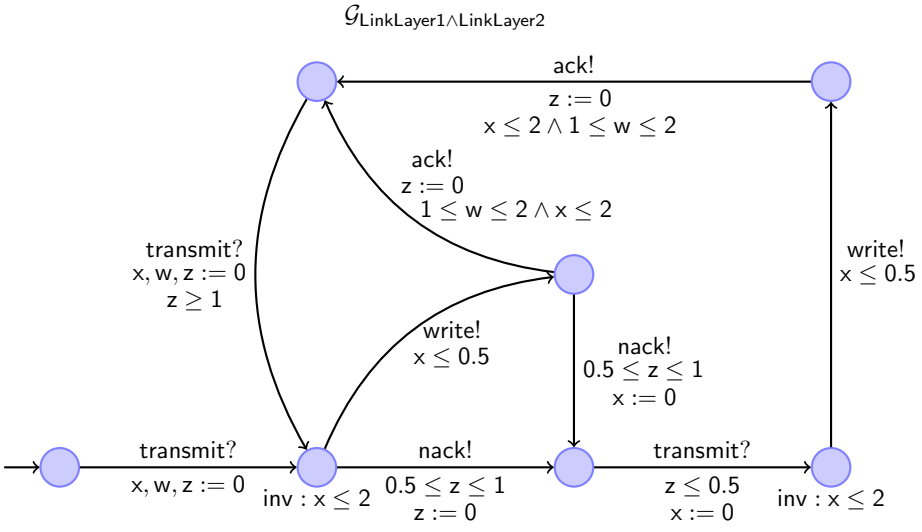

Figure 7.7: Guarantee of LinkLayer1 ∧ LinkLayer2

contract LinkLayer1 $\wedge$ LinkLayer2 is defined as the union of $\mathcal{R}_{\mathsf{LinkLayer1}}$ and $\mathcal{R}_{\mathsf{LinkLayer2}}$, and so is not depicted. The guarantee, on the other hand, is essentially the intersection of the behaviours from the guarantees of LinkLayer1 and LinkLayer2 respectively, and is represented in Figure 7.7. The structure of this guarantee closely matches that in Figure 5.16 for the untimed setting.

This contract-based reasoning methodology has allowed us to reason that any implementation of LinkLayer1 $\wedge$ LinkLayer2 can be placed in parallel with implementations of Client and Server so that the system as a whole will not generate any safety or bounded liveness errors.

7.9 Summary

This chapter has introduced an assume-guarantee framework for reasoning compositionally about the non-terminating timed components introduced in Section 6.3. The framework is a natural extension of the untimed theory in Chapter 5, by combining unique features from both the safety and progress frameworks, so as to capture real-time behaviours. As in the untimed setting, the operations of parallel composition, conjunction, disjunction and quotient are defined directly on contracts, and a range of compositionality results are provided for the operators with respect to the linear-time refinement preorder corresponding to implementation containment.

Under the premise that assumptions and guarantees are represented by arbitrary timed automata, refinement checking is undecidable [AD94]. However, if $\mathcal{S}_{\mathcal{P}}$ is represented by a deterministic timed automaton, then checking of $\mathcal{S}_{\mathcal{Q}} \sqsubseteq_{nt} \mathcal{S}_{\mathcal{P}}$ is PSPACE-complete. Additionally, Ouaknine and Worrell [OW04] show that language inclusion is decidable when $\mathcal{S}_{\mathcal{P}}$ has one clock and only finite-length words, but there is no primitive recursive bound on the complexity in this setting [ADOW05]. However, tool support provided by ECDAR [DLL$^+$10a] for the timed specification theory in [DLL$^+$10b] suggests that the high complexity associated with refinement checking is not a limitation in practice.

CHAPTER

EIGHT

Conclusion

This thesis has presented formalisms for modelling and reasoning about the interactions arising in component-based systems, when communication is asynchronous and non-blocking. An overarching aim has been to provide support for modelling protocols, distributed systems and asynchronous hardware designs, where an environment supporting handshaking is either an unrealistic expectation, or must be achieved in a domain that is inherently asynchronous. The frameworks provide modelling notations capable of capturing the essential behaviour of components (both the temporal ordering of interactions and, later, real-time constraints on the occurrence of interactions) in order to determine and reason about the communication mismatches (along with run-time errors and underspecification) that can arise through asynchrony.

