Modal Specification Theories for Component-Based Design

Sebastian Bauer

Dissertation an der
Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München
zur Erlangung des Grades Doctor rerum naturalium (Dr. rer. nat.)

vorgelegt von
Sebastian Bauer
aus München

München, den 28. August 2012

Erstgutachter: Prof. Dr. Rolf Hennicker
Zweitgutachter: Prof. Kim Guldstrand Larsen, Ph.D.
Tag der mündlichen Prüfung: 26. Oktober 2012

Abstract

Component-based software engineering has emerged as an important software engineering discipline to cope with the growing complexity of today's software systems. Components are encapsulated software units with well-defined interfaces. A key principle of component-based development is to build larger systems by composition of smaller, less complex components. An important application area concerns reactive systems in which the modelling and verification of component behaviours is essential. To support correct usage and implementation of reactive components, interfaces should be equipped with rigorous formal specifications of component behaviours.

This thesis presents a comprehensive study and analysis of specifications of interfaces for reactive components on the basis of modal input/output automata (MIOs), with a focus on interface refinement, interface composition and interface compatibility. MIOs are based on modal transition systems that were introduced by Larsen and Thomsen and that generalize labelled transition systems by distinguishing between may and must modalities for transitions. MIOs explicitly support loose specifications and offer an elegant approach to stepwise refinement. However, they lack a compatibility notion that is preserved by weak modal refinement, and support neither the integration of data specifications nor the specification of quantitative properties. In this thesis we develop an upwards and downwards closed hierarchy of novel specification theories for MIOs that remedy these shortcomings. Specification theories within the hierarchy are related by theory embeddings. The top element of our hierarchy of specification theories is given by a weak modal specification theory for MIOs including data and quantitative specifications, the bottom element is given by a strong modal specification theory for deterministic MIOs.

On the one hand, we define MIOs with data constraints that integrate control flow and data flow of an interface. This new model extends MIOs by variables which are controlled by the owning component and visible to the environment. Transitions are augmented with pre- and postconditions to describe the dependencies between communication and data states. On the other hand, $\mathcal{K}$-weighted MIOs address quantitative properties by labelling transitions with weights from a partially ordered weight structure $\mathcal{K}$. This generalized formalism is capable of expressing constraints on non-functional properties such as resource consumption or costs.

Our proposed modal specification theories use interface refinement that is either based on strong modal refinement, a white box refinement taking into account internal actions, or weak modal refinement, a black box refinement with observational abstraction of internal actions. Interface composition is defined by synchronous communication via input and output actions. Interface compatibility is based on the notion of strong or weak environment correctness which requires outputs of a component to be received by its environment, in the weak case after some internal steps.

This thesis studies also three particular aspects of (modal) specification theories. The first aspect concerns the verification of refinements. For finite MIOs with data constraints involving infinite variable domains, modal refinement is in general undecidable. We propose predicate abstraction to derive over- and under-approximations for concrete and abstract specifications, respectively, such that refinement between approximations (which is decidable) implies refinement between original specifications. Second, we introduce modal refinement distances for $\mathscr{K}$-weighted MIOs. Modal refinement distances are a generalization of strong modal refinement and measure how close a $\mathscr{K}$-weighted MIO is to refine another one by taking into account distances on weights. Third, we propose a contract approach for interface specifications that explicitly distinguish between assumptions on the environment and guarantees of a component, strictly following the principle of separation of concerns. We study the relation between specification theories and contract theories in an abstract setting, and we show how a contract theory can be built in a generic way on top of any specification theory. We identify behaviour and environment semantics of contracts which are the basis for further definitions of contract refinement and contract composition. The latter raises the problem of finding most permissive assumptions such that the mutual assumptions of composed contracts are satisfied. For complete specification theories supporting quotient, conjunction and a maximal environment operator, we show that a constructive definition of contract composition can be given. The generic contract framework is instantiated for strong specification theories based on deterministic MIOs and MIODs. In particular, we show that deterministic MIOs with strong modal refinement and strong environment correctness form a complete specification theory.

Finally, we have implemented several modal specification and contract theories in the MIO Workbench, an Eclipse-based tool with an intuitive and easy-to-use graphical user interface. It supports the design of MIOs and modal contracts as well as the verification of refinement and compatibility notions used in this thesis.

Zusammenfassung

Komponentenorientierte Softwareentwicklung hat sich als wichtiges Prinzip in der Entwicklung von komplexen Softwaresystemen durchgesetzt. Komponenten verkapseln Softwareelemente und bieten ihre Funktionalität über wohldefinierte Schnittstellen an. Ein Grundprinzip der Komponentenorientierung liegt im Entwickeln von größeren Systemen durch Komposition von kleineren, weniger komplexen Komponenten. Ein wichtiger Anwendungsbereich sind reaktive Systeme, bei denen die Modellierung und Verifikation von Komponentenverhalten unerlässlich ist.

In dieser Arbeit wird eine umfassende Studie über Schnittstellenspezifikation von reaktiven Komponenten durchgeführt. Der Ansatz basiert auf modalen Input/Output-Automaten (MIOs) und fokussiert auf Schnittstellenverfeinerung sowie Komposition und Kompatibilität. MIOs basieren auf modalen Transitionssystemen, die von Larsen und Thomsen eingeführt wurden und die üblichen Transitionssysteme durch die Unterscheidung von “may” und “must” Transitionen verallgemeinern. MIOs ermöglichen lose Spezifikationen sowie flexible Verfeinerungsbegriffe. Jedoch unterstützen MIOs weder einen Kompatibilitätsbegriff der durch schwache modale Verfeinerung erhalten bleibt, noch wurden bisher Datenspezifikationen oder die Spezifikation von quantitativen Eigenschaften integriert. Diese Arbeit entwickelt neuartige Spezifikationstheorien für MIOs, die die genannten Schwächen beheben. Die eingeführten Theorien werden durch Einbettungen in Beziehung gesetzt, wodurch eine nach oben und unten abgeschlossene Hierarchie ensteht. Eine schwache Theorie für MIOs mit Datenspezifikationen und quantitativen Eigenschaften bildet das oberste Element der Hierarchie, das unterste Element ist eine starke Theorie für deterministische MIOs.

Dazu werden MIOs mit Datenspezifikationen eingeführt, die sowohl Kontrollfluss als auch Datenfluss integrieren. Dieser neue Formalismus erweitert MIOs um nach außen hin sichtbare Variablen der Komponenten. Transitionen werden um Vor- und Nachbedingungen erweitert, um die Abhängigkeiten zwischen Kommunikation und Datenänderungen zu modellieren. Außerdem werden $\mathscr{K}$-gewichtete MIOs definiert, die quantitative Eigenschaften durch Elemente auf Transitionen aus einer partiell geordneten Menge $\mathscr{K}$ beschreiben. Dieser allgemeine Formalismus kann Bedingungen an nicht-funktionale Eigenschaften, wie beispielsweise Ressourcenverbrauch oder Kosten, ausdrücken.

Die eingeführten modalen Spezifikationstheorien basieren entweder auf starker Verfeinerung (mit expliziter Betrachtung von internen Aktionen) oder schwacher Verfeinerung (mit Abstraktion von internen Aktionen). Komposition ist definiert über synchrone Kommunikation mittels Input- und Output-Aktionen. Kompatibilität basiert auf starker oder schwacher Umgebungskorrektheit. Diese fordert, dass Outputs einer Komponente von der Umgebung angenommen werden (im schwachen Fall werden interne Aktionen davor erlaubt).

Diese Arbeit befasst sich außerdem mit drei weiteren Aspekten von modalen Spezifikationstheorien. Der erste Aspekt betrifft die Verifikation von Verfeinerung. Für endliche MIOs mit Datenspezifikationen und unendlichen Datenbereichen ist die Verfeinerung im Allgemeinen unentscheidbar. Daher wird eine Prädikatenabstraktion entwickelt, um Über- und Unterapproximationen für konkrete und abstrakte Spezifikationen zu konstruieren, so dass eine (entscheidbare) Verfeinerung zwischen Approximationen die Verfeinerung der ursprünglichen Spezifikationen impliziert. Als zweiten Aspekt werden modale Verfeinerungsdistanzen für $\mathscr{K}$-gewichtete MIOs untersucht. Modale Verfeinerungsdistanzen verallgemeinern starke modale Verfeinerung und geben ein Maß für die Präzision der Verfeinerung unter Beachtung der Transitionsgewichte an. Drittens werden Verträge im Kontext von Schnittstellenspezifikationen von Komponenten untersucht. Verträge unterscheiden explizit zwischen Annahmen an und Garantien für die Umgebung einer Komponenten und folgen damit strikt dem Prinzip “separation of concerns”, der Trennung von unterschiedlichen Aspekten einer Spezifikation. Der Zusammenhang zwischen Spezifikations- und Vertragstheorien wird auf einer abstrakten Ebene untersucht. Insbesondere wird gezeigt, wie eine Vertragstheorie, ausgehend von einer gegebenen Spezifikationstheorie, definiert werden kann. Dazu werden Verhaltens- und Umgebungssemantik von Verträgen definiert, die die Grundlage für Vertragsverfeinerung und Vertragskomposition darstellen. Bei der Vertragskomposition ist insbesondere wichtig, die schwächste Vorbedingung zu finden, so dass die gegenseiten Annahmen der zu komponierenden Verträge erfüllt sind. Für vollständige Spezifikationstheorien, die Konjunktion, Quotient und maximale Umgebungskorrektheit unterstützen, wird gezeigt, wie schwächste Annahmen konstruiert werden können. Dank der generischen Definition der Verträge erhält man Vertragstheorien für deterministische MIOs sowie MIOs mit Datenbedingungen. Insbesondere wird gezeigt, dass deterministische MIOs mit starker modaler Verfeinerung und starker Umgebungskorrektheit eine vollständige Spezifikationstheorie bilden.

Die MIO Workbench ist ein Eclipse-basiertes Werkzeug und implementiert die eingeführten Spezifikations- und Vertragstheorien für MIOs und modale Verträge. Die graphische Oberfläche der MIO Workbench ermöglicht die Modellierung von MIOs und modalen Verträge und unterstützt die Verifikation von Verfeinerungs- und Kompatibilitätsbegriffen.

Eidesstattliche Erklärung

Hiermit erkläre ich an Eidesstatt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

Sebastian Bauer
München, den 28. August 2012

Acknowledgements

I would like to express my cordial gratitude to my supervisor, Rolf Hennicker, for his guidance and great support. In countless discussions, he provided me with invaluable feedback and comments. In particular, during the write up of my thesis, his experienced advice on structuring and presentation issues were very helpful. The atmosphere was always open-minded, warm-hearted and friendly, which made me enjoy working with him at all times.

I am also deeply grateful to Kim G. Larsen, who made my half-year stay (Sept. 2010 - Feb. 2011) at the University of Aalborg, Denmark, possible. All the discussions with him and his group members had been very important for the development of my thesis. In particular, I would like to thank Jiri Srba, Ulrik Nyman and Uli Fahrenberg. My time in Aalborg was not only a scientific success, but also really enjoyable outside university; my sincere thank goes to the group members Claus Thrane and Line Juhl for our numerous evenings we enjoyed together. During my stay in Aalborg, I got to know Axel Legay from the University of Rennes and Andrzej Wasowski from the IT University of Copenhagen who constantly provided me scientific advice. I am also thankful for the invited two-weeks-stay at the University of Rennes in August 2011.

I would like to thank the PST group members for their support and discussions, in particular Martin Wirsing for creating a pleasant working atmosphere within the group and for supporting me in all aspects. I would like to thank my former room mate Matthias Ludwig for the interesting time we spent together working in the GLOWA project, Stephan Janisch for providing me with impressive knowledge about relevant literature, Philip Mayer for all his advice on programming and the MIO Workbench, and Andreas Schroeder and Annabelle Klarl for motivating me during the last phase of the write up of my thesis.

Finally, I am especially thankful for the patience of my friends, and the love, the endless support and encouragement of my parents.

Sebastian Bauer
November 2012

Contents

Chapter 1

Introduction

Reactive software systems are omnipresent in our everyday life. They are running on consumer hardware such as smart phones or televisions, but also in safety-critical air traffic control systems or medical equipment. Such systems are reactive [99] by continuously interacting with their environment and by responding to external stimuli. They are often specifically designed to provide functionalities on limited hardware with strict requirements on resource usage such as time, memory or energy [106, 164]. The design and verification of such increasingly complex systems still face major challenges [165].

An important and established approach to tackle the complexity of such systems is component-based development in which the system is decomposed into smaller and less complex components. Each component is implemented independently and provides access to its functionality by well-defined interfaces, abstracting from internal implementation details. To enable formal analysis and verification of global behavioural properties of reactive component-based systems, component interfaces must be equipped with specifications with concise and rigorous formal semantics. Ongoing research on component-based design [129] seeks for heterogeneous specification languages that adequately address functional as well as non-functional aspects of components.

1.1 Software Components and Component Interface Specifications

There is a common agreement that a (software) component is an independent unit which can be composed with other components to form a larger system. According to Szyperski [169], a software component is best described as follows:

> “A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A soft-

> ware component can be deployed independently and is subject to composition by third parties.”

This definition already mentions most of the key aspects of components and component-based development. *Composition* is the principle by which individual components are composed to form a larger component assembly. The specification of components is given by the specifications of the *component interfaces* which are the access points of the functionality provided by that component. In particular, these *interface specifications* should be seen as *contracts* between the user and the implementor of the component, and may include assumptions on the context (i.e. on the user of the component). The second sentence says that a component should be “deployed independently”. This suggests that the component is an independent, encapsulated entity with its own local memory that works correctly as long as the contracts at its interfaces are respected by the environment. Finally, the importance of independent components also manifests in the phrase “(a software component) is subject to composition by third parties”. Components are often delivered as compiled software units and later integrated by a system architect into a larger system, and therefore any context dependencies must be explicitly specified in the interface specifications to admit the construction of correct component assemblies.

The shape of such interface specification formalisms, which functional or non-functional aspects they should cover and their level of detail, turned out to be quite challenging [129, 165, 164]. In recent years, component-based design [129] has become increasingly important also for embedded systems which often operate under strict limitations like timing constraints or restricted availability of resources like power or fuel. Embedded systems call for novel heterogeneous specification formalisms. Balancing the strive for lightweight, efficiently analyzable specifications and including functional and non-functional aspects to assure component assemblies with strong global properties is still subject of research. Functional aspects should cover the temporal ordering of communication events and changing data states of the component. Non-functional aspects should allow for the integration of time, probabilities or resource consumption.

Another aspect that has received much attention is the integration of behavioural variability to allow for the design of software product lines [55, 77, 53]. The aim in such software product lines is to jointly design a family of variations of a component by a generic specification which can then later be refined to efficient, product- and platform-dependent component implementations.

Finally, a suitable formalism should support a notion of *refinement* to develop component-based systems in a top-down manner, from abstract specifications to concrete specifications or implementations. To ensure correct interaction between reactive components, a notion of *compatibility* is desired which expresses when components work together properly. Refinement should satisfy a *compositionality* property ensuring that refinement is preserved by composition, i.e. we can always replace a component by a refined one while retaining global system refinement. Furthermore, whenever compatibility has been shown on an abstract level of specifications, it should be preserved to any refined specifications.

Altogether, interface specification formalisms should be able to adequately address reactive systems by supporting

- *loose specifications* and *behavioural variability*, with a notion of *refinement* for stepwise development of specifications; ideally, refinement supports *observational abstraction*;
- *composition* for structurally combining specifications, with a notion of *compatibility* determining when specifications can work together properly,
- such that *compositionality of refinement* and *preservation of compatibility* is satisfied.

The formalism should be capable to address the specification of

- *functional properties*, allowing for the specification of *interaction protocols* (like the temporal ordering of communication events) together with *visible data states* of the component integrating control flow and data flow aspects,
- *non-functional properties* including *quantitative properties* like resource consumption (power or fuel) or costs, *timed properties* for expressing real-time constraints, and *probabilities* to enable probabilistic analysis of the behaviour.

Finally, the formalism should support

- *contracts* that explicitly distinguish between *assumptions* on the environment and *guarantees* of a component, strictly following the principle of separation of concerns.

1.2 State of the Art in Interface Specifications

Traditional approaches to the specification of component behaviours include process algebras like CSP [107], CCS [141], or ACP [34]. They focus on modelling parallel, distributed systems and support a variety of behavioural equivalence relations and simulation preorders. Process algebras have also been integrated into component models, for instance, Wright [4] defines a component model with ports and connectors, and CSP is used to model the behaviour of connectors. State machines [98] are another classical approach to modelling the reactive behaviour of components by communication protocols. State machines are also part of the Unified Modeling Language (UML) [149], the standard modelling language for object-oriented systems. Both approaches, based on process algebras and on state machines, are established formalisms for behaviour modelling, and they have seen many different extensions in the literature to cope with the specification of functional (like data) and non-functional properties.

Many recent works in the area of interface specifications are based on state transition systems aiming at light-weight formalisms. This line of research was fundamentally inspired by *I/O Automata* [135] that were introduced by Lynch and Tuttle as a formal model to describe concurrent, distributed systems. I/O automata distinguish between input, output and internal actions. Importantly, the transition relation in I/O automata is input-enabled since all input actions are required to be enabled in every state, modelling reactive components that can never refuse to receive an input. There exist several extensions of I/O automata in the literature, including timed I/O automata [116] and hybrid I/O automata [134]. Using I/O automata for assume-guarantee specification of component interfaces was investigated in [123].

In 2001, de Alfaro and Henzinger introduced *interface automata* [61] that are based on I/O automata, but drop the requirement of input-enabledness, i.e. not in every state there must exist a transition for each input action. Omitting an input in a specific state allows for expressing that this input should not be sent by the environment to the component in the actual state. Interface automata are an instance of interface theories [63] and support refinement, composition and compatibility. Refinement of interface automata is defined by an alternating simulation relation [63] that requires every output of the concrete interface to be simulated by the abstract interface, and vice versa for inputs. Therefore, refinement basically means providing less outputs and offering more inputs.

Compatibility of two interface automata is based on the absence of communication errors, that are reachable states of their composition in which one interface automaton can send out an output to another one which is, however, not ready to take that output due to the absence of a corresponding input transition. De Alfaro and Henzinger introduced a novel optimistic approach to compatibility [61] where two interface automata are compatible if there *exists* an environment in which no communication error is reachable. Since the introduction of interface automata in 2001, several approaches have been proposed that extend interface automata towards time [65], resources [52], data [58, 146], and assume-guarantee rules [76].

Although interface automata constitute a solid theory and offer several extensions to cope with other functional and non-functional properties, we still believe that interface automata, already in their basic form with alternating refinement, have some shortcomings and do not fully meet the requirements listed in Section 1.1 in a satisfactory manner. The reason is twofold. Firstly, there is no way of expressing that certain outputs *must* be performed, guaranteeing a certain progress. Hence, one can always construct trivial refinements consisting of a single input-enabled state, in which no outputs are possible. Secondly, interface automata do not fully support observational refinement. Although the direction in alternating refinement [61, 63] from concrete to abstract supports observational abstraction w.r.t. internal transitions, the direction from abstract to concrete is strict, meaning that no internal steps can be inserted when realizing an input of the abstract specification. One reason why this is disallowed lies in the definition of compatibility which requires inputs to happen *immediately* without delay. A weak simulation of inputs would render preservation of compatibility under refinement fail to hold. We claim that having an observational refinement is crucial for the applicability of interface theories for realistic examples, and in particular, the step from abstract to concrete should support the introduction of internal steps.

In 2007, Larsen et al. [124] proposed to use *modal input/output automata* (MIOs) for the modelling of component interfaces. MIOs are based on Modal Transition Systems (MTSs) which were introduced by Larsen and Thomsen [126] in 1988, and use input, output and internal actions as transition labels. MTSs differ from standard labelled transition systems by having two types of transitions: the *may*-transition relation models the allowed behaviour whereas the *must*-transition relation determines the required behaviour. With these two kinds of transitions at hand, one can express loose behavioural specifications by having proper *may*-transitions that need not be preserved by refinement.

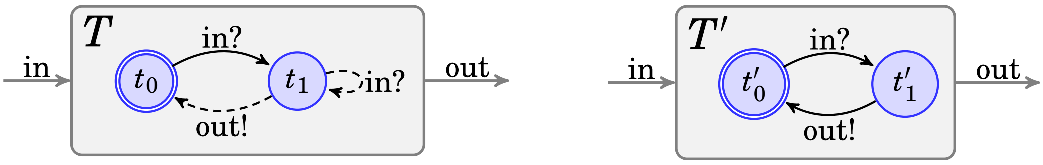


Figure 1.1: Two modal input/output automata T and T'

Examples of MIOs are depicted in Figure 1.1. The dashed transitions represent the allowed behaviour (may-transitions), whereas the transitions with solid line represent required behaviour (must-transitions). Input and output actions are annotated with question and exclamation marks, respectively. The MIO T in Figure 1.1 formulates that the input *in?* must be possible and the subsequent output *out!* is allowed but not required in refinements.

The notion of modal refinement [126] is central for MIOs and is a generalization of bisimulation relating an abstract and a concrete MIO. More precisely, modal refinement is a preorder between two MIOs and is defined in a simulation-like manner requiring the *may*-transitions of the concrete specification to be simulated by the abstract specification, and conversely for the *must*-transitions. A modal refinement T' of T can be seen in Figure 1.1: the may-transition labelled with output action *out!* has been strengthened to a must-transition which must then be preserved in any further refinement of T'.

The main difference to the interface automata is the flexibility of modalities that are independent of the types of actions. Any transition can be declared as a must-transition to be preserved in any refinement, for instance, the MIO T' shown in Figure 1.1 requires *out!* to be possible after each occurrence of *in?*. Therefore, MIOs offer more expressive power than interface automata and are suitable for specifying local liveness (a transition must be present) and local safety properties (a transition is disallowed).

In [124] Larsen et al. transferred the idea of compatibility of interface automata [61] to the level of MIOs. Recently, it was shown that modal transition systems (without input, output and internal actions and without compatibility) form a fully-fledged interface theory [158] supporting operators like conjunction and quotient on specifications which are useful in component-based design.