In all cases, the frameworks take the form of specification theories, by providing modelling formalisms along with refinement relations for identifying when a component can be used in place of another without violating key properties, such as safety and progress, together with a rich collection of compositional operators for building new complex components out of pre-existing subsystems. The range of compositional operators allows for large-scale system development, where components must be developed independently of one another, and incrementally adapted, based on evolving system requirements.

It has been demonstrated that the operators enjoy strong algebraic properties with respect to the refinement preorders, such as monotonicity, as well as conjunction and disjunction characterising the meet and join operations, and quotient being the adjoint of parallel. Where appropriate, operators are shown to be associative, commutative and idempotent. In each framework, the linear-time refinement relation is shown to be the weakest preorder characterising substitutivity, whilst maintaining progress (when stipulated). Based on these results, the equivalence defined as mutual refinement is shown to be fully abstract with respect to the operators of the specification theory.

The choice of which specification theory to use should be based on the properties needing to be ascertained by the user. The original theory suffices for ensuring communication mismatches cannot arise during system development, while the progress-sensitive framework ensures that a component must always make progress whenever dictated to do so by the specification. For the real-time frameworks, the choice between the terminating and non-terminating theory is dependent on whether a system is supposed to run indefinitely, or whether it is safe for the system to

halt at some point. The trace-based representations of components provide essential information for reasoning about their behaviour and interactions, whereas users can use the operational representation, which is closer to actual implementations.

The contract-based assume-guarantee frameworks can be used in one of two ways. First, they can be thought of as frameworks for component specifications, where components are modelled in one of the aforementioned specification theories. This allows for sound and complete assume-guarantee rules to be used for checking the preservation of safety and progress properties in component-based systems. The second approach is to treat the assume-guarantee framework as an alternative representation of components, that decomposes the assumptions placed on the environment from the guarantees made by the component. Under this arrangement, assumptions can be manipulated separately from guarantees, which supports a different way of thinking about the development cycle. Under both interpretations, component implementations can be obtained by use of the inference definition (Definition 5.6, for safety).

Under the assumptions of determinacy and regularity, refinement checking and application of the compositional operators, which are based on set containment and simple trace/set-theoretic operations respectively, can be performed in polynomial time. When dropping determinacy, the theory becomes PSPACE-complete, as in CSP [Ros10]. Given the set-theoretic nature of the constructions and refinement check, it is relatively straightforward to implement the theory as a tool.

Practical applicability of the substitutive specification theory has been demonstrated by Inverardi and Tivoli [IT13] in modelling protocol mediation patterns, while the progress-sensitive framework has been used for automatic mediator synthesis in [BCIJ13]. Small scale case studies throughout the dissertation show how the theories may be used in practice.

8.1 Future Work

There are a number of natural extensions that can be considered for the frameworks presented in this dissertation, ranging from automation, through to providing new frameworks capable of capturing different types of component behaviour.

- **Tool support.** An obvious strand of work concerns the development of a prototype tool for checking refinement and for automatically building components, given a machine-readable syntax for describing components and their compositions. Advantages over existing tools, such as FDR for CSP [Ros98], Ticc for interface automata [AdAS+06], and ECDAR for timed I/O systems [DLL+10a], is the rich collection of operators defined with respect to a refinement relation that is the weakest preorder preserving properties such as safety and progress. Given that the operations of the theories are defined in terms of set-theoretic operations along with projections and liftings, such an undertaking should be straightforward,

as it is essentially a marginally more complex version of the trace semantics from CSP. A timed theory would be slightly more involved, as a symbolic representation would be required for representing the timing behaviours. Techniques used for verification of timed automata would be relevant here, such as those discussed in [CDF+05] that recourse to the underlying zones of the automaton, rather than its regions, and are used as part of the Uppaal-Tiga tool [BDL04].

- **Liveness.** The specification theories of Chapter 3, and the assume-guarantee frameworks of Chapter 5, provide support for reasoning about the preservation of safety and progress properties. However, the restriction to finite-length traces prevents a proper treatment of liveness, such as in [LT89]. By including infinite-length traces, liveness can be considered outright, although consideration must then be afforded to fairness [RV96, Seg97].

- **Divergence.** The progress-sensitive assume-guarantee framework represented a contract by an assumption, a guarantee and a set of liveness traces on which an implementing component must make progress. However, divergent traces were not considered, meaning that any behaviour of a contract not required to make progress may diverge in an implementation. Such behaviour is generally undesirable (cf. [Jos92, Jon94] where divergence is equated with inconsistency), so it should be possible to say when divergence may and may not occur, which can be achieved by including a set of permitted divergence traces within contracts. Furthermore, this is useful for defining the hiding operations, which were omitted from the assume-guarantee frameworks because hiding of outputs can introduce divergence.