However, also the theory of modal input/output automata still holds some weaknesses to be discussed. In [109] Hüttel and Larsen proposed weak modal refinement, a natural extension of strong modal refinement [126] to take into account observational abstraction, very similar to the step from strong bisimulation to weak bisimulation [142]. So far, MIOs and compatibility notions have only be considered in the context of strong modal refinement. Moving from strong to weak modal refinement does, however, not conserve the properties of interface theories, as the compatibility notion used in [124] requires that an input action is immediately enabled as soon as the communication partner has enabled the corresponding output action. Clearly, weakly refining a MIO and adding internal steps before a transition with input action may introduce communication errors breaking compatibility. A minimal example illustrating this observation is shown in Figure 1.2. S and T are compatible in the sense of [124], however, their refinements S' and T' are *not* compatible. The refinement T' of T has introduced an internal step *int* before the input action *msg?*, and thus T' is not immediately ready to perform the action *msg?* to receive the output *msg!* from S'. This observation does also hold for interface automata if one would choose alternating refinement with a weak simulation from the abstract to the concrete interface.

In the literature one can find timed extensions [49, 35, 36] of modal transition systems as well as probabilistic extensions [45, 68] and a weighted extension [114]. However, referring to the list of requirements in Section 1.1, MIOs crucially lack any extension to cope with the specification of data states. An integrated formalism is desirable that combines modalities of MIOs with the loose specification of visible data states a component can adopt as well as data states a component requires to access in its environment. Extending MIOs by data is of great importance for practical use in modelling of large-scale systems. Also, MIOs lack the possibility of the specification of quantitative constraints like costs or resource consumption which is of importance once MIOs shall be used for components that are subject to non-functional requirements. Furthermore,

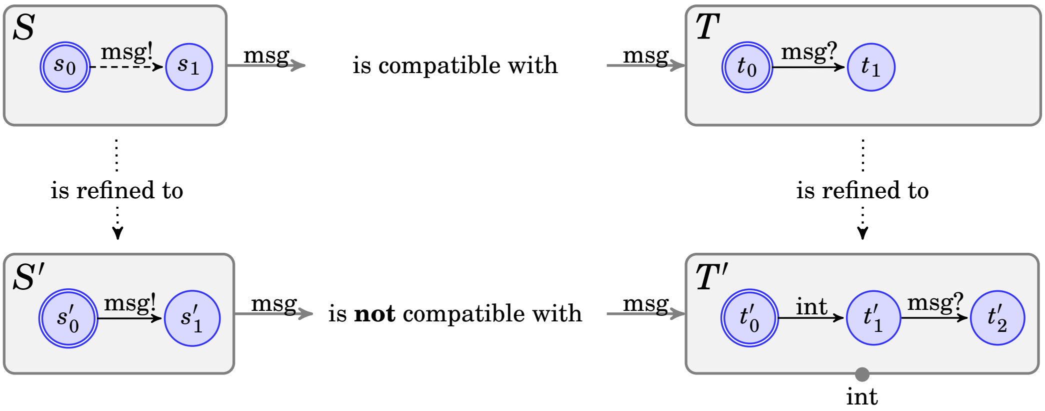


Figure 1.2: Weak modal refinement does not preserve compatibility

in the tradition of interface automata, assumptions and guarantees have been always simultaneously specified in a single MIO. To avoid cluttering of different viewpoints in a single specification, contracts offer a way of explicitly distinguishing component guarantees and assumptions on the environment, leading to a clean separation of concerns. Preliminary work was conducted by Goessler and Raclet [90] and by Quinton and Graf [154] for contracts based on modal transition systems. However, both works [90, 154] do not consider compatibility.

1.3 Research Goals

Our goal in this thesis is to contribute to the field of formal approches to component interface specifications based on modal input/output automata. The motivation of choosing modal input/output automata is that they are expressive enough to rule out the shortcomings of interface automata mentioned above, and modalities for transitions are crucial for obtaining observational specification theories while supporting loose specifications and behavioural variability. This thesis focuses on the following research goals.

- Development of modal input/output automata with
  - **data constraints** for the specification of changing data states of the component, integrating control flow and data flow aspects,
  - **quantitative properties** to address aspects like resource consumption,

  under consideration of

  - **strong modal refinement**, a white box refinement taking into account internal actions, and **weak modal refinement**, a black box refinement with observational abstracting of internal actions,
  - **strong and weak environment correctness** which is preserved by strong and weak modal refinement, respectively.

- Arranging the introduced modal specification theories in an upward and downward closed **hierarchy** obtained by defining embeddings between the theories

- Development of a **generic contract framework** for the specification of component interfaces by explicit assume-guarantee pairs that can be instantiated by any specification theory

In the next section, we give an overview of the results presented in this thesis.

1.4 Contributions

Specification Theories

All results in this thesis are introduced in the context of a *specification theory* that formalizes essential ingredients and properties of any formal theory supporting the compositional design of component interfaces. Our notion of a specification theory is inspired by de Alfaro and Henzinger's interface theories [62] and interface languages [63]. In our study, a specification theory *Th* is formally defined as a tuple

$$Th = (\mathfrak{S}, \mathfrak{S}^i, \otimes, \leq, \to)$$

where $\mathfrak{S}$ is a set of interface specifications, $\mathfrak{S}^i \subseteq \mathfrak{S}$ is a subset of implementations, $\otimes \subseteq \mathfrak{S} \times \mathfrak{S} \to \mathfrak{S}$ is a partial composition operator that composes two (composable) interface specifications, $\leq \subseteq \mathfrak{S} \times \mathfrak{S}$ is a preorder capturing refinement of interface specifications, and finally $\to \subseteq \mathfrak{S} \times \mathfrak{S}$ is an environment correctness predicate that determines pairs (S,E) of specifications expressing that S "works properly" in the environment E when they are composed. All constituents of a specification theory are left abstract. Instances of this framework must define what interface specifications are and what composition, refinement and environment correctness mean. A specification theory requires the following three properties:

Compositional refinement:

If $S \otimes E$ is defined, $S' \leq S$ and $E' \leq E$, then $S' \otimes E'$ is defined and $S' \otimes E' \leq S \otimes E$.

Preservation of environment correctness:

If $S \rightarrow E$, $S' \leq S$ and $E' \leq E$, then $S' \rightarrow E'$.

Finality of implementations:

For all $I \in \mathfrak{S}^i$ and all $S \in \mathfrak{S}$, $S \leq I$ implies $I \leq S$.

In any specification theory we are interested in the *implementation semantics* $[\![S]\!]$ of $S \in \mathfrak{S}$ which is defined as the set of all implementations refining S. A natural refinement preorder, called *thorough refinement*, can then be derived by relying on implementation semantics only: $S \in \mathfrak{S}$ thoroughly refines $T \in \mathfrak{S}$ if and only if $[\![S]\!] \subseteq [\![T]\!]$. An interesting question is in which cases refinement $\leq$ and thorough refinement is equivalent. This question is discussed whenever we introduce a specification theory in this thesis.

For stating precisely the relationships between specification theories we propose morphisms, embeddings and reflective embeddings. A morphism, similar to algebraic homomorphisms in algebraic specification [176], is a function between the sets of interface specifications of two specifications theories that preserves composition, refinement and environment correctness; embeddings are injective morphisms, and an embedding is reflective if refinement and environment correctness is also reflected, i.e. preserved in the opposite direction.

Let $Th_1 = (\mathfrak{S}_1, \mathfrak{S}_1^i, \leq_1, \otimes_1, \rightarrow_1)$ and $Th_2 = (\mathfrak{S}_2, \mathfrak{S}_2^i, \leq_2, \otimes_2, \rightarrow_2)$ be two specification theories. A **morphism** f from Th_1 to Th_2 is a total function $f : \mathfrak{S}_1 \rightarrow \mathfrak{S}_2$ such that, for all $S, T \in \mathfrak{S}_1$,

1. If $S \in \mathfrak{S}_1^i$, then $f(S) \in \mathfrak{S}_2^i$.
2. If $S \otimes_1 T$ is defined, then $f(S) \otimes_2 f(T)$ is defined and $f(S \otimes_1 T) = f(S) \otimes_2 f(T)$,
3. if $S \leq_1 T$, then $f(S) \leq_2 f(T)$,

4. if $S \rightarrow_1 T$, then $f(S) \rightarrow_2 f(T)$.

The morphism f is an **embedding** if f is injective.
The morphism f is a **reflective embedding** of Th_1 in Th_2 if it is an embedding (i.e. injective) and for all $S, T \in \mathfrak{S}_1$,

1. if $f(S) \in \mathfrak{S}_2^i$, then $S \in \mathfrak{S}_1^i$,
2. if $f(S) \otimes_2 f(T)$ is defined, then $S \otimes_1 T$ is defined and $f(S) \otimes_2 f(T) = f(S \otimes_1 T)$,
3. if $f(S) \leq_2 f(T)$, then $S \leq_1 T$,
4. if $f(S) \rightarrow_2 f(T)$, then $S \rightarrow_1 T$.

Establishing a morphism (or embedding) f from a specification theory Th_1 to a specification theory Th_2 allows to transfer any designs with environment correctness and refinement proofs from Th_1 to Th_2. If f is, moreover, a reflective embedding of Th_1 in Th_2 then the notions of implementations, refinement and environment correctness in Th_2, restricted to specifications from the image of f, coincide with the respective notions in Th_1. We may also say that Th_2 is a conservative extension of Th_1.

Specification Theories for MIOs

As our second contribution, we propose a specification theory for the set $\mathbb{MIO}$ of all MIOs and the set $\mathbb{MIO}^i$ of all implementations, based on modal synchronous composition $\otimes$ [126] and strong modal refinement $\leq_s$ [126]. The latter gives rise to implementation semantics $[\![S]\!]_s$ for any MIO $S \in \mathbb{MIO}$ as described before. Thorough refinement of MIOs, defined as inclusion of implementation semantics, is shown to be equivalent to strong modal refinement whenever the abstract MIO is deterministic, i.e. $[\![S]\!]_s \subseteq [\![T]\!]_s$ is equivalent to $S \leq_s T$ whenever T is deterministic. As environment correctness notion we consider strong environment correctness $\rightarrow_s$. Communication errors occur when an output is enabled in a specification and the environment in the current state does not have the corresponding input enabled. A MIO is a strongly correct environment for another MIO if there are no communication errors reachable in their composition. Communication errors are inspired by [61, 124], however, in contrast to [61, 124] we follow a pessimistic view on component compatibility. The result is a specification theory $Th_{strong}^{\mathbb{MIO}}$ for MIOs. We also introduce a specification theory $Th_{strong}^{d\mathbb{MIO}}$

that restricts the sets of MIOs and implementations to deterministic ones and that can be embedded in $Th^{\mathbb{MIO}}_{strong}$.

$$
\begin{aligned}
Th^{\mathbb{MIO}}_{strong} &= \left(\mathbb{MIO}, \mathbb{MIO}^i, \otimes, \leq_s, \rightarrow_s\right)\\
Th^{d\mathbb{MIO}}_{strong} &= \left(d\mathbb{MIO}, d\mathbb{MIO}^i, \otimes, \leq_s, \rightarrow_s\right)
\end{aligned}
$$

To address the problem of strong environment correctness not being preserved by weak modal refinement, we define the novel notion of *weak environment correctness* $\rightarrow_w$ that requires the input to happen possibly after some must-transitions labelled with internal actions, which constitutes a considerable relaxation of strong environment correctness. We derive a specification theory for MIOs, based on weak modal refinement $\leq_w$ [109] and weak environment correctness $\rightarrow_w$, in particular, we show that weak environment correctness is preserved by weak modal refinement.

$$
Th^{\mathbb{MIO}}_{weak} = \left(\mathbb{MIO}, \mathbb{MIO}^i, \otimes, \leq_w, \rightarrow_w\right)
$$

Specification Theories for MIOs with Data Constraints

To allow for modelling rich interfaces with data, we extend MIOs with variables yielding modal input/output automata with data constraints (MIODs). Each MIOD includes two sets of variables V^{prov} and V^{req}. *Provided* variables in V^{prov} are controlled by the owning component, with read and write access, and visible to the environment which has read access only. *Required* state variables also belong to the interface specification and model variables the component expects to be visible in the environment. Crucially, transitions are augmented with pre- and postconditions. A transition label is of the form $[\varphi]\alpha[\pi]$ where α is an action, the precondition φ is a predicate with variables from V^{prov} and V^{req} which acts as a guard restricting the enabledness of the respective transition to the data states (of the interface itself and the environment) which satisfy φ. The postcondition π is a predicate relating a previous data state of V^{prov} and V^{req} to a next data state of V^{prov}, thus determining all possible next (post) data states of the owning component depending on the previous data state. For the postconditions, we adhere to the idea of loose specifications and admit postconditions that are semantically relations (rather than functions). The set of all MIODs is denoted by $\mathbb{MIOD}$. The set $\mathbb{MIOD}^i$ consists of all implementations, that are MIODs for which every allowed transition is also required and all postconditions are assignments mapping a previous data state to exactly one post data state.

We show how modal synchronous composition $\otimes^d$ can be defined on MIODs. Moreover, strong modal refinement can be extended to MIODs accordingly, taking into account pre- and postconditions in transition labels. The idea of strong modal refinement $\leq_s^d$ for MIODs is illustrated below in Figure 1.3.

$$\begin{tikzcd}
T \arrow[d, dotted, "\text{is refined to}"] & t_1 \arrow[r, dashed, "{[\varphi]\alpha[\pi]}"] \arrow[dr, phantom, "\Uparrow\ \Uparrow"] & t_2 & t_1 \arrow[r, "{[\varphi]\alpha[\pi]}"] \arrow[dr, phantom, "\Downarrow\ \Uparrow"] & t_2 \\
S & s_1 \arrow[r, dashed, "{[\varphi']\alpha[\pi']}"] & s_2 & s_1 \arrow[r, "{[\varphi']\alpha[\pi']}"] & s_2
\end{tikzcd}$$

Figure 1.3: Idea of strong modal refinement for MIODs

Enabled may-transitions in the concrete interface specification S must be simulated by the abstract interface specification T, and both enabledness and effect on the data state must be allowed. Conversely, every must-transition in T with precondition φ must be present in S with a possibly stronger precondition φ', and the postcondition π' specifying the effect on the data state must imply π.

Analogous to the case of MIOs, we investigate the relationship between strong modal refinement $\leq_s^d$ for MIODs and thorough refinement (induced by $\leq_s^d$), and we show that they coincide if the abstract MIOD is deterministic.

When variables have large or infinite domains, the verification of strong modal refinement may become subject of the state explosion problem, or even undecidable. To deal with such problems, we propose predicate abstraction [93] for MIODs as a verification technique of strong modal refinement. In order to check a refinement $S \leq_s^d T$ between two MIODs S and T with the same set of variables V^{prov} and V^{req}, we show how to derive over- and under-approximations S^o and T^u of S and T, for which it holds $S \leq_s^d S^o$ and $T^u \leq_s^d T$ by construction of S^o and T^u. These approximations use a finite number of predicates in pre- and postconditions, partitioning the data state space of V^{prov} and V^{req}. Then, $S^o \leq_s^d T^u$ can be decided by encoding the finite number of predicates by Boolean variables. Hence, once one has established $S^o \leq_s^d T^u$, $S \leq_s^d T$ follows from transitivity of strong modal refinement.

We lift strong environment correctness to the level of MIODs, denoted $\rightarrow_s^d$, by taking into account the pre- and postconditions. The result is a specification theory $Th_{strong}^{\mathbb{MIOD}}$ for MIODs and a specification theory $Th_{strong}^{d\mathbb{MIOD}}$ for deterministic MIODs, both based on strong modal refinement and strong environment correctness. In particular, we will show that $Th_{strong}^{d\mathbb{MIOD}}$ can be embedded in $Th_{strong}^{\mathbb{MIOD}}$.

$$Th^{\mathbb{MIOD}}_{strong} = \left(\mathbb{MIOD}, \mathbb{MIOD}^i, \leq^d_s, \otimes^d, \rightarrow^d_s\right)$$
$$Th^{d\mathbb{MIOD}}_{strong} = \left(d\mathbb{MIOD}, d\mathbb{MIOD}^i, \leq^d_s, \otimes^d, \rightarrow^d_s\right)$$

We show how any implementation in $\mathbb{MIOD}$ can be equipped with a denotational semantics, formalized as input/output transition systems with data where a state consists of an abstract control state and a data state for the provided variables, and transitions are guarded by data states for the required variables. We prove that MIODs can be independently implemented in the above sense, in particular, we show that the denotational semantics is preserved by synchronous composition.

We also propose weak modal refinement $\leq^d_w$ and weak environment correctness $\rightarrow^d_w$ for MIODs, that basically follow the same ideas as for MIOs. The difference is here that internal steps may be inserted only if they do not change the current values of provided variables. This restriction is crucial for compositionality of weak modal refinement: if internal transitions could change provided state variables, new behaviour could emerge in the composition of the more concrete specifications that was not allowed by the composition of the more abstract specifications. With weak modal refinement and weak environment correctness, we arrive at a specification theory for MIODs:

$$Th^{\mathbb{MIOD}}_{weak} = \left(\mathbb{MIOD}, \mathbb{MIOD}^i, \otimes^d, \leq^d_w, \rightarrow^d_w\right)$$

Specification Theories for $\mathcal{K}$-Weighted MIOs

We propose $\mathcal{K}$-weighted MIOs ($\mathcal{K}$-WMIOs) that are capable to capture basic quantitative aspects of systems. Transitions are equipped with labels from a weight structure $\mathcal{K} = (K, \preceq, \oplus)$ consisting of a set of weights K, a partial order $\preceq$ on K, and a weight synchronization operator $\oplus$ that describes how weights are combined during parallel composition. Weight structures $\mathcal{K}$ can be instantiated to model quantitative constraints on non-functional properties such as resource consumption or costs. We adapt strong modal refinement to take into account weight refinement, i.e., in a refinement step a label $k \in K$ can be refined to another label $k' \in K$ whenever $k' \preceq k$. On this basis, we revisit strong modal refinement, implementation semantics, thorough refinement, and identify sufficient conditions for completeness of strong modal refinement. We define modal synchronous composition $\otimes^{\mathcal{K}}$ for $\mathcal{K}$-WMIOs and show that it is compositional with respect to strong modal refinement $\leq_s^{\mathcal{K}}$. Since we consider weights that model quantitative constraints like resource consumption that do not necessarily play a role in communication correctness between components, we define strong environment correctness $\rightarrow_s^{\mathcal{K}}$ like $\rightarrow_s$ for MIOs, by not considering the weights. The outcome are two specification theories $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ for $\mathcal{K}$-WMIOs and $Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIO}}$ for deterministic $\mathcal{K}$-WMIOs.

$$Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}} = \left(\mathcal{K}\text{-}\mathbb{WMIO}, \mathcal{K}\text{-}\mathbb{WMIO}^i, \otimes^{\mathcal{K}}, \leq_s^{\mathcal{K}}, \rightarrow_s^{\mathcal{K}}\right)$$
$$Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIO}} = \left(d\mathcal{K}\text{-}\mathbb{WMIO}, d\mathcal{K}\text{-}\mathbb{WMIO}^i, \otimes^{\mathcal{K}}, \leq_s^{\mathcal{K}}, \rightarrow_s^{\mathcal{K}}\right)$$

In [121] Larsen characterized strong modal refinement for MTSs by Hennessy-Milner-Logic (HML). We propose $\mathcal{K}$-HML and prove a logical characterization of strong modal refinement of $\mathcal{K}$-WMIOs.

We also defined weak modal refinement $\leq_w^{\mathcal{K}}$ and weak environment correctness $\rightarrow_w^{\mathcal{K}}$ for $\mathcal{K}$-WMIOs. The latter is again based on $\rightarrow_w$ and is independent of any weights. The former, weak modal refinement $\leq_w^{\mathcal{K}}$, requires simulation in both directions up to internal actions, very similar to weak modal refinement for MIOs. Importantly, however, we allow that weights are distributed to several transitions in a refinement. $\mathcal{K}$-WMIOs together with weak modal refinement and weak environment correctness are shown to form a specification theory.

$$Th_{weak}^{\mathcal{K}\text{-}\mathbb{WMIO}} = \left(\mathcal{K}\text{-}\mathbb{WMIO}, \mathcal{K}\text{-}\mathbb{WMIO}^i, \otimes^{\mathcal{K}}, \leq_w^{\mathcal{K}}, \rightarrow_w^{\mathcal{K}}\right)$$

A Hierarchy of Specification Theories

In the course of the thesis we build up step by step a hierarchy of specification theories for MIOs, MIODs, $\mathcal{K}$-WMIOs. Lastly, we define $\mathcal{K}$-WMIODs as the integration of MIODs and $\mathcal{K}$-WMIOs in a single formalism, and we define modal specification theories $Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIOD}}$, $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIOD}}$ and $Th_{weak}^{\mathcal{K}\text{-}\mathbb{WMIOD}}$. The obtained modal specification theories are arranged in a hierarchy, by defining suitable (reflective) embeddings between them. The complete picture showing all specification theories for MIOs, MIODs, $\mathcal{K}$-WMIOs and $\mathcal{K}$-WMIODs together with all embeddings ($\hookrightarrow$) and reflective embeddings ($\hookrightarrow\!\!\triangleright$) can be seen in the figure below.

$$\begin{tikzcd}
 & & Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{weak} & \\
Th^{\mathbb{MIOD}}_{weak} \arrow[urr, hook, -{Triangle[open]}] & Th^{\mathbb{MIO}}_{weak} \arrow[l, hook, -{Triangle[open]}] \arrow[rr, hook, -{Triangle[open]}] \arrow[ur, hook, -{Triangle[open]}] & & Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak} \arrow[ul, hook, -{Triangle[open]}] \\
 & & Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong} \arrow[uu, hook, ->] & \\
Th^{\mathbb{MIOD}}_{strong} \arrow[uu, hook, ->] \arrow[urr, hook, -{Triangle[open]}] & Th^{\mathbb{MIO}}_{strong} \arrow[uu, hook, ->] \arrow[l, hook, -{Triangle[open]}] \arrow[rr, hook, -{Triangle[open]}] \arrow[ur, hook, -{Triangle[open]}] & & Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[uu, hook, ->] \arrow[ul, hook, -{Triangle[open]}] \\
 & & Th^{d\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong} \arrow[uu, hook, -{Triangle[open]}] & \\
Th^{d\mathbb{MIOD}}_{strong} \arrow[uu, hook, -{Triangle[open]}] \arrow[urr, hook, -{Triangle[open]}] & Th^{d\mathbb{MIO}}_{strong} \arrow[uu, hook, -{Triangle[open]}] \arrow[l, hook, -{Triangle[open]}] \arrow[rr, hook, -{Triangle[open]}] \arrow[ur, hook, -{Triangle[open]}] & & Th^{d\mathcal{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[uu, hook, -{Triangle[open]}] \arrow[ul, hook, -{Triangle[open]}]
\end{tikzcd}$$

Modal Refinement Distances for $\mathcal{K}$-Weighted MIOs

With regard to refinement for quantitative specifications like $\mathcal{K}$-WMIOs, we think that a preorder as a refinement notion is not adequate. We argue that there is a need for a notion of refinement distance that is a (non-symmetric) function that maps pairs (S,T) of specifications to a value in $\mathbb{R}_{\geq 0}$ expressing how "well" S refines T; if S is a strong modal refinement of T then the refinement distance is 0, otherwise it is a value greater than 0 and it yields ∞ if S does not even refine T without considering the weights.