- **Timed operational theory.** To match the operational theory of components in the untimed setting, an operational theory of timed components should be developed, matching the trace-based framework. Given the inclusion of finite- and infinite-length traces, and the fact that limit traces are not necessarily included, the operational representation would need to come with liveness conditions (encoded by, say, Büchi states [Büc62]) and would thus closely mirror the formulation of timed automata in [AD94], although the semantic interpretation would differ.

- **Probabilistic extension.** A further direction for future work considers developing a framework that captures the probabilistic behaviour of components. There are already a number of probabilistic specification theories [CDL+10, XGG10, DCL11, DKL+11], so it would be necessary to see what could be offered that is novel in this area. A theory based on traces, following the style of this dissertation, would be difficult, because probabilistic traces tend not to enjoy compositionality. Therefore, the model is likely to be operational, with refinement defined in terms of probabilistic simulation [JL91]. There is also a choice between reactive semantics, whereby interactions occur non-deterministically, with the successor state

being determined probabilistically, or the generative semantics, where a probability distribution exists over output actions that a component can offer (see [vGSS95] for a detailed comparison). A further extension would add rates to components, so that different weightings can be ascribed to the potence of the interactions offered by a component, so as to consider the continuous-time behaviour of component-based systems [WSS97]. Combining probability and time is a further possibility [KNSS02, KNPS06].

8.2 Concluding Remarks

This dissertation was written during a period of renewed interest in interface theories, both qualitative and quantitative in nature, as witnessed by the modernity of the citations. Far from being the final word on the topic, the dissertation shares similarities with a number of other theories developed during this time, and there are invariably a number of theories that have been completely overlooked. To our knowledge, the key novelty of our work has been to develop a linear-time substitutive theory of components based on the conceptually simple ideas of trace containment and set-theoretic operations, while supporting a rich collection of operators for system development. This has been demonstrated through the ease by which progress-sensitive and real-time extensions have been developed, and in the natural formulation of assume-guarantee reasoning frameworks, which can be automated.

Bibliography

[ACI+10] Marco Autili, Chris Chilton, Paola Inverardi, Marta Kwiatkowska, and Massimo Tivoli. Towards a Connector Algebra. In Tiziana Margaria and Bernhard Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation, Proc. 4th International Symposium on Leveraging Applications (ISoLA'10)*, volume 6416 of *Lecture Notes in Computer Science*, pages 278–292. Springer, 2010.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, April 1994.

[AdAS+06] B. Thomas Adler, Luca de Alfaro, Leandro Dias Silva, Marco Faella, Axel Legay, Vishwanath Raman, and Pritam Roy. TICC: A Tool for Interface Compatibility and Composition. In Thomas Ball and Robert B. Jones, editors, *Computer Aided Verification, Proc. 18th International Conference (CAV'06)*, volume 4144 of *Lecture Notes in Computer Science*, pages 59–62. Springer, 2006.

[ADOW05] Parosh Abdulla, Johann Deneux, Joël Ouaknine, and James Worrell. Decidability and Complexity Results for Timed Automata via Channel Machines. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming, Proc. 32nd International Colloquium (ICALP'05)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1089–1101. Springer, 2005.

[AENT03] Nina Amla, E. Allen Emerson, Kedar Namjoshi, and Richard Trefler. Abstract Patterns of Compositional Reasoning. In Roberto Amadio and Denis Lugiez, editors, *Concurrency Theory, Proc. 14th International Conference (CONCUR'03)*, volume 2761 of *Lecture Notes in Computer Science*, pages 431–445. Springer, 2003.

[AG97] Robert Allen and David Garlan. A formal basis for architectural connection. *ACM Transactions on Software Engineering and Methodology*, 6(3):213–249, July 1997.

[AHKV98] 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.

[AL93] Martín Abadi and Leslie Lamport. Composing specifications. *ACM Transactions on Programming Languages and Systems*, 15(1):73–132, January 1993.

[AL95] Martín Abadi and Leslie Lamport. Conjoining specifications. *ACM Transactions on Programming Languages and Systems*, 17(3):507–534, May 1995.

[AP93] Martín Abadi and Gordon D. Plotkin. A logical view of composition. *Theoretical Computer Science*, 114(1):3–30, June 1993.

[Arb04] Farhad Arbab. Reo: a channel-based coordination model for component composition. *Mathematical Structures in Computer Science*, 14(3):329–366, June 2004.

[AV10] 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.

[BCF+08] Albert Benveniste, Benoît Caillaud, Alberto Ferrari, Leonardo Mangeruca, Roberto Passerone, and Christos Sofronis. Multiple Viewpoint Contract-Based Specification and Design. In Frank S. de Boer, Marcello M. Bonsangue, Susanne Graf, and Willem P. de Roever, editors, *Formal Methods for Components and Objects, Proc. 6th International Symposium (FMCO'08)*, volume 5382 of *Lecture Notes in Computer Science*, pages 200–225. Springer, 2008.

[BCIJ13] 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.

[BCK11] Nikola Benes, Ivana Cerna, and Jan Kretinsky. Modal Transition Systems: Composition and LTL Model Checking. In Tevfik Bultan and Pao-Ann Hsiung, editors, *Automated Technology for Verification and Analysis, Proc. 9th International Symposium (ATVA'11)*, volume 6996 of *Lecture Notes in Computer Science*, pages 228–242. Springer, 2011.

[BCN+12] Albert Benveniste, Benoît Caillaud, Dejan Nickovic, Roberto Passerone, Jean-Baptiste Raclet, Philipp Reinkemeier, Alberto Sangiovanni-Vincentelli, Werner Damm, Thomas A. Henzinger, and Kim G. Larsen. Contracts for System Design. Technical Report RR-8147, INRIA, November 2012.

[BDH+12] Sebastian Bauer, Alexandre David, Rolf Hennicker, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. Moving from Specifications to Contracts in Component-Based Design. In Juan Lara and Andrea Zisman, editors, *Fundamental Approaches to Software Engineering, Proc. 15th International Conference (FASE'12)*, volume 7212 of *Lecture Notes in Computer Science*, pages 43–58. Springer, 2012.

[BDL04] Gerd Behrmann, Alexandre David, and Kim G. Larsen. A Tutorial on Uppaal. In Marco Bernardo and Flavio Corradini, editors, *Formal Methods for the Design of Real-Time Systems, International School on Formal Methods for the Design of Computer, Communication and Software Systems (SFM'04)*, volume 3185 of *Lecture Notes in Computer Science*, pages 200–236. Springer, 2004.

[BHGQ10] Imene Ben-Hafaiedh, Susanne Graf, and Sophie Quinton. Reasoning about Safety and Progress Using Contracts. In Jin Song Dong and Huibiao Zhu, editors, *Formal Methods and Software Engineering, Proc. 12th International Conference on Formal Engineering Methods (ICFEM'10)*, volume 6447 of *Lecture Notes in Computer Science*, pages 436–451. Springer, 2010.

[BHR84] 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.

[BLPR09] Nathalie Bertrand, Axel Legay, Sophie Pinchinat, and Jean-Baptiste Raclet. A Compositional Approach on Modal Specifications for Timed Systems. In Karin Breitman and Ana Cavalcanti, editors, *Formal Methods and Software Engineering, Proc. 11th International Conference on Formal Engineering Methods (ICFEM'09)*, volume 5885 of *Lecture Notes in Computer Science*, pages 679–697. Springer, 2009.

[BR08] Purandar Bhaduri and S. Ramesh. Interface synthesis and protocol conversion. *Formal Aspects of Computing*, 20(2):205–224, March 2008.

[BSAR06] Christel Baier, Marjan Sirjani, Farhad Arbab, and Jan Rutten. Modeling component connectors in Reo by constraint automata. *Science of Computer Programming*, 61(2):75–113, July 2006.

[Büc62] J. Richard Büchi. On a Decision Method in Restricted Second Order Arithmetic. In *Proc. International Congress on Logic, Method, and Philosophy of Science*, pages 1–12. Stanford University Press, 1962.

[BV12] Ferenc Bujtor and Walter Vogler. Interface automata with error states. Technical Report 2012-09, Institut fur Informatik, Universitat Augsburg, 2012.

[CCJK12] 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.

[CDF+05] Franck Cassez, Alexandre David, Emmanuel Fleury, Kim G. Larsen, and Didier Lime. Efficient on-the-fly algorithms for the analysis of timed games. In Martín Abadi and Luca de Alfaro, editors, *Concurrency Theory, Proc. 16th International Conference (CONCUR'05)*, volume 3653 of *Lecture Notes in Computer Science*, pages 66–80. Springer, 2005.