We focus on $\mathcal{K}_{intv}$-WMIOs with the weight structure $\mathcal{K}_{intv} = (K_{intv}, \preceq_{intv}, \oplus_{intv})$ where K_{intv} consists of all integer intervals, the partial order $\preceq_{intv}$ is defined by set inclusion, and $\oplus_{intv}$ is interval addition. We then show for $\mathcal{K}_{intv}$-WMIOs how to lift strong modal refinement to a *modal refinement distance* given by the function

$$d_m : \mathcal{K}_{intv}\text{-}\mathbb{WMIO} \times \mathcal{K}_{intv}\text{-}\mathbb{WMIO} \to \mathbb{R}_{\geq 0} \cup \{\infty\}.$$

The function d_m is defined via recursive equations accumulating distances of integer intervals of matched transitions such that, for any $S,T \in \mathcal{K}_{intv}\text{-}\mathbb{WMIO}$ with the same action signature,

$d_m(S,T) = 0$ whenever S is a strong modal refinement of T;

$0 < d_m(S,T) < \infty$ whenever there is only a slight discrepancy between S and T such that an interval k of a transition in S does not refine the interval ℓ of the matched transition in T, i.e. $k \not\leq_{intv} \ell$;

$d_m(S,T) = \infty$ whenever there is an error in the discrete structure (given by may- and must-transitions) of S and T.

We investigate a thorough notion of modal refinement distances, denoted d_t, and we prove that in general $d_t(S,T) \leq d_m(S,T)$ for two $\mathcal{K}_{intv}$-WMIOs S and T. Equality is shown to hold whenever T is deterministic. Finally, we prove that the modal refinement distance d_m satisfies a quantified version of compositionality, more precisely, we prove that

$$d_m\big(S' \otimes^{\mathcal{K}_{intv}} T', S \otimes^{\mathcal{K}_{intv}} T\big) \leq d_m(S', S) + d_m(T', T)$$

for any $S, S', T, T' \in \mathcal{K}_{intv}$-$\mathbb{WMIO}$. The study of "environment correctness distances" is not further discussed in this thesis, however, it could be defined in principle in a similar fashion as modal refinement distances.

Contracts for Component Interfaces

Component contracts are a popular approach in component-based design: they explicitly distinguish between *assumptions* on the environment and *guarantees* of a component, strictly following the principle of separation of concerns. We study the relation between specification theories and contract theories in an abstract setting, and we show how a contract theory can be built in a generic way on top of any specification theory. In particular, for a contract $C = (A, G)$ with assumption A and guarantee G satisfying $G \to A$, we identify *behaviour semantics* and *environment semantics* $[\![C]\!]_{\mathrm{beh}}$ and $[\![C]\!]_{\mathrm{env}}$ of contracts, respectively. The environment semantics $[\![C]\!]_{\mathrm{env}}$ consists of all environments satisfying the assumption A, whereas the component behaviour semantics $[\![C]\!]_{\mathrm{beh}}$ is given by all implementations I satisfying the guarantee G whenever they are put in an environment satisfying A – this is formally defined by *relativized refinement* $I \leq_A G$, a generalization of refinement. Environment and component behaviour semantics are the basis for the definitions of *contract refinement* and *contract composition*. For the former we define a contract C' to thoroughly refine another contract C, if C' admits less correct component behaviours and more correct environments than C, formally $[\![C']\!]_{\mathrm{beh}} \subseteq [\![C]\!]_{\mathrm{beh}}$ and $[\![C']\!]_{\mathrm{env}} \supseteq [\![C]\!]_{\mathrm{env}}$. We prove that $C' = (A', G')$ thoroughly refines $C = (A, G)$ if and only if $A \leq A'$ and $G' \leq_A G$.

More complex is contract composition of two contracts that shall yield a new contract adequately describing the assumptions and guarantees of the new composed system. We make this precise by the definition of *contract dominance* that subsumes necessary conditions to ensure a sound composition of component behaviours and environments of the individual contracts. A contract composition is then defined as the strongest dominating contract. As a crucial result, we show that a constructive definition of contract composition is possible whenever the underlying specification theory is complete and contracts have normal forms.

- *Complete specification theories* form a subclass of specification theories that additionally offer the following operators: *Conjunction* $\wedge$ computes a greatest lower bound of two specifications w.r.t. refinement; *quotient* $/\!\!/$ yields a maximal solution X to the equation $S \otimes X \leq T$, i.e. $T /\!\!/ S$ is a most general specification X that can be composed with S in order to refine T; finally, the *maximal compatibility operator* $\mathsf{max}_{\cdot\rightarrow}(\cdot)$ computes, for given specifications S and E, a largest refinement E' of E such that $S \rightarrow E'$.
- A contract $C = (A, G)$ has a *normal form* if there is an equivalent contract (A, G^{nf}) with the same semantics such that a specification S is a behaviour of C if and only if $S \leq G^{nf}$, i.e. the behaviour semantics of (A, G^{nf}) is independent of A.

For specification theories that satisfy these assumptions we prove that the contract composition (i.e. a strongest dominating contract) of (A_1, G_1) and (A_2, G_2) can be defined by

$$\left(\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \rightarrow}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf})),\ G_1^{nf} \otimes G_2^{nf}\right).$$

When defining environment correctness of contracts by contract dominance, we arrive again at a specification theory, this time with the set of interface specifications consisting of all contracts.

Thanks to the generic setting of our considerations, we get contract theories “for free” for any specification theory. For the instantiation of the contract framework to the specification theory $Th^{d\mathbb{MIO}}_{strong}$ for deterministic MIOs, we show that, on the one hand, $Th^{d\mathbb{MIO}}_{strong}$ is *complete* (thanks to determinism which is a necessary assumption), and that, on the other hand, every contract over $Th^{d\mathbb{MIO}}_{strong}$ has a normal form. These results allow us to compute contract composition according to the above formula. Moreover, relativized refinement is characterized by a weakening operator.

We also instantiate the generic contract framework for the specification theory $Th^{d\mathbb{MIOD}}_{strong}$ for deterministic MIODs. In contrast to the first instantiation we do not define a complete specification theory, however, we give a direct definition of relativized refinement. We also discuss the role of pre- and postconditions in assumptions and guarantees.

Tool Support: The MIO Workbench

The MIO Workbench is a tool for the verification of MIOs and modal contracts. It is based on the Eclipse framework [75] and features a sophisticated graphical user interface. The tool was initially implemented by Philip Mayer in 2009 and presented for the first time in [27]. In the course of this thesis it was continuously extended and enhanced. The current version of the MIO Workbench implements all relations and operators of the specification theories $Th^{\mathbb{MIO}}_{strong}$ and $Th^{\mathbb{MIO}}_{weak}$, the complete specification theory $Th^{d\mathbb{MIO}}_{strong}$, and the generic contract framework instantiated for the complete specification theory $Th^{d\mathbb{MIO}}_{strong}$. The MIO Workbench supports an input language to define MIOs and modal contracts and to execute verification tasks like modal refinement and environment correctness checks. The parser for the input language is generated with the help of the Xtext framework [171] that also generates a text editor offering syntax highlighting. Importantly, the user interface features a series of editors (graphical MIO editor, graphical contract editor, generated text editor), a shell reusing the generated parser to allow execution of statements of the input language, and a verification view that provides a side-by-side view on two MIOs in order to visualize results of modal refinement and environment correctness checks.

1.5 Thesis Structure

In Chapter 2, we start with defining the abstract framework of specification theories, their morphisms and (reflective) embeddings. The modal specification theories are then introduced.

- In Chapter 3: $Th^{\mathbb{MIO}}_{strong}$, $Th^{d\mathbb{MIO}}_{strong}$ and $Th^{\mathbb{MIO}}_{weak}$
- In Chapter 4: $Th^{\mathbb{MIOD}}_{strong}$, $Th^{d\mathbb{MIOD}}_{strong}$ and $Th^{\mathbb{MIOD}}_{weak}$, with a predicate abstraction technique for the verification of strong modal refinement
- In Chapter 5:
  - $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{strong}$, $Th^{d\mathcal{K}\text{-}\mathbb{WMIO}}_{strong}$, $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$ for $\mathcal{K}$-WMIOs,
  - $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$, $Th^{d\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$, $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{weak}$ for $\mathcal{K}$-WMIODs combining data and quantitative aspects

Modal refinement distances are studied in Chapter 6 for interval weighted MIOs. How to move from specification theories to contracts is investigated in Chapter 7, with exemplifying instantiations to the modal specification theories $Th^{d\mathbb{MIO}}_{strong}$ and $Th^{d\mathbb{MIOD}}_{strong}$. The MIO Workbench that we have used and developed further in the course of this thesis is presented in Chapter 8. Finally, in Chapter 9, we summarize the results and elaborate on possible directions for future work.

Chapter 2

Specification Theories

In this chapter we propose a simple and general algebraic axiomatization, called *specification theory*, that is inspired by previous works on interface theories [62, 63] and captures the algebraic structure of formal theories supporting the top-down design of component-based systems featuring concurrent, reactive components.

We focus on the essential parts such a theory should provide. Firstly, a specification theory should support the stepwise development of component specifications by refining abstract specifications to more concrete ones until reaching specifications that can be considered as (abstractions of) implementations in which no design choices are left open. Secondly, we require a way of structurally composing specifications of concurrently running communicating components. Thirdly, a notion of environment correctness expresses whether a component communicates or interoperates correctly in a given environment.

This chapter introduces the formal notion of a specification theory and lays the foundation of the following parts of the thesis. All of the concrete theories introduced in this thesis later on will be shown to be instances of this abstract framework. The abstract algebraic definition of specification theories immediately leads to morphisms and (reflective) embeddings between specification theories, inspired by algebraic morphisms [176], and allows for arranging the introduced theories in a hierarchy stating precisely their relationships.

Outline. In Section 2.1 we present the formal definition of specification theories, together with associated notions of morphisms and (reflective) embeddings between specification theories. Section 2.2 summarizes related works and we conclude in Section 2.3 with a short summary.

2.1 Definition

Specification theories define a set of *specifications* and a subset of *implementations*, together with their (1) *refinement*, (2) *composition* and (3) *environment correctness* which are key concepts for any specification formalism aiming at top-down design of component-based systems.

(1) The refinement relation $\leq$ is a binary relation that relates "concrete" and "abstract" specifications. A statement $S \leq T$ reads "S refines T" and means that S is less abstract than T.

(2) The composition operator $\otimes$ allows to compose two specifications to a larger one. We admit partial composition operators that compose only composable specifications; composability of specifications usually expresses syntactical restrictions two communicating components must satisfy, like the matching of provided and required interfaces that are going to be connected.

(3) The environment correctness relation $\rightarrow$ expresses that a specification works properly in an environment. A statement $S \rightarrow E$ reads "E is a correct environment for S", or "S feels well in E". The idea of the environment correctness relation is that the component with specification S might require its environment to satisfy some constraints such that S does not fail (the precise definition of "failing" is up to the concrete instantiation). A derived symmetric compatibility notion is given by $S \leftrightarrows T$ if and only if $S \rightarrow T$ and $T \rightarrow S$.

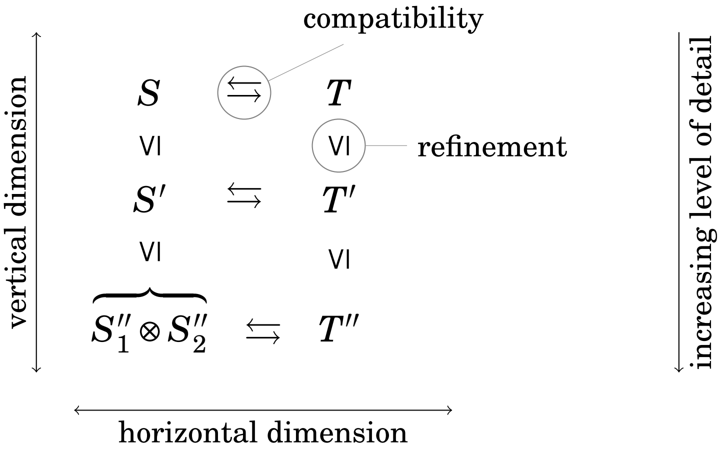


Figure 2.1: The two dimensions of a specification theory

The composition operator and environment correctness relation (and its derived notion of compatibility) concern the *horizontal dimension* of a specification theory, i.e. they operate on different components that are on the same level of abstraction. The *vertical dimension* is addressed by the refinement relation relating specifications for the same component but from different levels of abstractions. The two dimensions are illustrated in Figure 2.1.

There are a series of basic properties a specification theory requires. Firstly, refinement is a preorder (i.e. reflexive and transitive) and the composition operator is commutative and pseudo-associative. Secondly, refinement is preserved by composition.[1] Thirdly, environment correctness is preserved by refinement. Lastly, we identify a subset of specifications, called implementations, that are required to be final elements with respect to refinement.

Definition 2.1.1 (Specification Theory)
A specification theory

$$Th = (\mathfrak{S}, \mathfrak{S}^i, \leq, \otimes, \rightarrow)$$

consists of

- *a set $\mathfrak{S}$ ("specifications"),*
- *a subset $\mathfrak{S}^i \subseteq \mathfrak{S}$ ("implementations"),*
- *a reflexive and transitive relation $\leq \, \subseteq \mathfrak{S} \times \mathfrak{S}$ ("refinement") which gives rise to an equivalence relation on $\mathfrak{S}$: for all $S \in \mathfrak{S}$, $S \approx S'$ if and only if $S \leq S'$ and $S' \leq S$;*
- *a (strict) partial function $\otimes : \mathfrak{S} \times \mathfrak{S} \rightarrow \mathfrak{S}$ ("composition") which is commutative and pseudo-associative in the following sense:*[2]
  - *for all $S, E \in \mathfrak{S}$, if $S \otimes E$ is defined, then $E \otimes S$ is defined and $S \otimes E = E \otimes S$; "=" means set-theoretic equality of elements,*
  - *for all $S, E, E' \in \mathfrak{S}$, if S, E, E' are pairwise composable, then $(S \otimes E) \otimes E'$ and $S \otimes (E \otimes E')$ are defined and $(S \otimes E) \otimes E' = S \otimes (E \otimes E')$;*

  we call S and E composable if $S \otimes E$ is defined;
- *a binary relation $\rightarrow \, \subseteq \mathfrak{S} \times \mathfrak{S}$ ("environment correctness relation") such that, if $S \rightarrow E$, then $S \otimes E$ is defined; we write $S \leftrightarrows E$ and call S and E compatible if $S \rightarrow E$ and $E \rightarrow S$,*

such that the following properties are satisfied:

A1. Compositional refinement:

[1] In other words, refinement is required to be a precongruence with respect to composition.

[2] By $\otimes$ being *strict* we mean that assertions like "$(S \otimes T) \otimes U$ is defined" is equivalent to "$S \otimes T$ is defined and $(S \otimes T) \otimes U$ is defined".

If $S \otimes E$ is defined, $S' \leq S$ and $E' \leq E$, then $S' \otimes E'$ is defined and $S' \otimes E' \leq S \otimes E$.

A2. Preservation of environment correctness:

If $S \rightarrow E$, $S' \leq S$ and $E' \leq E$, then $S' \rightarrow E'$.

A3. Finality of implementations:

For all $I \in \mathfrak{S}^i$ and all $S \in \mathfrak{S}$, $S \leq I$ implies $I \leq S$ (and hence $I \approx S$).

Refinement $\leq$ together with the set of implementations $\mathfrak{S}^i$ leads to the notion of *implementation semantics* of a specification $S \in \mathfrak{S}$ which is defined as the set

$$[\![S]\!] \triangleq \left\{ I \in \mathfrak{S}^i \,\middle|\, I \leq S \right\}.$$

A specification $S \in \mathfrak{S}$ is said to be *consistent* if and only if $[\![S]\!] \neq \emptyset$. Thus, by this definition of consistency, any implementation $I \in \mathfrak{S}^i$ is consistent.

The implementation semantics gives rise to another notion of refinement, called *thorough refinement*: a specification $S \in \mathfrak{S}$ thoroughly refines another specification $T \in \mathfrak{S}$ if and only if $[\![S]\!] \subseteq [\![T]\!]$. An interesting question in every specification theory is the study of the relationship of refinement $\leq$ and thorough refinement. As refinement $\leq$ is transitive, an immediate consequence is that refinement implies thorough refinement, i.e. refinement implies restriction of the set of implementations.

Theorem 2.1.2 (Soundness of Refinement)
For any specifications $S, T \in \mathfrak{S}$, whenever $S \leq T$, then $[\![S]\!] \subseteq [\![T]\!]$.

Proof. Let $I \in [\![S]\!]$, then $I \leq S$. We also know $S \leq T$ by assumption, and since refinement is transitive, it follows that $I \leq T$. I is an implementation, hence $I \in [\![T]\!]$. □

The converse direction, called *completeness of refinement*, cannot be proven on this abstract level. Later in this thesis we see instances of specification theories that do not satisfy the converse implication, and we identify sufficient conditions for completeness of refinement to hold.

Specification theory morphisms and (reflective) embeddings. In the last part of this section, we consider morphisms, embeddings and reflective embeddings between specification theories that express relationships between them. Since specification theories can be considered as algebraic structures [176] a specification theory morphism, similar to an algebraic homomorphism between algebraic structures, is a function between two specification theories preserving the composition operator, the refinement relation and the environment correctness relation. An embedding is an injective morphisms, and reflective embeddings are embeddings for which refinement and environment correctness are also reflected, i.e. preserved in the opposite direction.

Definition 2.1.3 (Morphism, embedding)
Let $Th_1 = (\mathfrak{S}_1, \mathfrak{S}_1^i, \leq_1, \otimes_1, \rightarrow_1)$ and $Th_2 = (\mathfrak{S}_2, \mathfrak{S}_2^i, \leq_2, \otimes_2, \rightarrow_2)$ be two specification theories. A specification theory morphism *(or just* morphism*) from Th_1 to Th_2 is a total function $f : \mathfrak{S}_1 \rightarrow \mathfrak{S}_2$ such that, for all $S, T \in \mathfrak{S}_1$:*

1. *If $S \in \mathfrak{S}_1^i$, then $f(S) \in \mathfrak{S}_2^i$.*
2. *If $S \otimes_1 T$ is defined, then $f(S) \otimes_2 f(T)$ is defined and $f(S \otimes_1 T) = f(S) \otimes_2 f(T)$.*
3. *If $S \leq_1 T$, then $f(S) \leq_2 f(T)$.*
4. *If $S \rightarrow_1 T$, then $f(S) \rightarrow_2 f(T)$.*

An embedding *of Th_1 in Th_2 is an injective morphism from Th_1 to Th_2. If there exists an embedding of Th_1 in Th_2 then we write*

$$Th_1 \hookrightarrow Th_2 .$$

Establishing a specification theory morphism f from Th_1 to Th_2 demonstrates that (1.) f preserves implementations, (2.) translation of specifications from Th_1 to specifications in Th_2 via f is compositional with respect to $\otimes$ and, (3.) and (4.), any environment correctness or refinement assertions in Th_1 can be transferred to Th_2 in which they continue to hold. Embeddings additionally require injectivity. A typical example of a specification theory embedding f of Th_1 in Th_2 is when a component-based design consisting of a collection of specifications in Th_1 shall be transferred to a richer specification theory Th_2, e.g., as we shall see later in this thesis, to include additional aspects like data or quantitative properties. Then the embedding f ensures that any assertions in Th_1 involving $\otimes_1$, $\leq_1$, $\rightarrow_1$ are still valid in Th_2 after translation via f.

Theorem 2.1.4
Any functional composition of morphisms (embeddings) is again a morphism (embedding).

Proof. Let f_{12} be a morphism from Th_1 to Th_2, and let f_{23} be a morphism from Th_2 to Th_3. Then $f_{23} \circ f_{12}$ is a morphism from Th_1 to Th_3.

1. If $S \in \mathfrak{S}_1^i$, then $f_{12}(S) \in \mathfrak{S}_2^i$, hence $f_{23}(f_{12}(S)) \in \mathfrak{S}_3^i$.
2. Let $S, T \in \mathfrak{S}_1$. Then

$$\begin{aligned} f_{23}(f_{12}(S)) \otimes_3 f_{23}(f_{12}(T)) &= f_{23}(f_{12}(S) \otimes_2 f_{12}(T)) \\ &= f_{23}(f_{12}(S \otimes_1 T)). \end{aligned}$$

3. Assume that $S' \leq_1 S$ holds. Then it follows that $f_{12}(S') \leq_2 f_{12}(S)$, and then also $f_{23}(f_{12}(S')) \leq_3 f_{23}(f_{12}(S'))$.
4. Assume that $S \rightarrow_1 E$ is satisfied. Then $f_{12}(S) \rightarrow_2 f_{12}(E)$ which implies $f_{23}(f_{12}(S)) \rightarrow_3 f_{23}(f_{12}(T))$.

The functional composition of embeddings is again an embedding since functional composition preserves injectivity. □

Reflective embeddings are special embeddings g which also reflect implementations, refinement and environment correctness in images of g.

Definition 2.1.5 (Reflective embeddings)
Let $Th_1 = (\mathfrak{S}_1, \mathfrak{S}_1^i, \leq_1, \otimes_1, \rightarrow_1)$ and $Th_2 = (\mathfrak{S}_2, \mathfrak{S}_2^i, \leq_2, \otimes_2, \rightarrow_2)$ be two specification theories. A reflective embedding *of Th_1 in Th_2 is an embedding g of Th_1 in Th_2 such that for all $S, T \in \mathfrak{S}_1$,*

1. *if $g(S) \in \mathfrak{S}_2^i$, then $S \in \mathfrak{S}_1^i$,*
2. *if $g(S) \otimes_2 g(T)$ is defined, then $S \otimes_1 T$ is defined and $g(S) \otimes_2 g(T) = g(S \otimes_1 T)$,*
3. *if $g(S) \leq_2 g(T)$, then $S \leq_1 T$,*
4. *if $g(S) \rightarrow_2 g(T)$, then $S \rightarrow_1 T$.*

If there exists a reflective embedding of Th_1 in Th_2 we also write

$$Th_1 \hookrightarrow\!\!\!\!\!\rhd\; Th_2 \,.$$

Establishing a reflective embedding between two specification theories witnesses a stronger relationship than a morphism or an embedding does. If g is a reflective embedding of Th_1 in Th_2, then any design in Th_1 can be transferred to the images of g in Th_2 while preserving implementations, refinement and environment correctness (since g is also a morphism), but in addition, any new refinement and environment correctness assertions in Th_2 can be transferred back to Th_1 as long as specifications from $g(\mathfrak{S}_1) \subseteq \mathfrak{S}_2$ are involved only. Similar to the case of morphisms and embeddings, reflective embeddings can be functionally composed to form again a reflective embedding.

When given a specification theory Th, and a *prospective* specification theory Th', then it suffices to define a reflective embedding of Th' in Th in order to prove that Th' is indeed a specification theory.

Theorem 2.1.6
Let $Th = (\mathfrak{S}, \mathfrak{S}^i, \leq, \otimes, \rightarrow)$ be a specification theory. Let $Th' = (\mathfrak{S}', (\mathfrak{S}^i)', \leq', \otimes', \rightarrow')$ be a quintuple consisting of a set $\mathfrak{S}'$, a set $(\mathfrak{S}^i)'$, a (strict) partial function $\otimes' : \mathfrak{S}' \times \mathfrak{S}' \rightarrow \mathfrak{S}'$, a relation $\leq' \subseteq \mathfrak{S}' \times \mathfrak{S}'$ and a relation $\rightarrow' \subseteq \mathfrak{S}' \times \mathfrak{S}'$. If there is a function $g : \mathfrak{S}' \rightarrow \mathfrak{S}$ satisfying the conditions of a reflective embedding, then Th' is a specification theory, too.

Proof. We have to prove the conditions a specification theory has to satisfy, cf. Definition 2.1.1.

The domain of g is $\mathfrak{S}'$ and by the first condition in Definition 2.1.5 $g(I)$ is defined for every implementation $I \in (\mathfrak{S}^i)'$, hence $(\mathfrak{S}^i)' \subseteq \mathfrak{S}'$. Refinement $\leq'$ is clearly reflexive and transitive: For any $S \in \mathfrak{S}'$, $S \leq' S$ follows from $g(S) \leq g(S)$. Moreover, for any $S,T,U \in \mathfrak{S}'$, $S \leq' T \leq' U$ implies $g(S) \leq g(T) \leq g(U)$. By transitivity, $g(S) \leq g(U)$, and then $S \leq' U$ by the properties of g. Similarly, one can show pseudo-associativity and commutativity of $\otimes'$. We check that (A1), (A2) and (A3) are satisfied.

A1. Let $S,S',T,T' \in \mathfrak{S}'$ such that $S \otimes' T$ is defined, and assume that $S' \leq' S$ and $T' \leq' T$. Then $g(S) \otimes g(T)$ is defined, $g(S \otimes' T) = g(S) \otimes g(T)$, $g(S') \leq g(S)$ and $g(T') \leq g(T)$. Since (A1) holds in *Th* it follows that $g(S') \otimes g(T')$ is defined and $g(S') \otimes g(T') \leq g(S) \otimes g(T)$. From $g(S') \otimes g(T')$ it follows that $S' \otimes' T'$ is defined and $g(S' \otimes' T') = g(S') \otimes g(T')$. By transitivity of refinement we get $g(S' \otimes' T') \leq g(S \otimes' T)$. Thus $S' \otimes' T' \leq S \otimes' T$.

A2. Let $S,S',E,E' \in \mathfrak{S}'$ and assume that $S \rightarrow' E$ and $S' \leq' S$ and $E' \leq' E$. Then $g(S) \rightarrow g(E)$ and $g(S') \leq g(S)$ and $g(E') \leq g(E)$. Since (A2) holds in *Th* we get $g(S') \rightarrow g(E')$, thus $S' \rightarrow' E'$.

A3. Finally, every element in $(\mathfrak{S}^i)'$ is a final element with respect to refinement. Let $S \in (\mathfrak{S}^i)'$, $T \in \mathfrak{S}'$, and assume that $T \leq' S$. Then $g(T) \leq g(S)$ and $g(S) \in \mathfrak{S}^i$, hence $g(S) \leq g(T)$. The latter implies $S \leq' T$ which was to be shown. □

2.2 Related Work

Our work is inspired by de Alfaro and Henzinger's framework called interface theories [62], which was later reworked in [63]. The formal notion of interface theories was, to our knowledge, the first one to capture crucial operations, relations and their properties that should be satisfied in a theory for component-based top-down design with a focus on interface compatibility and a stepwise development methodology.

In their work [62], an interface theory consists of an interface algebra together with a component algebra thus distinguishing between interface specifications and component implementations. Interface theories were motivated by a common abstract framework with the rudimentary concepts for component-based design and which can be instantiated by concrete formalisms. Later, in [63], the authors introduced *interface languages* which simplified the approach by considering just interfaces (and no component implementations) with the requirements that independent implementability and incremental design is satisfied. De Alfaro and Henzinger defined in [62] a notion that is similar to our morphisms: an interface theory $\mathscr{I}_2$ is as expressive as $\mathscr{I}_1$ if there is a function from the set of interfaces of $\mathscr{I}_1$ to the set of interfaces of $\mathscr{I}_2$ such that composition, compatibility and refinement are preserved (and additional conditions hold for interconnects and the underlying component algebra).

In the 1980s, Joseph Goguen and Rod Burstall developed an abstract framework, called institution [92], defining the essential concepts that a logical system should satisfy. In comparison to specification theories, institutions comprise a notion of signature, model, sentence and a satisfaction relation between models and sentences of the same signature. A satisfaction condition expresses that the validity is invariant under change of signatures. Main differences between institutions and specification theories are the following:

- In specification theories interfaces play the role of both models and sentences, e.g. $I \leq S$ means that the implementation $I \in \mathfrak{S}^i$ satisfies the specification $S \in \mathfrak{S}$ which could also be written $I \models S$ in more logical terms.

- In specification theories we abstract from signatures of specifications.

- Institutions do not support a concept of structurally composing communicating models.

- Environment correctness is a property that could be considered as a sentence, however, environment correctness is a property between two models, one acting as component behaviour model and the other one as environment model.

An example of using institutions to formalize the design of component-based systems is the work on CommUnity which is a parallel program design language developed by Fiadeiro and Maibaum in [82, 81]. They show how a framework for the design of parallel programs (or concurrent components) can be formalized as an institution.

One aspect that is not covered by specification theories, often seen in other languages geared towards the specification of component-based systems, is the architectural (hierarchical) structure of these systems. Several formalisms have been proposed in the literature for the architectural design of systems, also referred to as architectural description languages (ADL), like, e.g., Wright [4] and Darwin [136]. A first attempt of including architectural aspects in the abstract setting of specification theories has been recently developed by Hennicker and Knapp in [102].

Publication history. The notion of specification theories is based on [27], however, rather than relying on a symmetric compatibility relation like in [27], we consider a non-symmetric environment correctness relation. The use of environment correctness is more general since the symmetric compatibility relation $\leftrightarrows$

can be obtained by →, see Definition 2.1.1. Morphisms have already been introduced in [22].

2.3 Summary

In this chapter we have proposed a uniform algebraic framework for specification theories supporting compositional top-down design of component-based systems. A specification theory should feature a composition operator to structurally combine communicating components, an environment correctness relation which expresses when communicating components interoperate properly with their environment, and a refinement relation supporting the stepwise development of implementations from abstract specifications.

We have identified important properties that specification theories should support, most importantly compositionality of refinement and preservation of environment correctness by refinement. We have shown how algebraic morphisms, embeddings and reflective embeddings allow to relate different specification theories.

Chapter 3

Modal Input/Output Automata

In this chapter we study modal input/output automata [124] as the basis for specification theories. Modal input/output automata are an extension of modal transition systems [126] which were introduced by Larsen and Thomsen in 1988 and which are labelled transition systems but with two types of transitions: a *may*-transition relation expresses allowed transitions that may be present in a refinement, whereas *must*-transitions are mandatory transitions that need to be preserved in any refinement. Modal transition systems together with modal refinement give rise to consider modal transition systems as abstractions of (sets of) labelled transition systems.

In recent years, modal transition systems have gained a lot of interest in the research community for formal modelling of component behaviour. The main reasons are twofold: From a theoretical point of view, they are a lightweight approach to the flexible refinement of transition system-based specifications, and the concept of must-transitions allows to express local liveness properties. On the other hand, they are from practical interest to the software product-line community [152, 55, 10, 84] since modal transition systems can be used to specify a whole set of different implementations (represented as ordinary labelled transition systems).

In the first part of this chapter, we recall modal input/output automata [124] as an extension of modal transition systems [126] by input, output and internal actions, and their basic notions like synchronous composition and modal refinement [126, 109]. For defining a suitable environment correctness notion for modal input/output automata, we follow the approach of [124], which is inspired by compatibility of interface automata [61], and consider as a communication error whenever a component can issue an output for which the environment is not ready to take that output as an input. As our first contribution in this chapter, we show that modal input/output automata together with strong modal refinement and strong environment correctness form a specification theory.

As strong environment correctness is not preserved by weak modal refinement, we propose as our second contribution a new observational version of environment correctness which we call weak environment correctness. Weak environment correctness allows non-observable actions to happen before the relevant input and we show that it is preserved by weak modal refinement hence giving rise to a novel observational specification theory for modal input/output automata.

Outline. In Section 3.1 we give a thorough introduction to modal input/output automata. Modal synchronous composition as our composition operator for modal input/output automata is introduced in Section 3.2. The first specification theories $Th^{\mathbb{MIO}}_{strong}$ and $Th^{d\mathbb{MIO}}_{strong}$ are defined in Section 3.3, with strong modal refinement and strong environment correctness. Weak modal refinement and weak environment correctness is presented in Section 3.4 which gives rise to another specification theory $Th^{\mathbb{MIO}}_{weak}$ that can be related to $Th^{\mathbb{MIO}}_{strong}$ by an embedding. Finally, in Section 3.5 we discuss related work, and summarize the content of this chapter in Section 3.6.

3.1 Definition

The behaviour of components is described with *external actions* which can be either *input* or *output actions* used for communication with the environment, and *internal actions* which model internal communication or computation.

Definition 3.1.1 (Action signature)
An action signature *is a triple* $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ *where* Σ^{in}*,* Σ^{out}*,* Σ^{int} *are the pairwise disjoint sets of input, output, and internal actions, respectively. We write* Σ^{ext} *for the set of external actions, that is* $\Sigma^{in} \cup \Sigma^{out}$*.*

Actions are usually denoted with small Greek letters. We write $\bigcup\Sigma$ for the union $\Sigma^{in} \cup \Sigma^{out} \cup \Sigma^{int}$. We usually refer with indices to the respective components of an action signature, i.e. when we are given an action signature Σ_1, then Σ_1^{in}, Σ_1^{out}, Σ_1^{int} are the sets of input, output and internal actions of Σ_1, respectively.

When two components with action signatures Σ_1 and Σ_2 are composed to form a larger system, every shared action becomes an internal action. In this thesis we restrict ourselves to the case of binary communication, so an action a is an external action of at most two components, an input of one component and an output of the other one. Other, more general communication schemes in related approaches are discussed in Section 3.5.

Composing action signatures is not always meaningful according to the above restrictions. Two action signatures Σ_1 and Σ_2 are *composable* if every shared action in $\bigcup\Sigma_1 \cap \bigcup\Sigma_2$ is either an input action in Σ_1 and an output action in Σ_2, or vice versa.

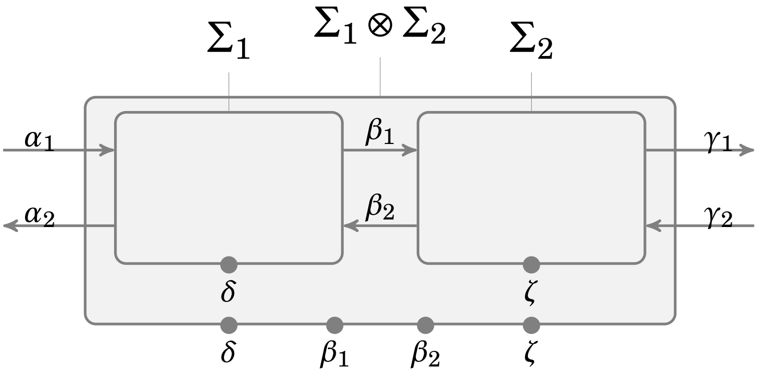


Figure 3.1: Composition of action signatures schematically

Definition 3.1.2 (Composability)
Two action signatures Σ_1 and Σ_2 are composable *if*

$$\left(\bigcup \Sigma_1\right) \cap \left(\bigcup \Sigma_2\right) = (\Sigma_1^{in} \cap \Sigma_2^{out}) \cup (\Sigma_1^{out} \cap \Sigma_2^{in}).$$

The composition of two composable action signatures results in a new action signature in which shared (external) actions become internal actions.

Definition 3.1.3 (Composition)
Let Σ_1 and Σ_2 be two composable action signatures. The composition of Σ_1 and Σ_2 *is defined by $\Sigma_1 \otimes \Sigma_2 \triangleq (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ where*

$$\begin{aligned}
\Sigma^{in} &= (\Sigma_1^{in} \cup \Sigma_2^{in}) \setminus \mathrm{shared}(\Sigma_1, \Sigma_2),\\
\Sigma^{out} &= (\Sigma_1^{out} \cup \Sigma_2^{out}) \setminus \mathrm{shared}(\Sigma_1, \Sigma_2),\\
\Sigma^{int} &= \Sigma_1^{int} \cup \Sigma_2^{int} \cup \mathrm{shared}(\Sigma_1, \Sigma_2),
\end{aligned}$$

with $\mathrm{shared}(\Sigma_1, \Sigma_2) = (\Sigma_1^{in} \cap \Sigma_2^{out}) \cup (\Sigma_2^{in} \cap \Sigma_1^{out})$.

Example 3.1.4
Figure 3.1 illustrates the composition of two composable action signatures Σ_1 and Σ_2, denoted $\Sigma_1 \otimes \Sigma_2$. Each frame represents an action signature; ingoing and outgoing arrows at the frame border show the input and output actions, bullets at the frame border represent internal actions. For instance, Σ_1 is the action signature determined by $\Sigma_1^{in} = \{\alpha_1, \beta_2\}$, indicated by incoming arrows, $\Sigma_1^{out} = \{\alpha_2, \beta_1\}$, indicated by outgoing arrows, and $\Sigma_1^{int} = \{\delta\}$, indicated by a bullet. When Σ_1 and Σ_2 are composed, the shared external actions $\beta_1 \in \Sigma_1^{out} \cap \Sigma_2^{in}$ and $\beta_2 \in \Sigma_2^{out} \cap \Sigma_1^{in}$ become internal actions in $\Sigma_1 \otimes \Sigma_2$, all other non-shared actions keep their types. Altogether, $\Sigma_1 \otimes \Sigma_2$ is determined by the output actions α_2 and γ_1, the input actions α_1 and γ_2, and the internal actions $\delta, \beta_1, \beta_2, \zeta$.

In several sections of this thesis we will focus only on the observable behaviour of components and hence treat internal actions as non-observable actions (traditionally called τ actions in the literature [142]). To achieve a uniform approach to refinement taking into account internal action names as well as observational refinement *not* taking into account internal action names, we would like to use the same action signatures in both viewpoints and avoid the explicit introduction of τ actions. Therefore, when we switch to the observational view on refinement in Section 3.4, we will always assume that each occurring internal action has a globally unique name, and if composition yields new internal actions then they are named with a new name. In practice, when composing two components with a common internal action in the observational setting, then this should be still composable and we achieve composability by renaming the clashing internal action (in one of the action signatures) to an internal action with a new name. To simplify presentation, we will not mention this renaming process in the following anymore and always assume the absence of name clashes of internal actions, but the reader should keep in mind that in the observational treatment of modal input/output automata names of internal actions do not have any relevance.

Modal input/output automata [124] are based on modal transition systems as introduced by Larsen and Thomsen in 1988 [126]. Modal transition systems are similar to labelled transition systems but equipped with two transition relations: a may-transition relation expresses which transitions are allowed to be present in a refinement, whereas a must-transition relation expresses which transitions are required to be present in any refinement.

Definition 3.1.5 (Modal transition system (MTS))
A modal transition system (MTS) *is a tuple*

$$(St, s_0, Act, \dashrightarrow, \longrightarrow)$$

where St is a set of states, $s_0 \in St$ is an initial state, Act is a set of actions, $\dashrightarrow \subseteq St \times Act \times St$ is a may-transition relation, and $\longrightarrow \subseteq St \times Act \times St$ is a must-transition relation such that $\longrightarrow \subseteq \dashrightarrow$.

Note that MTSs are usually assumed to be *syntactically consistent*, that is $\longrightarrow \subseteq \dashrightarrow$, i.e., every required transition is also allowed. In the literature one also finds mixed transition systems [7] which are MTSs with the requirement $\longrightarrow \subseteq \dashrightarrow$ dropped. Also note that modalities in modal transition systems express constraints for refinement – however, a must-transition does not necessarily mean that this transition must be performed in any system execution. This becomes clear if there is a choice between two must-transitions.

Following [124], we extend modal transition systems by replacing the set of actions *Act* by an action signature $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$, as introduced in the previous section. This extension by input/output/internal actions is in line with related works in the literature, like interface automata [61] or I/O automata [133].

Definition 3.1.6 (Modal input/output automaton (MIO))
A modal input/output automaton (MIO)[1] *is a tuple*

$$(St, s_0, \Sigma, \dashrightarrow, \longrightarrow)$$

where St is a set of states, $s_0 \in St$ is an initial state, $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ is an action signature, $\dashrightarrow \subseteq St \times \bigcup \Sigma \times St$ is a may-transition relation and $\longrightarrow \subseteq St \times \bigcup \Sigma \times St$ is a must-transition relation such that $\longrightarrow \subseteq \dashrightarrow$.

A MIO $(St, s_0, \Sigma, \dashrightarrow, \longrightarrow)$ is

- deterministic *if for any transitions $s \overset{\alpha}{\dashrightarrow} s'$ and $s \overset{\alpha}{\dashrightarrow} s''$, it holds that $s' = s''$,*

- *an* implementation *if $\longrightarrow \, = \, \dashrightarrow$.*

All facts and definitions that we provide for particular MIOs are independent of the names of the states of the MIO. In fact we will use MIOs as representatives of their isomorphism classes w.r.t. bijections on states. The set of those isomorphism classes is denoted by $\mathbb{MIO}$. The set of isomorphism classes for deterministic MIOs is denoted by $d\mathbb{MIO}$. The set of isomorphism classes of all (deterministic) implementations is denoted by $\mathbb{MIO}^i$ ($d\mathbb{MIO}^i$, respectively).

We introduce a couple of notations for MIOs which will be frequently used throughout this thesis and which apply also to all other variants of MIOs later on. We usually name MIOs by capital Latin letters. States are usually denoted by same small Latin letters as the MIO itself. If the elements of a MIO S are not explicitly listed then we refer with St_S, s_0, Σ_S, $\dashrightarrow_S$, $\longrightarrow_S$ to the respective elements of S. As usual, we write $s \overset{\alpha}{\dashrightarrow}_S s'$ for a transition $(s, \alpha, s') \in \dashrightarrow_S$. Moreover, $s \overset{\alpha}{\dashrightarrow}_S$ abbreviates $\exists s' \in S : s \overset{\alpha}{\dashrightarrow}_S s'$. For a sequence of actions $\sigma \in (\Sigma)^*$, the notation $s \overset{\sigma}{\dashrightarrow}_S s'$ means that σ is the empty sequence and $s = s'$, or $\sigma = \alpha_1 \alpha_2 \ldots \alpha_n$ for some $n \geq 1$ and

$$\exists s_1, \ldots, s_{n-1} \in S : s \overset{\alpha_1}{\dashrightarrow}_S s_1 \overset{\alpha_2}{\dashrightarrow}_S s_2 \cdots s_{n-1} \overset{\alpha_n}{\dashrightarrow}_S s'.$$

The label τ is used as a placeholder for an arbitrary internal action, e.g., a transition $s \overset{\tau}{\dashrightarrow}_S s'$ means that there exists $\alpha \in \Sigma^{int}$ such that $s \overset{\alpha}{\dashrightarrow}_S s'$, and $s \overset{\tau^*}{\dashrightarrow}_S s'$ stands for $s \overset{\sigma}{\dashrightarrow}_S s'$ where σ is a finite sequence of internal actions, possibly empty sequence and then $s = s'$. All above notations similarly apply to the must-transition relation.

Example 3.1.7
The example in Figure 3.2 shows a MIO T with action signature determined by $\Sigma_T^{in} = \{coin\}$, $\Sigma_T^{out} = \{coffee, tea\}$ and $\Sigma_T^{int} = \{brewWater\}$. The action signature is again indicated by the frame having incoming and outgoing arrows for input and

[1] The abbreviation MIO ['mi:o] stands for modal input/output automaton, and we write MIOs for the plural form modal input/output automata.