[CDL+10] Benoît Caillaud, Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wasowski. Compositional Design Methodology with Constraint Markov Chains. In *Quantitative Evaluation of Systems, Proc. 7th International Conference (QEST'10)*, pages 123–132. IEEE Computer Society, 2010.

[ČGL93] Kārlis Čerāns, Jens Chr. Godskesen, and Kim G. Larsen. Timed modal specification — Theory and tools. In Costas Courcoubetis, editor, *Computer Aided Verification, Proc. 5th International Conference (CAV'93)*, volume 697 of *Lecture Notes in Computer Science*, pages 253–267. Springer, 1993.

[CGP03] Jamieson M. Cobleigh, Dimitra Giannakopoulou, and Corina S. Păsăreanu. Learning Assumptions for Compositional Verification. In Hubert Garavel and John Hatcliff, editors, *Tools and Algorithms for the Construction and Analysis of Systems, Proc. 9th International Conference (TACAS'03)*, volume 2619 of *Lecture Notes in Computer Science*, pages 331–346. Springer, 2003.

[CJK13a] Chris Chilton, Bengt Jonsson, and Marta Kwiatkowska. An Algebraic Theory of Interface Automata. Technical Report CS-RR-13-02, Department of Computer Science, University of Oxford, 2013.

[CJK13b] 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.

[CJK14] Chris Chilton, Bengt Jonsson, and Marta Kwiatkowska. Compositional assume-guarantee reasoning for input/output component theories. *Science of Computer Programming*, 2014.

[CKW12] 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.

[CKW13] Chris Chilton, Marta Kwiatkowska, and Xu Wang. Revisiting Timed Specification Theory II : Realisability. *Computing Research Repository*, abs/1304.7590, 2013.

[CLM89] Edmund M. Clarke, David E. Long, and Kenneth L. McMillan. Compositional model checking. In *Logic in Computer Science, Proc. 4th Annual Symposium (LICS'89)*, pages 353–362. IEEE Computer Society, 1989.

[CLSV05] Ling Cheung, Nancy Lynch, Roberto Segala, and Frits Vaandrager. Switched Probabilistic I/O Automata. In Zhiming Liu and Keijiro Araki, editors, *Theoretical Aspects of Computing, Proc. 1st International Colloquium (ICTAC'04)*, volume 3407 of *Lecture Notes in Computer Science*, pages 494–510. Springer, 2005.

[Col93] Pierre Collette. Application of the Composition Principle to Unity-like Specifications. In Marie-Claude Gaudel and Jean-Pierre Jouannaud, editors, *Theory and Practice of Software Development, Proc. 4th International Joint Conference CAAP/FASE (TAPSOFT'93)*, volume 668 of *Lecture Notes in Computer Science*, pages 230–242. Springer, 1993.

[CT12] A. Cimatti and S. Tonetta. A property-based proof system for contract-based design. In *Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*, pages 21 –28, September 2012.

[dAH01] Luca de Alfaro and Thomas A. Henzinger. Interface automata. *ACM SIGSOFT Software Engineering Notes*, 26(5):109–120, September 2001.

[dAH05] 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.

[dAHS02] Luca de Alfaro, Thomas A. Henzinger, and Mariëlle Stoelinga. Timed Interfaces. In Alberto Sangiovanni-Vincentelli and Joseph Sifakis, editors, *Embedded Software, Proc. 2nd International Conference (EMSOFT'02)*, volume 2491 of *Lecture Notes in Computer Science*, pages 108–122. Springer, 2002.

[DCL11] Benoît Delahaye, Benoît Caillaud, and Axel Legay. Probabilistic contracts: a compositional reasoning methodology for the design of systems with stochastic and/or non-deterministic aspects. *Formal Methods in System Design*, 38(1):1–32, February 2011.

[DHJP08] 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.

[Dil88] David L. Dill. *Trace theory for automatic hierarchical verification of speed-independent circuits*. PhD thesis, Carnegie Mellon University, 1988.

[DKL+11] Benoît Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, Falak Sher, and Andrzej Wasowski. Abstract Probabilistic Automata. In Ranjit Jhala and David Schmidt, editors, *Verification, Model Checking, and Abstract Interpretation, Proc. 12th International Conference (VMCAI'11)*, volume 6538 of *Lecture Notes in Computer Science*, pages 324–339. Springer, 2011.

[DLL+10a] Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. ECDAR: An Environment for Compositional Design and Analysis of Real Time Systems. In Ahmed Bouajjani and Wei-Ngan Chin, editors, *Automated Technology for Verification and Analysis, Proc. 8th International Symposium (ATVA'10)*, volume 6252 of *Lecture Notes in Computer Science*, pages 365–370. Springer, 2010.

[DLL+10b] Alexandre David, Kim G. Larsen, Axel Legay, Ulrik Nyman, and Andrzej Wasowski. Timed I/O automata: a complete specification theory for real-time systems. In Karl Henrik Johansson and Wang Yi, editors, *Hybrid Systems: Computation and Control, Proc. 13th ACM International Conference (HSCC'10)*, pages 91–100. ACM, 2010.

[DLL+12] Alexandre David, Kim G. Larsen, Axel Legay, Mikael H. Moller, Ulrik Nyman, Anders P. Ravn, Arne Skou, and Andrzej Wasowski. Compositional verification of real-time systems using ECDAR. *International Journal on Software Tools for Technology Transfer*, 14(6):703–720, November 2012.

[dNS95] Rocco de Nicola and Roberto Segala. A process algebraic view of input/output automata. *Theoretical Computer Science*, 138(2):391–423, 1995.

[DvB99] Jarad Drissi and Gregor v. Bochmann. Submodule construction for systems of I/O automata. Technical Report 1133, DIRO, University of Montreal, 1999.

[EGP08] Michael Emmi, Dimitra Giannakopoulou, and Corina Păsăreanu. Assume-Guarantee Verification for Interface Automata. In Jorge Cuellar, Tom Maibaum, and Kaisa Sere, editors, *Formal Methods, Proc. 15th International Symposium (FM'08)*, volume 5014 of *Lecture Notes in Computer Science*, pages 116–131. Springer, 2008.

[GL94] Orna Grumberg and David E. Long. Model checking and modular verification. *ACM Transactions on Programming Languages and Systems*, 16(3):843–871, May 1994.

[GR83] Adele Goldberg and David Robson. *Smalltalk-80: the language and its implementation*. Addison-Wesley Longman Publishing Co., Inc., 1983.

[HKKG13] Tingting Han, Christian Krause, Marta Kwiatkowska, and Holger Giese. Modal Specifications for Probabilistic Timed Systems. In Luca Bortolussi and Herbert Wiklicky, editors, *Quantitative Aspects of Programming Languages and Systems, Proc. 11th International Workshop (QAPL'13)*, volume 117, pages 66–80. EPTCS, 2013.

[Hoa69] Charles A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, October 1969.

[Hoa85] Charles A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[ISJ+09] Valérie Issarny, Bernhard Steffen, Bengt Jonsson, Gordon Blair, Paul Grace, Marta Kwiatkowska, Radu Calinescu, Paola Inverardi, Massimo Tivoli, Antonia Bertolino, and Antonino Sabetta. CONNECT Challenges: Towards Emergent Connectors for Eternal Networked Systems. In *Engineering of Complex Computer Systems, Proc. 14th IEEE International Conference (ICEECS'09)*, pages 154–161. IEEE Computer Society, 2009.

[IT13] 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.

[JHJ89] 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.

[JK07] Mark B. Josephs and Hemangee K. Kapoor. Controllable delay-insensitive processes. *Fundamenta Informaticae*, 78(1):101–130, January 2007.

[JL91] Bengt Jonsson and Kim G. Larsen. Specification and Refinement of Probabilistic Processes. In *Logic in Computer Science, Proc. 6th Annual IEEE Symposium (LICS'91)*, pages 266–277. IEEE Computer Society, 1991.

[Jon87] Bengt Jonsson. Modular Verification of Asynchronous Networks. In Fred B. Schneider, editor, *Principles of Distributed Computing, Proc. 6th Annual ACM Symposium (PODC'87)*, pages 152–166. ACM, 1987.

[Jon91] Bengt Jonsson. A Hierarchy of Compositional Models of I/O Automata. Technical Report SICS:R91:04, Swedish Institute of Computer Science, 1991.

[Jon94] Bengt Jonsson. Compositional specification and verification of distributed systems. *ACM Transactions on Programming Languages and Systems*, 16(2):259–303, 1994.

[Jos92] Mark B. Josephs. Receptive process theory. *Acta Informatica*, 29(1):17–31, February 1992.

[JT96] Bengt Jonsson and Yih-Kuen Tsay. Assumption/guarantee specifications in linear-time temporal logic. *Theoretical Computer Science*, 167(1-2):47–72, 1996.