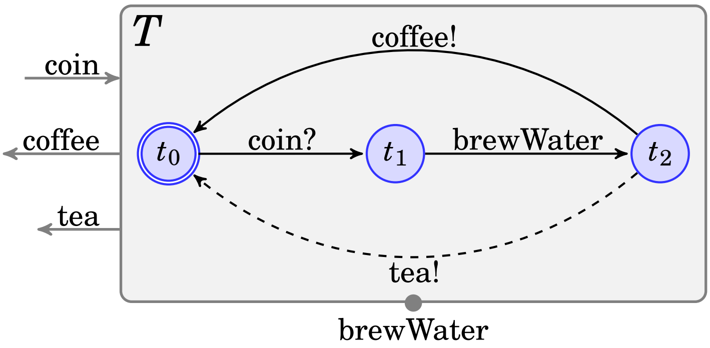


Figure 3.2: Vending machine which optionally dispenses tea

output actions, respectively, and a bullet for the internal actions. The states and transitions are depicted within the frame. The initial state t_0 is indicated by a double circle. May-transitions are drawn with a dashed arrow, must-transitions with solid arrows. May-transitions underlying must-transitions are not drawn for simplicity.

To emphasize the type of actions labelling transitions, we often write a question mark after an input action and an exclamation mark after an output action.

The MIO T specifies a simple vending machine. After throwing a coin into the machine's slot, the machine starts brewing water and then dispenses either coffee or tea. Coffee must always be possible in state t_2, whereas tea is an optional feature that might or might not be possible in a refinement. This intuitive explanation of the semantics of may- and must-transitions is formalized by modal refinement defined in Section 3.3 below.

3.2 Modal Synchronous Composition

Modal transition systems can be synchronously composed [126] by synchronizing on shared actions, and by interleaving of other non-shared actions. In this thesis, we solely consider blocking semantics of message passing, i.e. if a shared action is enabled in one MIO but not in the other one, then there is no transition for that action in the composition. Modal synchronous composition [126] mimics the usual parallel composition of labelled transition systems [142] on the level of modal transition systems. Crucially, there is a must-transition in the composition labelled with a shared action only if the synchronized transitions are must-transitions.

According to the restrictions for composing action signatures, we say that two MIOs S_1 and S_2 are *composable* if their action signatures Σ_1 and Σ_2 are composable.

Definition 3.2.1 (Modal Synchronous Composition)
Let $S_j = (St_j, s_{0,j}, \Sigma_j, \dashrightarrow_j, \longrightarrow_j) \in \mathbb{MIO}$ with $\Sigma_j = (\Sigma_j^{in}, \Sigma_j^{out}, \Sigma_j^{int})$, $j \in \{1,2\}$, such that S_1 and S_2 are composable. The modal synchronous composition *of S_1 and S_2 is defined by the MIO*

$$S_1 \otimes S_2 \triangleq (St_1 \times St_2, (s_{0,1}, s_{0,2}), \Sigma_1 \otimes \Sigma_2, \dashrightarrow, \longrightarrow)$$

where the transition relations $\dashrightarrow$, $\longrightarrow$ are defined by the following rules:

$$\frac{s_1 \overset{\alpha}{\dashrightarrow}_1 s_1' \quad \alpha \notin \Sigma_2^{ext}}{(s_1, s_2) \overset{\alpha}{\dashrightarrow} (s_1', s_2)} \qquad \frac{s_1 \xrightarrow{\alpha}_1 s_1' \quad \alpha \notin \Sigma_2^{ext}}{(s_1, s_2) \xrightarrow{\alpha} (s_1', s_2)}$$

$$\frac{s_2 \overset{\alpha}{\dashrightarrow}_2 s_2' \quad \alpha \notin \Sigma_1^{ext}}{(s_1, s_2) \overset{\alpha}{\dashrightarrow} (s_1, s_2')} \qquad \frac{s_2 \xrightarrow{\alpha}_2 s_2' \quad \alpha \notin \Sigma_1^{ext}}{(s_1, s_2) \xrightarrow{\alpha} (s_1, s_2')}$$

$$\frac{s_1 \overset{\alpha}{\dashrightarrow}_1 s_1' \quad s_2 \overset{\alpha}{\dashrightarrow}_2 s_2' \quad \alpha \in \Sigma_1^{ext} \cap \Sigma_2^{ext}}{(s_1, s_2) \overset{\alpha}{\dashrightarrow} (s_1', s_2')} \qquad \frac{s_1 \xrightarrow{\alpha}_1 s_1' \quad s_2 \xrightarrow{\alpha}_2 s_2' \quad \alpha \in \Sigma_1^{ext} \cap \Sigma_2^{ext}}{(s_1, s_2) \xrightarrow{\alpha} (s_1', s_2')}$$

As we consider MIOs up to bijections between the sets of states and names of internal actions are relevant for composability of MIOs, modal synchronous composition is commutative and pseudo-associative.[2]

Lemma 3.2.2
Modal synchronous composition $\otimes$ is commutative and pseudo-associative.

Proof. Given two composable MIOs S_1 and S_2, then $S_1 \otimes S_2$ and $S_2 \otimes S_1$ only differ in their state names, hence $S_1 \otimes S_2 = S_2 \otimes S_1$ since we consider MIOs up to bijection between the sets of states. To show pseudo-associativity, assume three pairwise composable MIOs S_1, S_2 and S_3. Then also $(S_1 \otimes S_2) \otimes S_3$ and $S_1 \otimes (S_2 \otimes S_3)$ are defined. We only show that $S_1 \otimes (S_2 \otimes S_3)$ is defined, the other claim is symmetric. Assume that (w.l.o.g., the other cases are similar) there is $\beta \in \Sigma_1^{out} \cap \Sigma_{23}^{out}$ where Σ_{23}^{out} is the set of output actions of $S_2 \otimes S_3$. Either $\beta \in \Sigma_2^{out}$, then $S_1 \otimes S_2$ would not be defined; or $\beta \in \Sigma_3^{out}$, but then $S_1 \otimes S_3$ would not be defined; hence $\Sigma_1^{out} \cap \Sigma_{23}^{out} = \emptyset$. Thus we can conclude that $S_1 \otimes (S_2 \otimes S_3)$ is defined.

We do not further elaborate on the transition relations in $(S_1 \otimes S_2) \otimes S_3$ and $S_1 \otimes (S_2 \otimes S_3)$. The proof that their may- and must-transition relations coincide is straightforward. □

[2] If internal action names are relevant then one could even show associativity in the sense that for all $S, E, E' \in \mathfrak{S}$, if $(S_1 \otimes S_2) \otimes S_3$ is defined, then $S_1 \otimes (S_2 \otimes S_3)$ is defined and $(S_1 \otimes S_2) \otimes S_3 = S_1 \otimes (S_2 \otimes S_3)$. However, if internal action names are not relevant (which will be the case for weak modal refinement) this stronger claim fails to hold because composability of $(S_1 \otimes S_2)$ and S_3 does not imply composability of S_2 and S_3 in general. A counterexample is as follows: If S_1 has input action $\alpha?$ and S_2 and S_3 have output action $\alpha!$, then $(S_1 \otimes S_2) \otimes S_3$ is defined because internal action names are not relevant, however, $S_2 \otimes S_3$ is undefined.

3.3 The Specification Theories $Th^{\mathbb{MIO}}_{strong}$ and $Th^{d\mathbb{MIO}}_{strong}$

3.3.1 Strong Modal Refinement

Refinement allows to relate an abstract specification with a more concrete specification providing means to develop a system in a stepwise refinement process. Strong modal refinement [126] requires that, on the one hand, the concrete specification does not have more transitions than allowed by the abstract specification, and on the other hand, the concrete specification has all the transitions that are required by the abstract specification. Modal refinement is formalized as a bisimulation-like relation, however, each direction of the simulation is restricted to may- or must-transitions.

Definition 3.3.1 (Strong modal refinement)
Let $S,T \in \mathbb{MIO}$ with the same action signature $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$. S is a strong modal refinement *of T, denoted $S \leq_s T$, if there exists a refinement relation $R \subseteq St_S \times St_T$ such that $(s_0,t_0) \in R$, and for all $(s,t) \in R$, for all $\alpha \in \bigcup\Sigma$,*

1. *for all $s' \in St_S$, if $s \overset{\alpha}{\dashrightarrow}_S s'$ then there exists $t' \in St_T$ such that $t \overset{\alpha}{\dashrightarrow}_T t'$ and $(s',t') \in R$,*

2. *for all $t' \in St_T$, if $t \xrightarrow{\alpha}_T t'$ then there exists $s' \in St_S$ such that $s \xrightarrow{\alpha}_S s'$ and $(s',t') \in R$.*

It is easy to see that strong modal refinement is a preorder, i.e. reflexive and transitive. If T, the abstract specification, consists of may-transitions only, then strong modal refinement coincides with simulation [142]; if T is an implementation, that is the may- and must-transition relations coincide, then strong modal refinement is equivalent to strong bisimulation [142]. We write $S \approx_s T$ if both $S \leq_s T$ and $T \leq_s S$ hold.

Lemma 3.3.2
Every implementation $I \in \mathbb{MIO}^i$ is a final element with respect to strong modal refinement.

Proof. Let $I \in \mathbb{MIO}^i$ and assume another MIO $S \in \mathbb{MIO}$ such that $S \leq_s I$, with refinement relation R. We define a new refinement relation $R^{-1} = \{(i,s) \mid (s,i) \in R\}$, and show that R^{-1} proves $I \leq_s S$. Clearly, $(i_0,s_0) \in R^{-1}$. Let $(i,s) \in R^{-1}$.

1. Assume $i \overset{\alpha}{\dashrightarrow}_I i'$. Since I is an implementation, we know $i \xrightarrow{\alpha}_I i'$. From $(s,i) \in R$ it follows that there exists $s \xrightarrow{\alpha}_S s'$ such that $(s',i') \in R$, and thus $s \overset{\alpha}{\dashrightarrow}_S s'$ and $(i',s') \in R^{-1}$.

2. Assume $s \xrightarrow{\alpha}_S s'$. Then also $s \overset{\alpha}{\dashrightarrow}_S s'$. From $(s,i) \in R$ it follows that there exists $i \overset{\alpha}{\dashrightarrow}_I i'$ such that $(s',i') \in R$. Thus $(i',s') \in R^{-1}$, and since I is an implementation, we know that $i \xrightarrow{\alpha}_I i'$. □

Example 3.3.3
Strong modal refinement defines refinement hierarchies, as shown for a simple example in Figure 3.3. All depicted MIOs have action signature $(\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ with $\Sigma^{int} = \{\alpha, \beta\}$ and $\Sigma^{in} = \Sigma^{out} = \emptyset$. For simplicity, frames and action signatures are not drawn in this figure. Dotted arrows illustrate refinements: $S \dashrightarrow T$ expresses that S is a strong modal refinement of T. Note that the transitive closure of $\dashrightarrow$ also expresses valid refinements. The topmost MIO allows α or β to happen, which is refined to either only allow α, only allow β, require α and allow β, or require β and allow α (second row, from left to right). The third row consists of implementations that are final elements with respect to strong modal refinement.

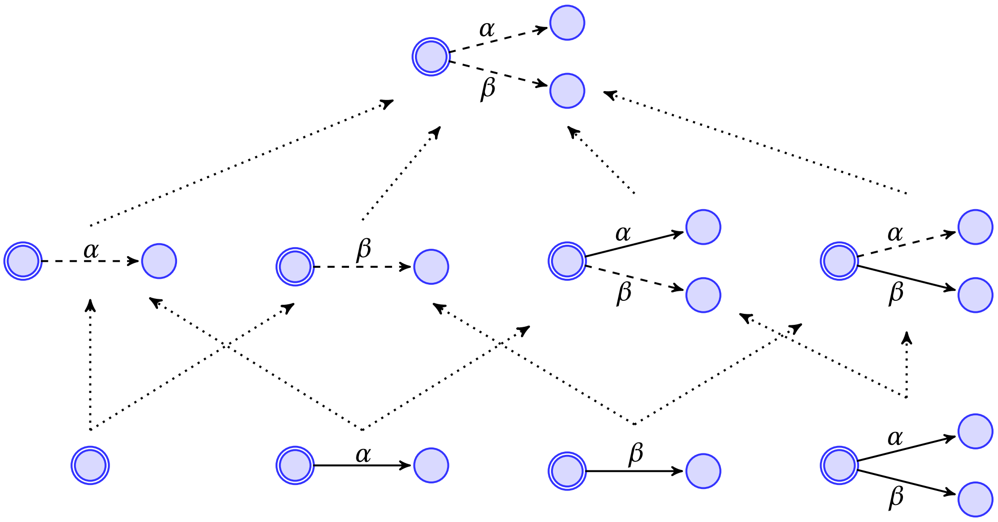


Figure 3.3: Refinement hierarchy induced by $\leq_s$

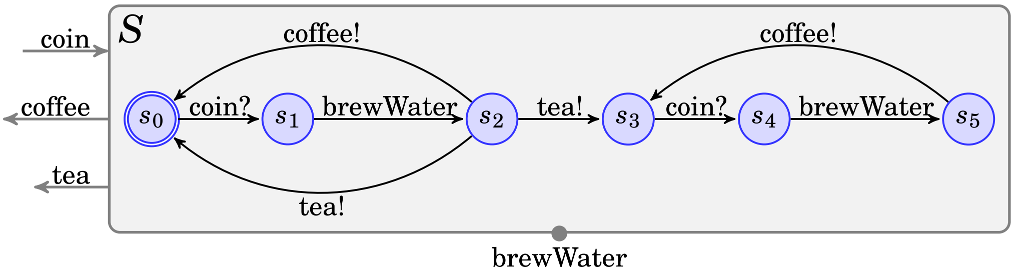


Figure 3.4: A strong modal refinement of the MIO in Figure 3.2

Example 3.3.4
We consider again the simple vending machine specification shown in Figure 3.2. One possible strong modal refinement is shown in Figure 3.4: In state s_2, the machine can decide between coffee and tea, and for tea it can non-deterministically

decide between two transitions, one leading to the initial state and the other one to a state in which the machine will only dispense coffee (not tea anymore). Thus this particular implementation S of T is allowed to eventually stop dispensing tea. Formally, the strong modal refinement $S \leq_s T$ is proven by the refinement relation $\{(s_0,t_0),(s_1,t_1),(s_2,t_2),(s_3,t_0),(s_4,t_1),(s_5,t_2)\}$.

Recall that a MIO is called an implementation whenever the may- and must-transition relations coincide. Also recall that the *implementation semantics* of S, see Section 2.1, consists of all those implementations $I \in \mathbb{MIO}^i$ that refine S:

$$[\![S]\!]_s \triangleq \left\{ I \in \mathbb{MIO}^i \,\middle|\, I \leq_s S \right\}.$$

Example 3.3.5

The implementation semantics of the topmost MIO in Figure 3.3 is the set of MIOs shown in the third row, together with all implementations that are $\approx_s$-equivalent to some MIO in the third row.

As explained in Section 2.1, the implementation semantics leads to the notion of consistency: a MIO S is consistent if $[\![S]\!]_s \neq \emptyset$. Since we assumed that MIOs satisfy $\longrightarrow \subseteq \dashrightarrow$, we obtain an implementation I of S by refining all may-transitions to must-transitions; formally, $\dashrightarrow_I = \longrightarrow_I \triangleq \dashrightarrow_S$. Thus, every MIO is consistent.

Remark 3.3.6

If one is interested in specifying inconsistent MIOs, one can for instance drop the requirement $\longrightarrow \subseteq \dashrightarrow$ which yields mixed transition systems [7]. Another possibility is to explicitly introduce distinguished inconsistent states as it is done, e.g., in the work of Gössler and Raclet [90] under the name of pseudo-modal specifications.

In Section 2.1 we have introduced a derived “thorough” notion of refinement: A MIO S thoroughly refines another MIO T (with the same action signature) if every implementation of S is also an implementation of T. As a consequence of Theorem 2.1.2 in Section 2.1, strong modal refinement implies thorough refinement.

The question of completeness of strong modal refinement was first investigated, to our best knowledge, by Huth in [108]. More recently, Beneš et al. [32] proved that strong modal refinement is complete if the abstract MIO is deterministic. An example that, in general, thorough refinement does not imply strong modal refinement can be found in [32].

Theorem 3.3.7 (Relative completeness of strong modal refinement)

Let $S,T \in \mathbb{MIO}$ with the same action signature and assume that T is deterministic. Then $[\![S]\!]_s \subseteq [\![T]\!]_s$ implies $S \leq_s T$.

Note that, in particular, refinement is also complete if only deterministic MIOs and deterministic implementations are considered. Recently, it has been shown that deciding thorough refinement for *finite* modal transition systems is EXPTIME-complete [31] and since strong modal refinement can be computed in polynomial time by a greatest fixed-point computation (similarly as in the case of strong bisimulation [115]), strong modal refinement is widely accepted as the choice for practical applications.

Remark 3.3.8
Interestingly, modal synchronous composition of MIOs, introduced in Section 3.2, is an over-approximation in the sense of the implementation semantics: $[\![S \otimes T]\!]_s \supsetneq \{I \otimes J \mid I \in [\![S]\!]_s, J \in [\![T]\!]_s\}$. *See [32, 28] for more details and discussion on this issue.*

In what follows, we show that strong modal refinement is compositional: If S and T are composable MIOs and we replace S and T by strong modal refinements S' and T', respectively, then S' and T' are composable and the refined system $S' \otimes T'$ is a strong modal refinement of $S \otimes T$. In [126] the claim was shown for general static constructs on modal processes which instantiates also to modal synchronous composition of MIOs.

Theorem 3.3.9 (Compositional refinement)
Let $S, S', T, T' \in \mathbb{MIO}$. *If* $S' \leq_s S$ *and* $T' \leq_s T$ *and* S *and* T *are composable, then* S' *and* T' *are composable and* $S' \otimes T' \leq_s S \otimes T$.

Proof. It is sufficient to prove the claim for $T = T'$. S' and T are composable because action signatures are not changed during strong modal refinement. The construction of the desired refinement relation is straightforward: If R_S is the relation demonstrating the refinement $S' \leq_s S$, then the relation $R = \{((s',t),(s,t)) \mid t \in St_T, (s',s) \in R_S\}$ proves the refinement $S' \otimes T \leq_s S \otimes T$. □

3.3.2 Strong Environment Correctness

In this section, we introduce our first environment correctness notion for MIOs. Following the approach of [124] which is inspired by interface compatibility of interface automata [61], we consider as a communication error when an output can be sent by the specification, however, the environment is not required to be able to perform the corresponding input which is the case when there is no must-transition enabled with that input action. *Strong environment correctness* between a specification (given as a MIO) and an environment (also itself a MIO) is consequently the absence of any communication error in the above sense in any reachable state of the composition of specification and environment under consideration.

Strong environment correctness is based on the compatibility notion in [124], however, our notion is firstly not symmetric (outputs of the environment are not considered) and secondly adapted to the pessimistic view on compatibility: E is a correct environment for a specification S if $E \otimes F$ is also a correct environment for S for *all* F that are composable with E (as opposed to *some* in the optimistic approach of [124]).

Another important point here is, for this first version of environment correctness, that the environment must be immediately ready to receive the output, as opposed to a relaxed form in which the environment may do some steps in between, before receiving the output. The former one, strong environment correctness, is shown to be a suitable environment correctness relation for MIOs with strong modal refinement, the latter is referred to by *weak environment correctness* in the following, and formally defined in Section 3.4.2 for MIOs with weak modal refinement.

For given MIOs S and E, E is a *strongly correct environment* of S whenever no communication errors in $S \otimes E$ are reachable that are caused from outputs of S which are inputs of E.

Definition 3.3.10 (Strong environment correctness)
Let $S, E \in \mathbb{MIO}$ be composable. E is a strongly correct environment *for S, written $S \rightarrow_s E$, if for all reachable states (s,e) in $S \otimes E$, for all $\alpha \in \Sigma_S^{out} \cap \Sigma_E^{in}$,*

$$\textit{if } s \overset{\alpha!}{\dashrightarrow}_S, \textit{ then } e \xrightarrow{\alpha?}_E .$$

Strong environment correctness only requires that, whenever a shared output is enabled, then in the environment the corresponding input must be enabled as well. This, of course, does not imply that the synchronization actually happens (in case other actions are interleaved). However, one can show that states with outputs enabled guarantee a certain deadlock-freedom, more precisely, one can show that this state is eventually left. In order to prove this, we have to assume *finite* state transition systems (i.e. finitely many states and finitely many transitions) and *strong fairness* [130], i.e. we only consider *fair* computations: A computation is said to be fair if it is finite or if every transition which is enabled infinitely many times is also taken infinitely many times. Then, strong environment correctness implies that, when a MIO is in a state with an output enabled, the state is eventually left.

Theorem 3.3.11 (Preservation of strong environment correctness)
Let $S, S', E, E' \in \mathbb{MIO}$. If $S \rightarrow_s E$ and $S' \leq_s S$ and $E' \leq_s E$, then $S' \rightarrow_s E'$.