[KLSV11] Dilsun K. Kaynar, Nancy Lynch, Roberto Segala, and Frits Vaandrager. *The Theory of Timed I/O Automata, Second Edition*. Synthesis Lectures on Distributed Computing Theory. Morgan and Claypool Publishers, 2011.

[KNPS06] Marta Kwiatkowska, Gethin Norman, David Parker, and Jeremy Sproston. Performance Analysis of Probabilistic Timed Automata using Digital Clocks. *Formal Methods in System Design*, 29(1):33–78, July 2006.

[KNSS02] Marta Kwiatkowska, Gethin Norman, Roberto Segala, and Jeremy Sproston. Automatic verification of real-time systems with discrete probability distributions. *Theoretical Computer Science*, 282(1):101–150, June 2002.

[Lar90] Kim G. Larsen. Modal Specifications. In Joseph Sifakis, editor, *Automatic Verification Methods for Finite State Systems, Proc. International Workshop*, volume 407 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 1990.

[LNW06] Kim G. Larsen, Ulrik Nyman, and Andrzej Wasowski. Interface Input/Output Automata. In Jayadev Misra, Tobias Nipkow, and Emil Sekerinski, editors, *Formal Methods, Proc. 14th International Symposium (FM'06)*, volume 4085 of *Lecture Notes in Computer Science*, pages 82–97. Springer, 2006.

[LNW07] 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.

[LT89] Nancy Lynch and Mark Tuttle. An introduction to input/output automata. *CWI Quarterly*, 2(3):219–246, September 1989.

[LV95] Nancy Lynch and Frits Vaandrager. Forward and Backward Simulations. *Information and Computation*, 121(2):214–233, September 1995.

[LV96] Nancy Lynch and Frits Vaandrager. Forward and Backward Simulations: II. Timing-Based Systems. *Information and Computation*, 128(1):1–25, July 1996.

[LV07] Gerald Lüttgen and Walter Vogler. Conjunction on processes: Full abstraction via ready-tree semantics. *Theoretical Computer Science*, 373(1-2):19–40, March 2007.

[LV10] Gerald Lüttgen and Walter Vogler. Ready simulation for concurrency: It's logical! *Information and Computation*, 208(7):845–867, July 2010.

[LV13] Gerald Lüttgen and Walter Vogler. Modal Interface Automata. *Logical Methods in Computer Science*, 9(3:4):1–28, August 2013.

[LX90] Kim G. Larsen and L. Xinxin. Equation Solving Using Modal Transition Systems. In *Logic in Computer Science, Proc. 5th Annual IEEE Symposium (LICS'90)*, pages 108–117. IEEE Computer Society, 1990.

[Mai01] Patrick Maier. A Set-Theoretic Framework for Assume-Guarantee Reasoning. In Fernando Orejas, Paul G. Spirakis, and Jan Leeuwen, editors, *Automata, Languages and Programming, Proc. 28th International Colloquium (ICALP'01)*, volume 2076 of *Lecture Notes in Computer Science*, pages 821–834. Springer, 2001.

[Mai03] Patrick Maier. Compositional Circular Assume-Guarantee Rules Cannot Be Sound and Complete. In Andrew D. Gordon, editor, *Foundations of Software Science and Computation Structures, Proc. 6th International Conference (FOSSACS'03)*, volume 2620 of *Lecture Notes in Computer Science*, pages 343–357. Springer, 2003.

[MC81] Jayadev Misra and K. Mani Chandy. Proofs of networks of processes. *IEEE Transactions on Software Engineering*, 7(4):417–426, July 1981.

[McI68] M. Douglas McIlroy. Mass Produced Software Components. In *Software engineering: Report of a conference sponsored by the NATO Science Committee*, pages 138–150. NATO, 1968.

[Mey92] Bertrand Meyer. Applying "design by contract". *Computer*, 25(10):40–51, October 1992.

[Mil80] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980.

[MTC+00] Simon Moore, George Taylor, Paul Cunningham, Robert Mullins, and Peter Robinson. Using stoppable clocks to safely interface asynchronous and synchronous subsystems. In *AINT (Asynchronous INTerfaces) Workshop*, Delft, Netherlands, 2000.

[NT10] Kedar S. Namjoshi and Richard J. Trefler. On the completeness of compositional reasoning methods. *ACM Transactions on Computational Logic*, 11(3:16):1–22, May 2010.