Proof. Assume that $S \rightarrow_s E$ and $S' \leq_s S$ with refinement relation R_S and $E' \leq_s E$ with refinement relation R_E. Let (s', e') be a reachable state in $S' \otimes E'$ such that $s' \overset{\alpha!}{\dashrightarrow}_{S'}$ with $\alpha \in \Sigma_S^{out} \cap \Sigma_E^{in}$. Since $S' \leq_s S$ and $E' \leq_s E$, there exist states $s \in St_S$ and $e \in St_E$ such that $(s', s) \in R_S$ and $(e', e) \in E$ and (s, e) reachable in $S \otimes E$. It follows that $s \overset{\alpha!}{\dashrightarrow}_S s$ and hence by $S \rightarrow_s E$ and reachability of (s, e), $e \xrightarrow{\alpha?}_E$. Again by refinement $(e', e) \in R_E$ we get that $e' \xrightarrow{\alpha?}_{E'}$ which was to be shown. □

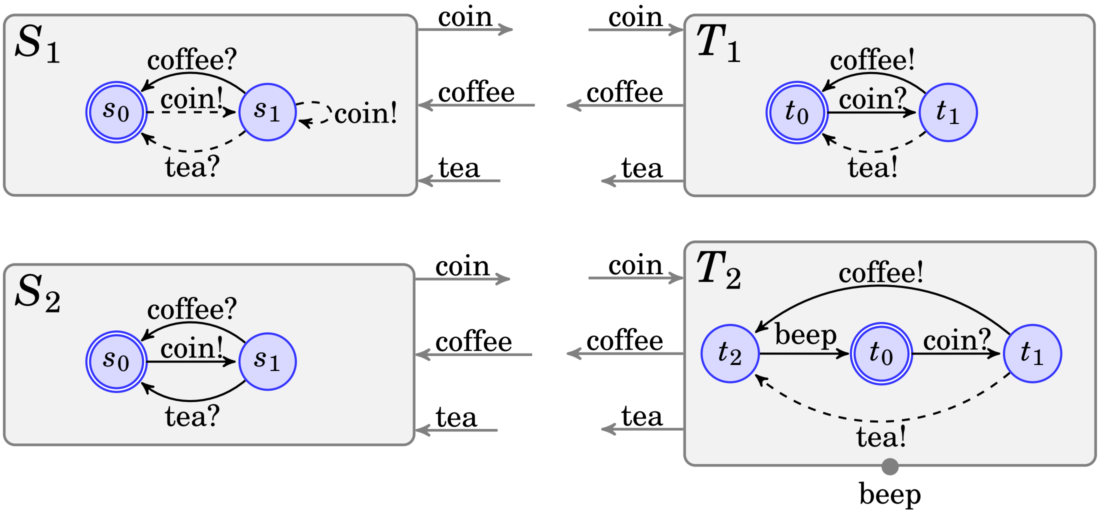


Figure 3.5: MIOs S_1, S_2 specifying a client, and T_1, T_2 specifying a vending machine

Example 3.3.12
Figure 3.5 illustrates strong environment correctness with four MIOs. On the left hand side, S_1 and S_2 specify two clients of the vending machine that can insert coins into the machine and then take out either coffee or tea. Observe that $S_2 \leq_s S_1$, i.e. S_1 is more abstract than S_2 since the client may not insert a coin at all, can throw more than one coin into the machine, and may not accept a tea. On the right hand side, T_1 and T_2 specify two vending machines, with T_2 implementing a beep after dispensing a drink and before accepting the next coin. Note that $T_2 \not\leq_s T_1$ and $T_1 \not\leq_s T_2$ because T_2 has an additional internal action.

We can see that both $S_2 \rightarrow_s T_1$ and $T_1 \rightarrow_s S_2$ hold: in all reachable states of $S_2 \otimes T_1$, for all enabled (shared) output may-transitions there exist corresponding input must-transitions in the other automaton. However, $S_1 \not\rightarrow_s T_1$ because (s_1, t_1) is a reachable state in $S_1 \otimes T_1$ and the optional transition with output coin! *is not accepted by T_1, and $T_1 \not\rightarrow_s S_1$ because in the same reachable state (s_1, t_1) the optional output* tea! *of T_1 is not required to be accepted by S_1. It is satisfied that $T_2 \rightarrow_s S_2$, however, we have that $S_2 \not\rightarrow_s T_2$: in the reachable state (s_0, t_2), T_2 is not immediately ready to accept the next* coin! *sent by S_2.*

3.3.3 Definition of $Th^{\mathbb{MIO}}_{strong}$ and $Th^{d\mathbb{MIO}}_{strong}$

Having established preservation of environment correctness enables us to ultimately define our first specification theories for MIOs:

Corollary 3.3.13
$Th^{\mathbb{MIO}}_{strong} \triangleq (\mathbb{MIO}, \mathbb{MIO}^i, \leq_s, \otimes, \rightarrow_s)$ *is a specification theory.*
$Th^{d\mathbb{MIO}}_{strong} \triangleq (d\mathbb{MIO}, d\mathbb{MIO}^i, \leq_s, \otimes, \rightarrow_s)$ *is a specification theory.*

Proof. We show that $Th^{\mathbb{MIO}}_{strong}$ satisfies all the required properties of a specification theory, see Definition 2.1.1 in Section 2.1.

- $\mathbb{MIO}^i$ is a (proper) subset of $\mathbb{MIO}$.
- Strong modal refinement $\leq_s$ is reflexive and transitive.
- Modal synchronous composition $\otimes$ was shown in Lemma 3.2.2 to be commutative and pseudo-associative.
- Strong environment correctness $\rightarrow_s$ implies composability: whenever we have $S \rightarrow_s E$, then $S \otimes E$ is defined; see Definition 3.3.10.
- Compositional refinement (A1) is shown in Theorem 3.3.9.
- Preservation of environment correctness (A2) is shown in Theorem 3.3.11.
- Finality of implementations (A3) has been shown in Lemma 3.3.2.

This finishes the proof that $Th^{\mathbb{MIO}}_{strong}$ is a specification theory. Also $Th^{d\mathbb{MIO}}_{strong}$ is a specification theory as (A1), (A2) and (A3) also hold for deterministic MIOs; note that $\otimes$ applied to deterministic MIOs yields a deterministic MIO again. □

Since the two specification theories $Th^{d\mathbb{MIO}}_{strong}$ and $Th^{\mathbb{MIO}}_{strong}$ only differ in the set of specificiations and implementations, we can define a reflective embedding of $Th^{d\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIO}}_{strong}$:

Corollary 3.3.14
The identity function $id : d\mathbb{MIO} \rightarrow \mathbb{MIO}$ is a reflective embedding of $Th^{d\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIO}}_{strong}$.

Proof. Clearly, the identity function is injective. All other conditions of Definition 2.1.5 are trivial as well. □

3.4 The Specification Theory $Th^{\mathbb{MIO}}_{weak}$

3.4.1 Weak Modal Refinement

Similar to the step from simulation to weak simulation taking into account silent moves [142, 88] one can introduce a notion of modal refinement that treats transitions labelled by internal actions as non-observable steps (or traditionally called τ-transitions). Such an observational refinement for modal transition systems was already proposed in 1989 by Hüttel and Larsen [109] under the name of weak modal refinement. This notion is the generalization of strong modal refinement [126], similar to the generalization of bisimulation for labelled transition systems to weak bisimulation [142].

Weak modal refinement requires that every must-transition in the abstract specification must be simulated in the concrete specification, possibly enclosed by must-transitions with internal actions. May-transitions in the concrete specification must be simulated analogously, modulo internal actions, in the abstract specification. In contrast to strong modal refinement, names of internal actions do not matter (cf. Section 3.1) and thus there is no restriction on the set of internal actions of refined and refining MIO; the refining MIO can, in particular, introduce new internal actions.

Definition 3.4.1 (Weak modal refinement)
Let $S, T \in \mathbb{MIO}$ with the same sets of input and output actions. S is a weak modal refinement *of T, denoted $S \leq_w T$, if there exists a* refinement relation *$R \subseteq St_S \times St_T$ such that $(s_0, t_0) \in R$, and for all $(s,t) \in R$, for all $\alpha \in (\bigcup \Sigma_S) \cup (\bigcup \Sigma_T)$, the following conditions are satisfied:*

1. *For any $s' \in St_S$, if $s \overset{\alpha}{\dashrightarrow}_S s'$ and*
    - *$\alpha \in \Sigma^{int}_S$ then there exists $t' \in St_T$ such that $t \overset{\tau^*}{\dashrightarrow}_T t'$ and $(s',t') \in R$,*
    - *$\alpha \in \Sigma^{ext}_S$ then there exists $t' \in St_T$ such that $t \overset{\tau^*\alpha\tau^*}{\dashrightarrow}_T t'$ and $(s',t') \in R$.*
2. *For any $t' \in St_T$, if $t \xrightarrow{\alpha}_T t'$ and*
    - *$\alpha \in \Sigma^{int}_T$ then there exists $s' \in St_S$ such that $s \xrightarrow{\tau^*}_S s'$ and $(s',t') \in R$,*
    - *$\alpha \in \Sigma^{ext}_T$ then there exists $s' \in St_S$ such that $s \xrightarrow{\tau^*\alpha\tau^*}_S s'$ and $(s',t') \in R$.*

It is easy to see that weak modal refinement is a preorder, i.e. reflexive and transitive. If S and T do not contain internal transitions, then weak modal refinement coincides with strong modal refinement. If both S and T are implementations, then $S \leq_w T$ is equivalent to weak bisimulation [88] between S and T.

The implementation semantics of $S \in \mathbb{MIO}$ with respect to weak modal refinement is given by $[\![S]\!]_w \triangleq \{I \in \mathbb{MIO}^i \mid I \leq_w S\}$. As a consequence of Theorem 2.1.2 in Section 2.1, $S \leq_w T$ implies $[\![S]\!]_w \subseteq [\![T]\!]_w$.

Remark 3.4.2
The converse direction, $[\![S]\!]_w \subseteq [\![T]\!]_w$ implying $S \leq_w T$, does not hold in general, as it does not hold even for strong modal refinement; see Section 3.3.1. Finding sufficient conditions for MIOs S and T such that $[\![S]\!]_w \subseteq [\![T]\!]_w$ implies $S \leq_w T$ is out of the scope of this thesis. Since relative completeness of strong modal refinement could be shown for deterministic MIOs one possible choice would be to consider notions like weak determinism [101].

Lemma 3.4.3
Every implementation $I \in \mathbb{MIO}^i$ is a final element with respect to weak modal refinement.

Proof. Let $I \in \mathbb{MIO}^i$ and assume another MIO $S \in \mathbb{MIO}$ such that $S \leq_w I$ demonstrated by a refinement relation R. We define a new refinement relation $R^{-1} = \{(i,s) \mid (s,i) \in R\}$, and show that R^{-1} proves $I \leq_w S$. Clearly, $(i_0, s_0) \in R^{-1}$. Let $(i,s) \in R$.

1. Assume $i \overset{\alpha}{\dashrightarrow}_I i'$ and $\alpha \in \Sigma_I^{ext}$. Since I is an implementation, we know that $i \xrightarrow{\alpha}_I i'$. From $(s,i) \in R$ it follows that there exists $s \xrightarrow{\tau^*\alpha\tau^*}_S s'$ such that $(s',i') \in R$, thus $s \overset{\tau^*\alpha\tau^*}{\dashrightarrow}_S s'$ and $(i',s') \in R^{-1}$. The case $\alpha \in \Sigma_I^{int}$ is similar.

2. Assume $s \xrightarrow{\alpha}_S s'$ and $\alpha \in \Sigma_S^{ext}$. Then also $s \overset{\alpha}{\dashrightarrow}_S s'$. From $(s,i) \in R$ it follows that there exists $i \overset{\tau^*\alpha\tau^*}{\dashrightarrow}_I i'$ such that $(s',i') \in R$. Since I is an implementation we know that $i \xrightarrow{\tau^*\alpha\tau^*}_I i'$; moreover, $(i',s') \in R^{-1}$. The case $\alpha \in \Sigma_S^{int}$ is similar. □

Example 3.4.4
We continue our example of the vending machine T (see Figure 3.2) and illustrate weak modal refinement. In Figure 3.6, a MIO S is shown which is a weak modal refinement of T. First of all, in comparison to T the action signature has been extended by new internal actions grindCoffeeBeans *and* beep. *A refinement relation proving $S \leq_w T$ is $\{(s_0,t_0),(s_1,t_1),(s_2,t_2),(s_3,t_2),(s_4,t_0)\}$. Note that state s_3 does not need to offer a transition for the action* tea *since it is a may-transition in the abstract specification T. However, it would not be possible to introduce an internal transition before* tea *in S, since it would have to implement* coffee *according to the requirements of state t_2 in T.*

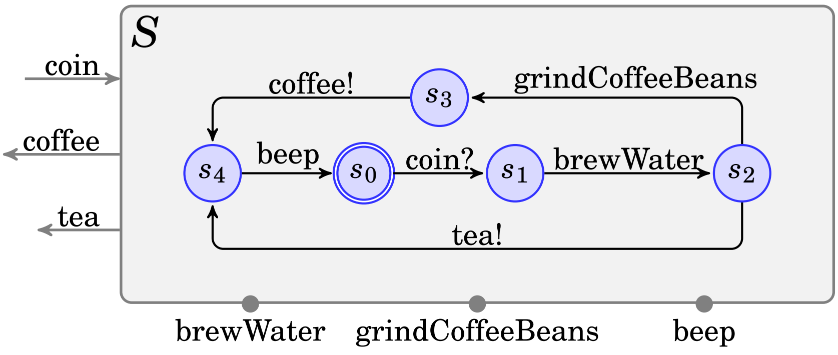


Figure 3.6: A weak modal refinement S of the MIO T in Figure 3.2

Compositionality of weak modal refinement was already shown in [109] for modal transition systems.

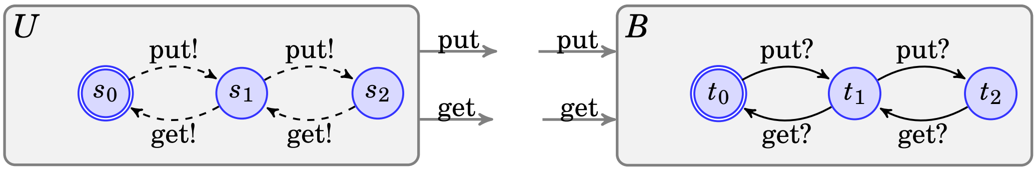


Figure 3.7: A buffer with capacity two and an environment

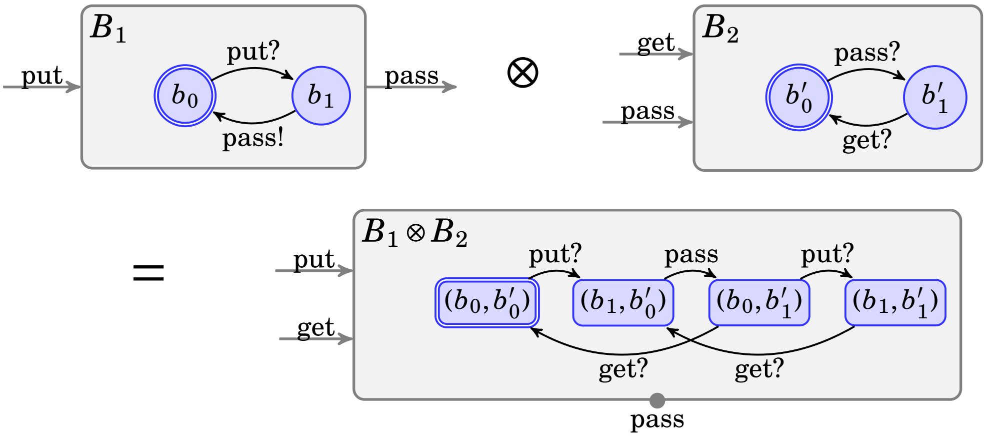


Figure 3.8: Realization of a buffer of size two by two buffers of capacity one

Theorem 3.4.5 (Compositional refinement)
Let $S, S', T, T' \in \mathbb{MIO}$. If $S' \leq_w S$ and $T' \leq_w T$ and S and T are composable, then S' and T' are composable and $S' \otimes T' \leq_w S \otimes T$.

Proof. The proof is along the lines of the proof of compositionality of strong modal refinement: It is sufficient to prove the theorem for the special case $T = T'$. S' and T are composable because the set of external actions are not changed and composability does not depend on names of internal actions (see page 32). If R_S is the relation demonstrating the refinement $S' \leq_w S$, then the relation $R = \{((s', t), (s, t)) \mid t \in St_T, (s', s) \in R_S\}$ proves the refinement $S' \otimes T \leq_w S \otimes T$. □

3.4.2 Weak Environment Correctness

When moving from strong modal refinement to weak modal refinement it is clear that we cannot choose strong environment correctness as our environment correctness notion as weak modal refinement can introduce internal transitions that may defer input transitions. The following examples illustrates this situation.

Example 3.4.6
Figure 3.7 shows a MIO B specifying a buffer with capacity two, and the MIO U describes the correct usage of the buffer. Obviously, B is a strongly correct environment for U.

Let us assume that we would like to realize the buffer by two buffers of capacity one, as shown in Figure 3.8: B_1 accepts elements and then passes them to the second buffer B_2. The removal of elements (action get*) are handled by B_2, once an element has been passed to it. Their composition $B_1 \otimes B_2$ yields the MIO shown in Figure 3.8. It can be easily verified that $B_1 \otimes B_2$ is a weak modal refinement of B. However, $B_1 \otimes B_2$ is not a strongly correct environment for U any more: the state $(s_1,(b_1,b'_0))$ is reachable in $U \otimes (B_1 \otimes B_2)$ and*

$$s_1 \overset{put!}{\dashrightarrow}_U, \quad but\ (b_1,b'_0) \overset{put?}{\not\longrightarrow}_{B_1 \otimes B_2} \ .$$

It is rather clear how we have to adapt our environment correctness notion such that it is preserved by weak modal refinement: the input transition is not required to be enabled immediately, but after some internal must-transitions. In this sense $B_1 \otimes B_2$ is a weakly correct environment *for U: in state (b_1,b'_0), $B_1 \otimes B_2$ has an internal must-transition (labelled with* pass*) to the state (b_0,b'_1) in which* put? *is necessarily enabled.*

Note that it is essential that these internal transitions that defer the input transition are must*-transitions – if they were* may*-transitions, they could be dropped in the next refinement step invalidating preservation of environment correctness.*

Definition 3.4.7 (Weak environment correctness)
Let $S,E \in \mathbb{MIO}$ be composable. E is a weakly correct environment for *S, written $S \rightarrow_w E$, if for all reachable states (s,e) in $S \otimes E$, for all $\alpha \in \Sigma_S^{out} \cap \Sigma_E^{in}$,*

$$if\ s \overset{\alpha!}{\dashrightarrow}_S,\ then\ \xrightarrow{\tau^* \alpha?}_E \ .$$

Remark 3.4.8
There is an even more flexible notion of environment correctness for which all the following results in this thesis equally hold: one can also allow transitions labelled with non-shared output actions *before the relevant input transition. The (symmetric) notion of ultra-weak output-compatibility in [112] follows this idea.*

Similar to the case of strong environment correctness, weak environment correctness guarantees a certain progress. When assuming finite state transition systems and strong fairness [130], whenever a MIO is in a state with an output enabled, the state is eventually left.

We can show now that weak environment correctness is preserved by weak modal refinement, turning it into a suitable environment correctness notion for a specification theory for MIOs based on weak modal refinement.

Theorem 3.4.9 (Preservation of weak environment correctness)
Let $S,S',E,E' \in \mathbb{MIO}$. If $S \rightarrow_w E$ and $S' \leq_w S$ and $E' \leq_w E$, then $S' \rightarrow_w E'$.

Proof. Assume that $S \rightarrow_w E$ and $S' \leq_w S$ with refinement relation R_S and $E' \leq_s E$ with refinement relation R_E. Let (s', e') be a reachable state in $S' \otimes E'$ such that $s' \overset{\alpha!}{\dashrightarrow}_{S'}$ with $\alpha \in \Sigma^{out}_S \cap \Sigma^{in}_E$. Since $S' \leq_w S$ and $E' \leq_w E$, there exist states $s \in St_S$ and $e \in St_E$ such that $(s', s) \in R_S$ and $(e', e) \in E$ and (s, e) reachable in $S \otimes E$. It follows that $s \overset{\alpha!}{\dashrightarrow}_S s$, hence by $S \rightarrow_w E$ there is $e \xrightarrow{\tau^* \alpha?}_E$. From $(e', e) \in R_E$ it follows, by repeated application of the conditions of weak modal refinement for must-transitions, that $e' \xrightarrow{\tau^* \alpha?}_{E'}$ which was to be shown. □

3.4.3 Definition of $Th^{\mathbb{MIO}}_{weak}$

We arrive at a specification theory for MIOs with weak modal refinement and weak environment correctness.

Corollary 3.4.10
$Th^{\mathbb{MIO}}_{weak} \triangleq (\mathbb{MIO}, \mathbb{MIO}^i, \leq_w, \otimes, \rightarrow_w)$ is a specification theory.

Proof. We show that $Th^{\mathbb{MIO}}_{weak}$ satisfies all the required properties of a specification theory, see Definition 2.1.1 in Section 2.1.

- $\mathbb{MIO}^i$ is a (proper) subset of $\mathbb{MIO}$.
- Weak modal refinement $\leq_w$ is reflexive and transitive.
- Modal synchronous composition $\otimes$ was shown in Lemma 3.2.2 to be commutative and pseudo-associative.
- Weak environment correctness $\rightarrow_w$ implies composability: whenever we have $S \rightarrow_s E$, then $S \otimes E$ is defined; see Definition 3.4.7.
- Compositional refinement (A1) is shown in Theorem 3.4.5.
- Preservation of environment correctness (A2) is shown in Theorem 3.4.9.
- Finality of implementations (A3) has been shown in Lemma 3.4.3. □

Finally, we would like to study the relationship between the two specification theories $Th^{\mathbb{MIO}}_{strong}$ and $Th^{\mathbb{MIO}}_{weak}$ introduced in this chapter. Intuitively, the latter specification theory is more flexible because it features observational notions of refinement and environment correctness treating internal actions as non-observable. This can be stated more precisely:

Theorem 3.4.11
The identity function $id : \mathbb{MIO} \rightarrow \mathbb{MIO}$, defined by $\mathrm{id}(S) = S$ for any $S \in \mathbb{MIO}$, is an embedding of $Th^{\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIO}}_{weak}$.

Proof. Carefully inspecting the definition of strong/weak modal refinement and environment correctness one can easily verify that strong modal refinement implies weak modal refinement and strong environment correctness implies weak environment correctness. The other conditions of an embedding are trivial. □

By the above theorem any design with associated refinement and environment correctness proofs in $Th^{\mathbb{MIO}}_{strong}$ also hold in $Th^{\mathbb{MIO}}_{weak}$. Furthermore, the above examples have also shown that defining a *reflective* embedding between the two specification theories is not possible, since weak modal refinement does not imply strong modal refinement and similarly weak environment correctness does not imply strong environment correctness.

We conclude this section by drawing a first fragment of the envisaged hierarchy. Figure 3.9 shows the reflective embedding of $Th^{d\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIO}}_{strong}$ (denoted by $\hookrightarrow\!\!\rhd$) from Section 3.3.3, and the embedding of $Th^{\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIO}}_{weak}$ (denoted by $\hookrightarrow$) shown in Theorem 3.4.11 above.

$$\begin{tikzcd}
Th^{\mathbb{MIO}}_{weak} \\
Th^{\mathbb{MIO}}_{strong} \arrow[u, hook] \\
Th^{d\mathbb{MIO}}_{strong} \arrow[u, hook, -{Triangle[open]}]
\end{tikzcd}$$

Figure 3.9: Strong and weak specification theories for MIOs

3.5 Discussion and Related Work

Our approach is based on modal input/output automata [124], that are in turn based on modal transition systems [126] and extended by the distinction of input, output and internal actions. Using state transition systems with inputs and outputs to model the communication between concurrent, reactive components is in the line with established approaches like I/O automata [133] and interface automata [61]. A key feature that distinguishes it from other works is the use of modalities. May- and must-transitions allow for an elegant and flexible refinement notion supporting the stepwise design of component-based systems.

Expressivity and Refinement. Since the introduction of modal transition systems, several extensions have been proposed in the literature. Disjunctive modal transition systems [127], 1-selecting modal transition systems [79], transition systems with obligations [29], and acceptance automata [156] are all generalizations of modal transition systems offering an increased expressivity with respect to the transitions that need to or are allowed to be enabled in a particular state. For instance, a disjunctive modal transition system can express that a vending machine must output either coffee or tea, but at least one of them. We believe that our approach to environment correctness can be integrated in all of the afore mentioned extensions (provided one distinguishes between input and output actions). However, so far, only modal transition systems offer a notion of weak modal refinement which we think is of great importance for practical examples.

Recently, Beneš et al. [30] proposed parametric modal transition systems with the goal to unify previous attempts to increasing expressivity. Parametric modal transition systems do not only allow loose specification in the sense of local variability, but also permit to specify that in a state with several may-transitions enabled, the choice of transitions which are implemented needs to be persistent for the whole implementation. Parameters encode this choice accordingly. However, it is not clear whether weak modal refinement can be defined for parametric modal transition systems.

As already said, having an observational refinement notion is crucial for practical examples, in particular, if the formalism should support hierarchical components which naturally involve the hiding of actions. To the best of our knowledge, there are two observational refinement notions that have been proposed for modal transition systems: weak modal refinement [109], and an observational implementation relation [83] for modal transition systems that gives rise to a branching modal refinement notion, similar to branching bisimulation [175] for labelled transition systems.

Interestingly, interface automata [61] use alternating refinement (more inputs, less outputs) which actually is observational from concrete to abstract. However, the direction from abstract to concrete is *not* observational which is a severe drawback since invisible actions are usually introduced when going from abstract to concrete. Moreover, as their notion of compatibility is similar to strong environment correctness, inputs cannot be delayed by internal actions in the refinement because internal actions can always be dropped in further refinements. Clearly, the lack of must-modalities is a disadvantage of interface automata in comparison to MIOs. Another shortcoming is that there is no clean notion of an implementation: any interface automaton can always be refined to some automaton without any outputs.

Environment correctness. The idea of our notion of environment correctness, that is outputs must be able to be accepted by the environment, goes back to interface compatibility of interface automata [61]. However, in their work interface compatibility is a symmetric relation between interfaces, and moreover features an optimistic treatment of open non-shared actions. In this thesis, we are following the pessimistic approach that requires that if E is a correct environment of S then E can be further composed with other environments F (that are composable with $S \otimes E$) and still obtain that $E \otimes F$ is a correct environment of S. In [61], the authors deviate from this view by saying that S and T are compatible if there exists an environment of $S \otimes T$ which renders all incompatibilities (i.e. states in which S can issue an output not being accepted by T, or vice versa) in $S \otimes T$ unreachable.

During the completion of this thesis, we have already investigated other, more general versions of environment correctness. Stuck-freedom, introduced by Fournet et al. in [86], is a notion for CCS [142], and intuitively expresses that a processes should not get stuck in the context of other processes whenever there are some transitions still enabled. They introduce a conformance relation extending observational refinement of CCS process, which is a precongruence and preserves stuck-freedom of processes.

We have restricted composition to binary synchronous communication only, and our notion of environment correctness is strictly bound to this communication schema. Clearly, other forms of communication should be supported in a richer specification theory. Frameworks like BIP [40], Reo [8] or Wright [4] offer quite flexible means to specify the behaviour of components and their connectors. BIP [40] features user-defined interaction models which are sets of possible interactions between components, formally powersets of action names, and allow for modelling, e.g., rendezvouz and (atomic) broadcast communication. Reo [8] is a specification language for the coordination of concurrent processes or components in distributed systems. The focus of Reo is on specifying the behaviour of connectors and their communication patterns rather than on the behaviour of the components that communicate through those connectors. Complex connectors can be built with primitive channel types like synchronous, asynchronous or lossy channels with a well-defined behaviour. Different formal semantics of Reo connectors has been proposed, for instance, in [9] a coalgebraic semantics of Reo connectors in terms of relations on timed data streams is defined. In [11] Reo connectors are translated to constraint automata which are transition systems with data, and transitions are equipped with names of participating components and constraint equations that must be satisfied by the data items involved in the communication. Both for BIP and Reo, refinement notions (on the level of formal semantics) only cover observational equivalences (since no modalities are involved). Explicit notions of compatibility like our environment correctness are not studied – they are left to the level of formal semantics and suitable logics to express communication properties. In Wright [4], architectural component connectors are given a semantics by specifying them with CSP [107]. Wright focuses not only on the behavioural description of connectors, but also defines architectural compatibility by deadlock freeness of connectors which is formalized by failures-divergence refinement of CSP.

Other related formalisms. Beside interface automata, many other specification formalisms have been proposed to model the behaviour of component-based systems. The following list of related approaches constitute a small selection of formalisms introduced in recent years and is far from being complete, but contains the most relevant ones comparable to our approach. The main distinction from the below approaches to our work is that they all (except of [47, 48]) do not use any kind of modalities which usually delimits refinement to an equivalence relation not explicitly supporting the stepwise refinement methodology.

In 2001, Plasil and Visnovsky introduced behaviour protocols [151, 117] which is a process algebraic-like language for specifying component architecture and the behaviour of communicating components and their operation calls. Behaviour protocols have been extensively used in the SOFA component model [43]. Unique to behaviour protocols is their ability to express operation calls as well as returns, and nestings. In [151] a conformance relation for behaviour protocols is defined that entails substitutability of components. In [1], compatibility questions and possible communication errors during composition are investigated, in particular, they consider an compatibility error called bad activity that is very similar to our notion of strong environment correctness. However, they do not investigate whether their conformance relations preserve compatibility.

The FRACTAL component model [41] uses pNets [12] to model component behaviours. pNets are kind of blueprints for a component assembly with a transducer. The latter is an LTS with transitions labelled with synchronization vectors enabling multiway synchronization. The blueprint nature of pNets simplifies the modelling of hierarchical systems. Along the same way, component-interaction automata [50] are again an extension of LTS, where transitions are labelled with input, output and internal actions. The composition operator is parameterized by sets of (binary) synchronizations. Substitutability properties are shown for an equivalence relation on component-interaction automata. However, environment correctness notions have not been explicitly considered so far.

Finally, we finish our list of related approaches by mentioning the work of Carrez et al. [47, 48] on components with behavioural contracts. For expressing component contracts they enhance a process algebra by may and must modalities for input and output actions. The semantics of the modalities is slightly different to our work, e.g., an input action labelled with the modality *must* is a constraint on the environment and means that the sender is required to (immediately) send the input. This is not possible in our setting, however, in Chapter 7, we achieve similar expressivity by moving from MIOs to modal contracts.

Publication history. The derived notion $\leftrightarrows_w$ that is defined by weak environment correctness $\rightarrow_w$ in both directions (see Section 2.1) coincides with the notion of weak modal compatibility introduced in [27].

3.6 Summary

In this chapter, we have proposed the modal specification theories $Th^{\mathbb{MIO}}_{strong}$, $Th^{d\mathbb{MIO}}_{strong}$ and $Th^{\mathbb{MIO}}_{weak}$. The corresponding environment correctness notions require that for every output the environment must be able to accept it – in the case of strong environment correctness, the input must be immediately possible, and in the case of the novel notion of weak environment correctness, the input must be possible after some internal steps. The latter was the key to being able to define the observational specification theory $Th^{\mathbb{MIO}}_{weak}$ for MIOs based on weak modal refinement. Finally, we have related our three specification theories by (reflective) embeddings.

Chapter 4

Modal Input/Output Automata with Data Constraints

In this chapter we extend the previously introduced specification theories by taking into account the specification of data. For each interface specification of a component, we distinguish between *provided* and *required* state variables. Provided state variables are local to a component, describe the visible data states a component can adopt, and are accessible from the environment. Required state variables belong also to the interface specification of a component, however, they are not related to the data states of the component itself but to the data states the component can observe in its environment. We do not consider *internal* state variables in our approach and stick to abstract control states as before (to allow graphical representation of specifications, and to reduce technicalities), however, the control structure could be equally well specified with internal variables.

We introduce modal input/output automata with data constraints (MIODs) which enhance modal input/output automata with predicates over a state signature. We use a generic propositional language with Boolean connectives to express pre- and postconditions that are added to transition labels describing the admissible data states of a component before and after performing an action.[1] In other words, pre- and postconditions express in which global states the transition is enabled and to which next provided data states the transition is allowed to lead, respectively. Following the loose approach of modal transition systems, postconditions are also allowed to be loose in the sense that, for one global data state, there may be possible many different next provided data states.

On this basis we study modal synchronous composition, strong and weak modal refinement and strong and weak environment correctness of MIODs. In

[1] Note that MIODs form our syntactic level of specifications. As already noted above, the control structure of the transition systems can as well be specified with purely symbolic methods using the underlying propositional language, e.g. by guarded commands [70] adapted to actions and modalities.

addition to relationships between control states, we take special care of the relationships between data constraints in all these cases. For the verification of strong modal refinement, we also propose a predicate abstraction technique that can reduce the state space of large finite specifications. In the case of infinite-domain variables, rendering the refinement problem undecidable in general, predicate abstraction serves as a sound verification technique. We show that MIODs again form a specification theory in which our previous specification theories for MIOs can be embedded.

We also define a denotational semantics of implementations of MIODs, formally given by input/output automata with data states (IODs) which have a provided data state in each state, and transitions are guarded by required data states. Importantly, we prove independent implementabiliy of implementations of MIODs in terms of IODs.

Outline. In Section 4.1 we provide the basic definitions for MIODs, followed by parallel composition of MIODs in Section 4.2. The specification theories $Th^{\mathbb{MIOD}}_{strong}$ and $Th^{d\mathbb{MIOD}}_{strong}$ based on strong modal refinement and strong environment correctness are defined in Section 4.3. Predicate abstraction is discussed in Section 4.4. In Section 4.5 the denotational semantics of implementations is defined in terms of IODs. The specification theory $Th^{\mathbb{MIOD}}_{weak}$ based on weak modal refinement and weak environment correctness is defined in Section 4.6. Finally, in Section 4.7, we summarize related work and we finish this chapter with a short summary in Section 4.8.

4.1 Definition

We start by introducing state variables which are the basis for modelling concrete data states and for defining state and transition predicates which appear as pre- and postconditions on the transitions of MIODs.

In the following we assume given two disjoint global sets LV of logical variables and SV of state variables. We also assume a predefined data universe $\mathcal{U}$.

Extended action signatures. Actions are extended by formal parameters which are used to pass values along with actions. An *extended action signature* is a triple $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ with each action $a \in \bigcup \Sigma$ having an associated set of formal parameters $par(a) \subseteq \mathrm{LV}$. Equality of actions, so far based on the action name, is extended to require also equality of formal parameters. Analogously to action signatures, extended action signatures Σ_1 and Σ_2 are composable if $(\bigcup \Sigma_1) \cap (\bigcup \Sigma_2) = (\Sigma_1^{in} \cap \Sigma_2^{out}) \cup (\Sigma_1^{out} \cap \Sigma_2^{in})$.

State signatures. In order to model data states and to equip transitions with pre- and postconditions we use state variables of two kinds, which both belong to the given global set SV of state variables.

- *Provided* state variables describe the local, externally visible data states a component can adopt.
- *Required* state variables are used to refer to the data states a component expects to be visible in its environment.

Definition 4.1.1 (State Signature)
A state signature *is a pair* $V = (V^{prov}, V^{req})$ *in which* $V^{prov} \subseteq \mathrm{SV}$ *and* $V^{req} \subseteq \mathrm{SV}$ *are disjoint sets of provided and required state variables, respectively, such that* $V^{prov} \cup V^{req} = \mathrm{SV}$.

In the rest of this chapter, we stipulate the assumption that every component can observe any visible state variables in its environment, hence for any state signature $V = (V^{prov}, V^{req})$ we assume that $V^{prov} \cup V^{req} = \mathrm{SV}$, see Definition 4.1.1. This assumption admits considerable technical simplifications, however, our approach easily extends to the general setting where one allows state signatures (V^{prov}, V^{req}) with $V^{prov} \cup V^{req} \subsetneq \mathrm{SV}$.

Remark 4.1.2
We do not explicitly include internal variables in state signatures. The control states given by modal input/output automata can be considered as the internal states a component can exhibit. Of course, those control states could be very well specified syntactically with (deterministic) transition predicates relating internal data states. This approach is followed, e.g., by de Alfaro et al. in [58] where they extend interface automata by variables and the control structure is specified by transition predicates over internal variables.

State signatures of different components can be composed if their sets of provided state variables do not overlap, as we require that every state variable in SV is controlled by at most one component. More formally, two state signatures $V_1 = (V_1^{prov}, V_1^{req})$ and $V_2 = (V_2^{prov}, V_2^{req})$ are *composable* if $V_1^{prov} \cap V_2^{prov} = \emptyset$. The composition of composable state signatures is defined as follows.

Definition 4.1.3 (Composition of state signatures)
Let $V_1 = (V_1^{prov}, V_1^{req})$ *and* $V_2 = (V_2^{prov}, V_2^{req})$ *be two composable state signatures. The* composition of V_1 and V_2 *is defined as the state signature*

$$V_1 \otimes V_2 \triangleq (V^{prov}, V^{req})$$

with

$$\begin{aligned} V^{prov} &= V_1^{prov} \cup V_2^{prov}, \\ V^{req} &= \left(V_1^{req} \cup V_2^{req}\right) \setminus V^{prov}. \end{aligned}$$

Observe that in the composition $V_1 \otimes V_2$ of the composable state signatures V_1 and V_2 the set of provided state variables are defined by the (disjoint) union of the provided state variables of V_1 and V_2; the set of required state variables of $V_1 \otimes V_2$ are determined by the set of all required state variables of V_1 and V_2 that are neither provided state variables of V_1 nor V_2.

Predicates on states. We use a generic, basic framework to deal with predicates and states. For any sets $\mathrm{W}, \mathrm{W}' \subseteq \mathrm{SV}$ of state variables and set $X \subseteq \mathrm{LV}$ of logical variables, we assume a set $\mathscr{S}(\mathrm{W}, X)$ of *state predicates* and a set $\mathscr{T}(\mathrm{W}, \mathrm{W}', X)$ of *transition predicates*. State predicates, often denoted by φ, refer to single states and transition predicates, often denoted by π, to pairs of states (pre- and poststates). We require that $\mathscr{S}(\mathrm{W}, X)$ and $\mathscr{T}(\mathrm{W}, \mathrm{W}', X)$ are monotonic w.r.t. set inclusion in all arguments, and that both sets are closed under the usual logical connectives like negation ($\neg$) and conjunction ($\wedge$) as well as derived connectives like implication ($\Rightarrow$).

Data states and satisfaction relation. For any $\mathrm{W} \subseteq \mathrm{SV}$, we define the set $\mathscr{D}(\mathrm{W})$ of W-*data states* (or just *data states* if W is clear from the context) to consist of all functions $\delta : \mathrm{W} \to \mathscr{U}$ assigning values from the predefined data universe $\mathscr{U}$ to state variables in W; an element $\delta \in \mathscr{D}(\mathrm{W})$ defines a concrete data state w.r.t. W. For each subset $X \subseteq \mathrm{LV}$, we define the set $\mathscr{D}(X)$ of all valuations $\rho : X \to \mathscr{U}$. The unique data state for an empty set of variables is denoted by ε.

We assume that state predicates $\varphi \in \mathscr{S}(\mathrm{W}, X)$ are equipped with a *satisfaction relation* $(\delta; \rho) \models^X_{\mathrm{W}} \varphi$ for data states $\delta \in \mathscr{D}(\mathrm{W})$ and valuations $\rho \in \mathscr{D}(X)$. If $X = \emptyset$ then we also write $\delta \models^X_{\mathrm{W}} \varphi$. Similarly, for transition predicates $\pi \in \mathscr{T}(\mathrm{W}, \mathrm{W}', X)$ we assume a satisfaction relation $(\delta, \delta'; \rho) \models^X_{\mathrm{W},\mathrm{W}'} \pi$, for two data states $\delta \in \mathscr{D}(\mathrm{W})$ (prestate) and $\delta' \in \mathscr{D}(\mathrm{W}')$ (poststate) and valuation $\rho \in \mathscr{D}(X)$. Super- and subscripts of the satisfaction relation are omitted in the following if they are clear from the context. For $\varphi \in \mathscr{S}(\mathrm{W}, X)$, we write $\models_\forall \varphi$ to express that φ is universally valid, i.e. $(\delta; \rho) \models \varphi$ for all $\delta \in \mathscr{D}(\mathrm{W})$ and all $\rho \in \mathscr{D}(X)$. Similarly, we write $\models_\exists \varphi$ to express that φ is satisfiable, i.e. there exists $\delta \in \mathscr{D}(\mathrm{W})$ and $\rho \in \mathscr{D}(X)$ such that $(\delta; \rho) \models \varphi$. Universal validity and satisfiability of transition predicates are defined analogously. The logical connectives are interpreted as usual, e.g. $(\delta; \rho) \models \varphi_1 \wedge \varphi_2$ if and only if $(\delta; \rho) \models \varphi_1$ and $(\delta; \rho) \models \varphi_2$. We require that the language contains a universally valid state (and transition) predicate *true*. Morever, we assume that every $\delta \in \mathscr{D}(\mathrm{W})$ can be characterized by a predicate $\varphi_\delta \in \mathscr{S}(\mathrm{W}, \emptyset)$, i.e. $\delta \models \varphi_\delta$ and for any $\delta' \in \mathscr{D}(\mathrm{W})$, $\delta' \models \varphi_\delta$ implies $\delta' = \delta$. For any state predicate $\varphi \in \mathscr{S}(\mathrm{W}, \emptyset)$ we write $\mathscr{D}(\varphi)$ for the set $\{\delta \in \mathscr{D}(\mathrm{W}) \mid \delta \models \varphi\}$.

We will frequently use state predicates in combination with transition predicates. Therefore, we require that every state predicate is also a transition predicate where state variables refer to the prestate only; i.e. given a state predicate $\varphi \in \mathscr{S}(\mathrm{W}, X)$, we require that $\varphi \in \mathscr{T}(\mathrm{W}, \mathrm{W}', X)$ for any $\mathrm{W}' \subseteq \mathrm{SV}$, and for all

$\delta \in \mathscr{D}(\mathrm{W})$, all $\delta' \in \mathscr{D}(\mathrm{W}')$ and all $\rho \in \mathscr{D}(X)$, it is satisfied that $(\delta,\delta';\rho) \models^{X}_{\mathrm{W},\mathrm{W}'} \varphi$ if and only if $(\delta;\rho) \models^{X}_{\mathrm{W}} \varphi$. Given a state predicate $\varphi \in \mathscr{S}(\mathrm{W},X)$ and $\mathrm{W}' \subseteq \mathrm{W}$, we write $(\varphi)'$ for the transition predicate in $\mathscr{T}(\mathrm{W}',\mathrm{W},X)$ for which for all $\delta \in \mathscr{D}(\mathrm{W})$, $\delta' \in \mathscr{D}(\mathrm{W}')$ and all $\rho \in \mathscr{D}(X)$, $(\delta;\rho) \models^{X}_{\mathrm{W}} \varphi$ if and only if $(\delta',\delta;\rho) \models^{X}_{\mathrm{W}',\mathrm{W}} (\varphi)'$.

Finally, we require that a satisfaction condition holds, similar to the condition in institutions [91]. For transition predicates π, the satisfaction condition is as follows: For all $\mathrm{W}_1 \subseteq \mathrm{W}'_1 \subseteq \mathrm{SV}$, $\mathrm{W}_2 \subseteq \mathrm{W}'_2 \subseteq \mathrm{SV}$ and $X \subseteq X' \subseteq \mathrm{LV}$, for all $\delta \in \mathscr{D}(\mathrm{W}'_1)$ and $\delta' \in \mathscr{D}(\mathrm{W}'_2)$ and $\rho \in \mathscr{D}(X')$, for all $\pi \in \mathscr{T}(\mathrm{W}_1,\mathrm{W}_2,X)$ it holds that

$$(\delta,\delta';\rho) \models^{X'}_{\mathrm{W}'_1,\mathrm{W}'_2} \pi \quad \text{if and only if} \quad (\delta|_{\mathrm{W}_1},\delta'|_{\mathrm{W}_2};\rho|_X) \models^{X}_{\mathrm{W}_1,\mathrm{W}_2} \pi$$

where $f|_A$ denotes the usual restriction of a function f to a subset A of its definition domain. An analogous satisfaction condition is required for state predicates. The satisfaction condition is implicitly used throughout the proofs in this chapter.

The above definitions are generic and sufficient for the following considerations. Therefore, we do not fix a particular syntax for signatures and predicates here, neither a particular definition of the satisfaction relation. We claim that our notions could be easily instantiated in the context of a particular assertion language based, e.g., on the equational or first-order logic calculus or on set-theoretic notations like in Z [167]. How this would work in the case of the Object Constraint Language (OCL) [148] is sketched in [39].

Example 4.1.4

Our running example, inspired by the example presented in [57], is a simple system consisting of two components modelling a researcher and a vending machine. In short, the researcher can drop coins into the machine's slot, and can request coffee or tea. After drinking the coffee or tea, the researcher can publish a new paper for which she is awarded money from the university.