[OW04] Joël Ouaknine and James Worrell. On the language inclusion problem for timed automata: Closing a decidability gap. In Harald Ganzinger, editor, *Logic in Computer Science, Proc. 19th Annual IEEE Symposium (LICS'04)*, pages 54–63. IEEE Computer Society, 2004.

[OW07] Joël Ouaknine and James Worrell. On the decidability and complexity of Metric Temporal Logic over finite words. *Logical Methods in Computer Science*, 3(1:8):1–27, February 2007.

[Par72] David Lorge Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, December 1972.

[PGB+08] Corina Păsăreanu, Dimitra Giannakopoulou, Mihaela Bobaru, Jamieson Cobleigh, and Howard Barringer. Learning to divide and conquer: applying the L* algorithm to automate assume-guarantee reasoning. *Formal Methods in System Design*, 32(3):175–205, June 2008.

[Pnu85] Amir Pnueli. In Transition From Global to Modular Temporal Reasoning about Programs. In Krzysztof R. Apt, editor, *Logics and Models of Concurrent Systems*, NATO ASI Series, pages 123–144. Springer, 1985.

[Rac08] Jean-Baptiste Raclet. Residual for component specifications. *Electronic Notes in Theoretical Computer Science*, 215:93–110, June 2008.

[RBB+09a] 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.

[RBB+09b] 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.

[RBB+11] 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.

[Ros98] A. William Roscoe. *The Theory and Practice of Concurrency*. Prentice Hall, 1998.

[Ros10] A. William Roscoe. *Understanding Concurrent Systems*. Springer, 1st edition, 2010.

[RV96] Judi Romijn and Frits Vaandrager. A note on fairness in I/O automata. *Information Processing Letters*, 59(5):245–250, September 1996.

[Seg97] Roberto Segala. Quiescence, fairness, testing, and the notion of implementation. *Information and Computation*, 138(2):194–210, November 1997.

[Sha96] Mary Shaw. Procedure calls are the assembly language of software interconnection: Connectors deserve first-class status. In David Alex Lamb, editor, *Studies of Software Design, ICSE'93 Workshop*, volume 1078 of *Lecture Notes in Computer Science*, pages 17–32. Springer, 1996.

[Sif05] Joseph Sifakis. A Framework for Component-based Construction (Extended Abstract). In Bernhard Aichernig and Bernhard Beckert, editors, *Software Engineering and Formal Methods, Proc. 3rd IEEE International Conference (SEFM'05)*, pages 293–300. IEEE Computer Society, 2005.

[Tre11] 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.

[TWS06] Lothar Thiele, Ernesto Wandeler, and Nikolay Stoimenov. Real-time interfaces for composing real-time systems. In Sang Lyul Min and Yi Wang, editors, *Embedded Software, Proc. 6th ACM & IEEE International Conference (EMSOFT'06)*, pages 34–43. ACM, 2006.

[Ver94] Tom Verhoeff. *A Theory of Delay-Insensitive Systems*. PhD thesis, Dept. of Math. and C.S., Eindhoven Univ. of Technology, May 1994.

[vG94] 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.

[vGSS95] Rob J. van Glabbeek, Scott A. Smolka, and Bernhard Steffen. Reactive, Generative, and Stratified Models of Probabilistic Processes. *Information and Computation*, 121(1):59–80, August 1995.

[Wan90] Yi Wang. Real-time behaviour of asynchronous agents. In Jos Baeten and Jan Willem Klop, editors, *Theories of Concurrency: Unification and Extension, Proc. 1st International Conference (CONCUR'90)*, volume 458 of *Lecture Notes in Computer Science*, pages 502–520. Springer, 1990.

[WSS97] Sue-Hwey Wu, Scott A. Smolka, and Eugene W. Stark. Composition and behaviors of probabilistic I/O automata. *Theoretical Computer Science*, 176(1-2):1–38, April 1997.

[XGG10] Dana Xu, Gregor Gössler, and Alain Girault. Probabilistic Contracts for Component-Based Design. In Ahmed Bouajjani and Wei-Ngan Chin, editors, *Automated Technology for Verification and Analysis, Proc. 8th International Symposium (ATVA'10)*, volume 6252 of *Lecture Notes in Computer Science*, pages 325–340. Springer, 2010.

[ZYM01] Bin Zhou, Tomohiro Yoneda, and Chris J. Myers. Framework of Timed Trace Theoretic Verification Revisited. In *Proc. 10th Asian Test Symposium*, pages 437–442. IEEE Computer Society, 2001.