We start by exemplifying the use of state signatures in our running example. Let us fix the data universe $\mathscr{U} \triangleq \mathbb{Z}$ given by the set of all integers, so all state variables are of integer type and hence data states assign integer values to state variables. We now describe only the researcher component R, the actions and state variables of the vending machine component M is described later.

The extended action signature Σ_R of the researcher component is given by

$$\begin{aligned} \Sigma_R^{in} &= \{coffee, tea\}, \\ \Sigma_R^{out} &= \{publish, coin(x), selectCoffee, selectTea\}, \\ \Sigma_R^{int} &= \{\}. \end{aligned}$$

The set of input actions consists of coffee *and* tea *to pick up a dispensed coffee or tea, respectively. Output actions are* publish *to write and publish a paper,* coin(x) *to drop a coin with value x into the machine's coin slot,* selectCoffee *and* selectTea *(press the coffee and tea button, respectively). The only operation having a formal parameter is* coin(x)*. The state signature V_R is determined by*

$$V_R^{prov} = \{m\},$$
$$V_R^{req} = \{p, c\}.$$

The component R has as a provided state variable m modelling the amount of money the researcher has. Required state variables are p which models the machine's coffee price and c which models the machine's current credit – both are assumed to be visible at the display unit of the vending machine.

Transition Labels of MIODs. We are now able to define the kind of labels which can occur in a modal input/output automaton with data constraints. Given an extended action signature Σ and a state signature V, the set $\mathscr{L}(\Sigma, V)$ consists of expressions of the form

$$[\varphi]a[\pi]$$

where $\varphi \in \mathscr{S}(V^{prov} \cup V^{req}, par(a))$ is the precondition, $a \in \bigcup\Sigma$ is the action, and $\pi \in \mathscr{T}(V^{prov} \cup V^{req}, V^{prov}, par(a))$ is the postcondition.

Preconditions φ are state predicates which can refer to any kind of state variable, i.e. to provided variables local to a component as well as to required variables in the environment. This means that any action of a component can be guarded by a condition which can be checked in an implementation by inspecting the local data state of the component and by querying the data state visible in the environment. Postconditions π express to which provided data states the transition may lead to. Note that π cannot express any constraints on the (uncontrolled) required variables which are owned by the environment. It should be noted that there is no strict need to distinguish between pre- and postconditions, however, we think that it is more convenient to separate conditions on the enabledness of a transition from conditions for the change of data values.

In our previous work [22] the allowed variables in postconditions depended on whether it is a postcondition for an input/internal action or an output action. In the case of input/internal action, the definition in [22] equals our definition here. However, in [22] postconditions for output actions express requirements on required state variables instead of provided ones, i.e. $\pi \in \mathscr{T}(V^{prov} \cup V^{req}, V^{req}, par(a))$. The idea of postconditions referring to required state variables is that once an output is issued to another component, the sender would like to express an assumption on how the receiving component should change its provided state variables. The methodology presented in this thesis stipulates that, in contrast to mixing assumptions and guarantees in a single interface specification, they should be clearly separated from each other. This approach is followed in Chapter 7 where we formalize a theory of contracts. We see in Chapter 7 that applying the ideas to MIODs allows us to express assumptions on state changes in the environment.

The next definition extends modal input/output automata to take into account constraints on data states. The resulting transition systems provide means of loose interface specifications for components with data states. They do not only specify the control flow of behaviours but also the effect on data states in terms of pre- and postconditions.

Definition 4.1.5 (MIOD)
A modal input/output automaton with data constraints (MIOD)

$$(St, s_0, \varphi_0, \Sigma, V, \dashrightarrow, \longrightarrow)$$

consists of a set of control states St, an initial control state $s_0 \in St_S$, an initial data state predicate $\varphi_0 \in \mathscr{S}(V^{prov}, \emptyset)$, an extended action signature $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$, a state signature $V = (V^{prov}, V^{req})$, and a may- and must-transition relation

$$\dashrightarrow, \longrightarrow \subseteq St \times \mathscr{L}(\Sigma, V) \times St.$$

A *state* is a pair (s, δ) consisting of a control state $s \in St$ and a data state $\delta \in \mathscr{D}(V^{prov})$ for the provided state variables V^{prov}. A state (s, δ) is *reachable* if $s = s_0$ and $\delta \models \varphi_0$ or there exists $N \geq 1$ such that for all $0 \leq i < N$ there exists a may-transition

$$s_i \overset{[\varphi_i]\alpha_i[\pi_i]}{\dashrightarrow} s_{i+1}$$

and data states $\delta_i \in \mathscr{D}(V^{prov})$, $\nu_i \in \mathscr{D}(V^{req})$, and $\rho_i \in \mathscr{D}(par(\alpha_i))$ such that

$$(\delta_i \cdot \nu_i, \delta_{i+1}; \rho_i) \models \varphi_i \wedge \pi_i \text{ for all } 0 \leq i < N$$

and $\delta_0 \models \varphi_0$ and $(s, \delta) = (s_N, \delta_N)$.

A MIOD is well-formed if the initial state predicate is satisfiable and all reachable must-transitions are also allowed by the may-transition relation, and postconditions on must-transitions are satisfiable.

Definition 4.1.6 (Well-formed MIOD)
A MIOD $S = (St, s_0, \varphi_0, \Sigma, V, \dashrightarrow, \longrightarrow)$ is well-formed *if*

1. $\models_\exists \varphi_0$ *and*

2. *for all reachable states (s, δ), if there exists*

$$s \xrightarrow{[\varphi]\alpha[\pi]} s'$$

and $(\delta \cdot \nu; \rho) \models \varphi$ for some $\nu \in \mathscr{D}(V^{req})$ and $\rho \in \mathscr{D}(par(\alpha))$, then there exists $\delta' \in \mathscr{D}(V^{prov})$ such that

   (a) $(\delta \cdot \nu, \delta'; \rho) \models \pi$ *and*

(b) *there exists*

$$s \overset{[\varphi']\alpha[\pi']}{\dashrightarrow} s'$$

such that $(\delta \cdot v, \delta'; \rho) \models \varphi' \wedge \pi'$.

A sufficient condition for a MIOD to be well-formed is given in the following lemma: if any must-transition is also a may-transition and if each postcondition of any must-transition is satisfiable, then well-formedness follows. The proof is straightforward.

Lemma 4.1.7
Let S be a MIOD. If

1. $\models_\exists \varphi_0$,
2. $\longrightarrow \subseteq \dashrightarrow$, *and*
3. *for all must-transitions*

$$s \xrightarrow{[\varphi]\alpha[\pi]} s',$$

data states $\delta \in \mathscr{D}(V^{prov})$, $v \in \mathscr{D}(V^{req})$ *and* $\rho \in \mathscr{D}(par(\alpha))$, *if* $(\delta \cdot v; \rho) \models \varphi$, *then there exists* $\delta' \in \mathscr{D}(V^{prov})$ *such that* $(\delta \cdot v, \delta'; \rho) \models \pi$,

then S is well-formed.

Next, we define when a MIOD is an implementation which is the case if, first, there is only one initial provided data state possible and, second, every allowed change of the provided data state is also required by some must-transition.

Definition 4.1.8 (Implementation)
A MIOD $S = (St, s_0, \varphi_0, (\Sigma^{in}, \Sigma^{out}, \Sigma^{int}), (V^{prov}, V^{req}), \dashrightarrow, \longrightarrow)$ *is an* implementation *if*

1. $|\mathscr{D}(\varphi_0)| = 1$ *and*
2. *for any reachable state* (s, δ) *and any may-transition*

$$s \overset{[\varphi]\alpha[\pi]}{\dashrightarrow} s',$$

if $(\delta \cdot v, \delta'; \rho) \models \varphi \wedge \pi$ *for some* $\delta, \delta' \in \mathscr{D}(V^{prov})$, $v \in \mathscr{D}(V^{req})$, $\rho \in \mathscr{D}(par(\alpha))$, *then there exists a must-transition*

$$s \xrightarrow{[\varphi']\alpha[\pi']} s'$$

such that

(a) $(\delta \cdot v, \delta'; \rho) \models \varphi' \wedge \pi'$ *and* *(b) if $\delta'' \in \mathscr{D}(V^{prov})$ is another data state such that $(\delta \cdot v, \delta''; \rho) \models \varphi' \wedge \pi'$ then $\delta'' = \delta'$.*

Again, we can give less complex and sufficient conditions for a MIOD to be an implementation: if there is only one initial provided data state possible, the may-transition relation coincides with the must-transition relation, and every postcondition describes a unique next provided data state, then the MIOD under consideration is an implementation. The proof is straightforward and therefore omitted.

Lemma 4.1.9
Let $S = (St, s_0, \varphi_0, \Sigma, V, \dashrightarrow, \longrightarrow)$ be a MIOD. If

1. $|\mathscr{D}(\varphi_0)| = 1$ *and*

2. $\dashrightarrow\, = \,\longrightarrow$ *and*

3. *for every must-transition*
$$s \xrightarrow{[\varphi]\alpha[\pi]} s'$$
if $(\delta \cdot v; \rho) \models \varphi$ for some $\delta \in \mathscr{D}(V^{prov})$, $v \in \mathscr{D}(V^{req})$, $\rho \in \mathscr{D}(par(\alpha))$, then there exists a unique $\delta' \in \mathscr{D}(V^{prov})$ such that $(\delta \cdot v, \delta') \models \pi$,

then S is an implementation.

Example 4.1.10
We continue our running example of the researcher and the vending machine. In Figures 4.1 and 4.2, MIODs are shown specifying the behaviour of the researcher and the vending machine, respectively.

The state signatures are illustrated by boxes at the bottom frame border; a filled box represents a provided state variable, an open box represents a required state variable. Initial state predicates are written in the upper left corner, below the name of the MIOD. Must-transitions are drawn with solid arrows and may-transitions with dashed arrows. Every must-transition implicitly implies the presence of a may-transition with the same label and source and target state. Preconditions are written above or in front of and postconditions below or after actions. We use a simple language for the predicates, with common arithmetic operations and relations with their usual interpretation. The primed variables in postconditions indicate that we refer to its value in the poststate. Preconditions of the form [true] are omitted. An omitted postcondition stands for a conjunction of predicates $x' = x$ for all provided state variables $x \in V^{prov}$. Moreover, postconditions π which do not mention provided state variables $x_1, \ldots, x_n$ are understood as $\pi \wedge \bigwedge_{i=1}^{n} x_i' = x$. For instance, the vending machine is never allowed to change the price for a coffee (i.e. the value of the provided state variable p never changes).

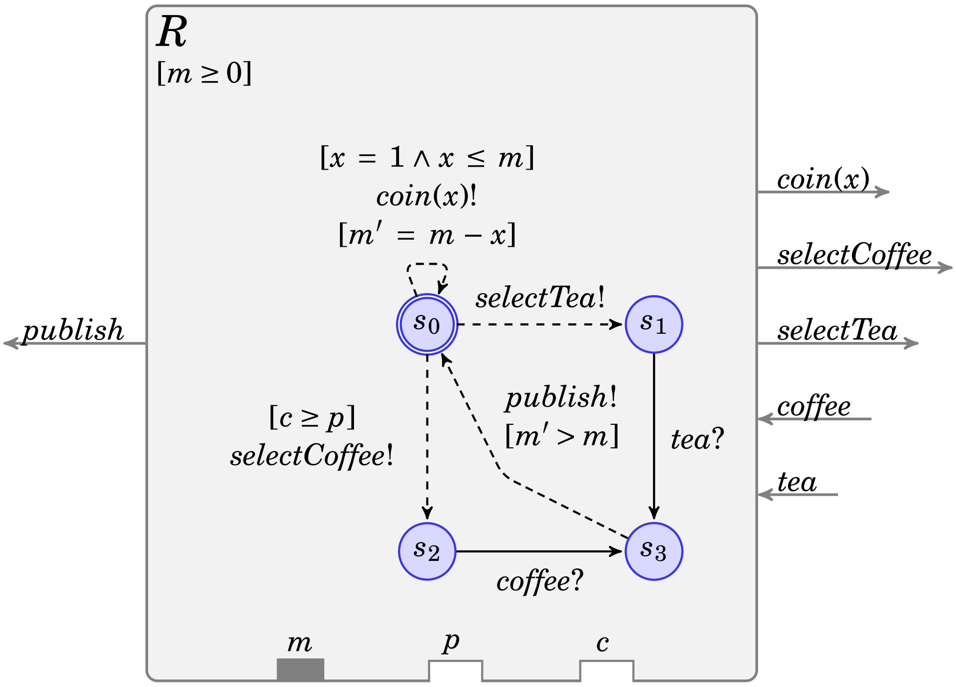


Figure 4.1: MIOD R specifying the behaviour of the researcher

The researcher can throw 1€ *coins into the slot of the machine increasing the credit of the machine by at least* 1€*. When the credit exceeds the coffee price, the researcher may press a button to request a coffee (*selectCoffee!*). The researcher may also press the tea button (*selectTea!*) which is dispensed for free by the machine. After the machine has dispensed either coffee or tea, the researcher may publish a paper for which she receives new money. Note that the vending machine possibly also accepts 2€ coins.*

We use MIODs as representatives of their isomorphism classes w.r.t. bijections on states. The set of those isomorphism classes of all well-formed MIODs is denoted by $\mathbb{MIOD}$, the set of isomorphism classes of all well-formed implementations is denoted by $\mathbb{MIOD}^i$.

4.2 Modal Synchronous Composition

MIODs can be composed to specify the behaviour of concurrent systems of interacting components with data states. The composition operator extends the modal synchronous composition of MIOs, see Section 3.2, to take into account pre- and postconditions.

The modal synchronous composition $S_1 \otimes^d S_2$ of two MIODs S_1 and S_2 synchronizes transitions whose labels refer to shared actions. For instance, a tran-

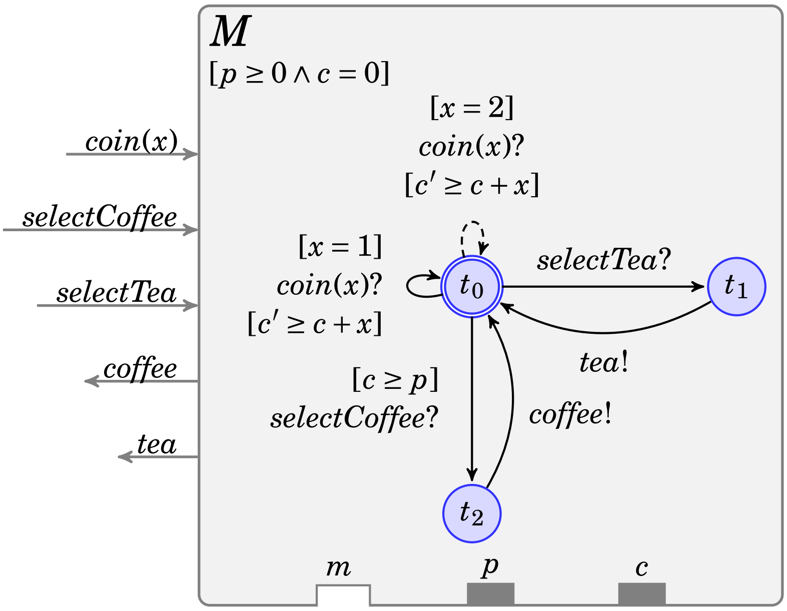


Figure 4.2: MIOD M specifying the behaviour of the vending machine

sition with label $[\varphi_1]\alpha![\pi_1]$ of S_1 is synchronized with a transition with label $[\varphi_2]\alpha?[\pi_2]$ of S_2 which results in a transition with label $[\varphi_1 \wedge \varphi_2]\alpha[\pi_1 \wedge \pi_2]$ where the original preconditions and postconditions are combined by logical conjunction. Transitions whose labels concern shared actions which cannot be synchronized are dropped (as usual) while all other transitions (with non-shared actions) are interleaved in the composition.

Two MIODs S_1 and S_2 are *composable* if their extended action signatures and their state signatures are composable.

Definition 4.2.1 (Modal synchronous composition)
Let $S_i = (St_i, s_{0,i}, \varphi_{0,i}, \Sigma_i, V_i, \dashrightarrow_i, \longrightarrow_i) \in \mathbb{MIOD}$, $i \in \{1,2\}$, *with* $\Sigma_i = (\Sigma_i^{in}, \Sigma_i^{out}, \Sigma_i^{int})$. *The* modal synchronous composition of S_1 and S_2 *is defined by the MIOD*

$$S_1 \otimes^d S_2 \triangleq \left(St_1 \times St_2, (s_{0,1}, s_{0,2}), \varphi_{0,1} \wedge \varphi_{0,2}, \Sigma_1 \otimes \Sigma_2, V_1 \otimes V_2, \dashrightarrow, \longrightarrow\right)$$

where the transition relations $\dashrightarrow$, $\longrightarrow$ *are defined by the following rules:*

$$\frac{s_1 \overset{[\varphi_1]\alpha[\pi_1]}{\dashrightarrow}_1 s_1' \quad \alpha \notin \Sigma_2^{ext}}{(s_1, s_2) \overset{[\varphi_1]\alpha[\pi_1]}{\dashrightarrow} (s_1', s_2)} \qquad \frac{s_1 \xrightarrow{[\varphi_1]\alpha[\pi_1]}_1 s_1' \quad \alpha \notin \Sigma_2^{ext}}{(s_1, s_2) \xrightarrow{[\varphi_1]\alpha[\pi_1]} (s_1', s_2)}$$

$$\frac{s_2 \overset{[\varphi_2]\alpha[\pi_2]}{\dashrightarrow}_2 s_2' \quad \alpha \notin \Sigma_1^{ext}}{(s_1, s_2) \overset{[\varphi_2]\alpha[\pi_2]}{\dashrightarrow} (s_1, s_2')} \qquad \frac{s_2 \xrightarrow{[\varphi_2]\alpha[\pi_2]}_2 s_2' \quad \alpha \notin \Sigma_1^{ext}}{(s_1, s_2) \xrightarrow{[\varphi_2]\alpha[\pi_2]} (s_1, s_2')}$$

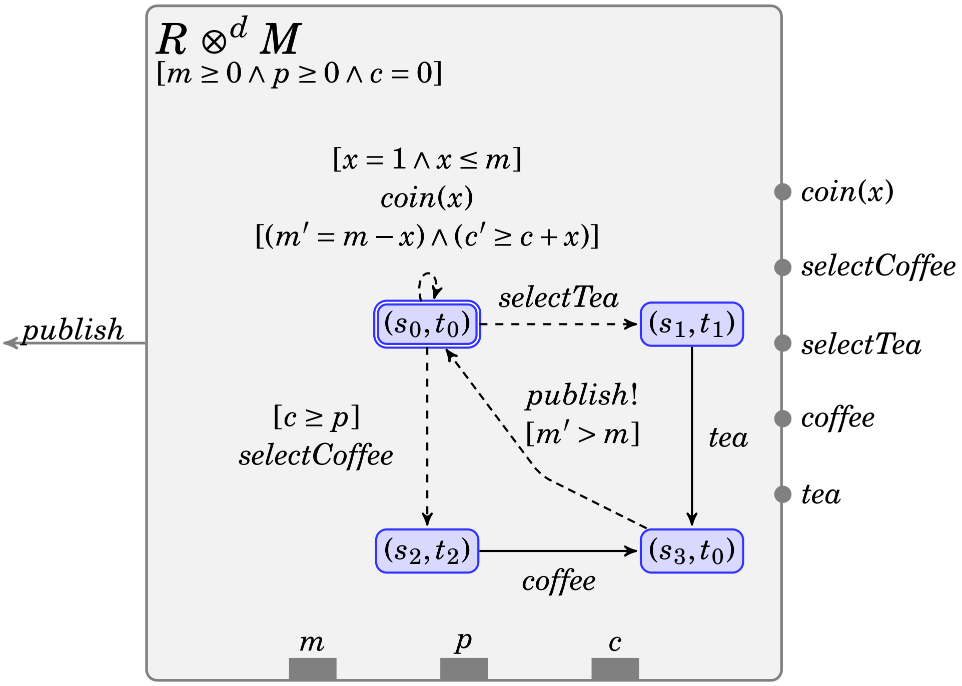


Figure 4.3: Modal synchronous composition of MIODs R and M

$$\frac{s_1 \overset{[\varphi_1]\alpha[\pi_1]}{\dashrightarrow}_1 s_1' \qquad s_2 \overset{[\varphi_2]\alpha[\pi_2]}{\dashrightarrow}_2 s_2'}{(s_1, s_2) \overset{[\varphi_1 \wedge \varphi_2]\alpha[\pi_1 \wedge \pi_2]}{\dashrightarrow} (s_1', s_2')} \qquad \frac{s_1 \xrightarrow{[\varphi_1]\alpha[\pi_1]}_1 s_1' \qquad s_2 \xrightarrow{[\varphi_2]\alpha[\pi_2]}_2 s_2'}{(s_1, s_2) \xrightarrow{[\varphi_1 \wedge \varphi_2]\alpha[\pi_1 \wedge \pi_2]} (s_1', s_2')}$$

Note that the composition is well-defined for composable MIODs S_1 and S_2: it respects the conditions on the state variables of labels that are allowed to occur in pre- and postconditions, and $S_1 \otimes^d S_2$ is well-formed. The proof is straightforward and therefore omitted. Modal synchronous composition is also commutative and pseudo-associative.

Example 4.2.2
Figure 4.3 shows the composition $R \otimes^d M$ of the two abstract specifications R and M of the researcher and vending machine (compare also to Figures 4.1 and 4.2). Pre- and postconditions of synchronized transitions are conjoined. Since all required state variables of R and M are provided by M and R, respectively, we only have provided state variables left. All shared actions become internal actions in the composition. Note also that, concerning the modalities, a transition in the composition labelled with a shared action is only a must-transition if both synchronized input and output transitions were must-transitions (which is the case for the must-transitions labelled with coffee *and* tea*).*

4.3 The Specification Theory $Th^{\mathbb{MIOD}}_{strong}$ and $Th^{d\mathbb{MIOD}}_{strong}$

4.3.1 Strong Modal Refinement

We follow the basic idea of strong modal refinement in which must-transitions of the abstract specification must be respected by the more concrete specification and, conversely, may-transitions of the concrete specification must be allowed by the abstract one. Strong modal refinement for well-formed MIODs S and T takes into account data specifications and is formulated by a relation in $St_S \times St_T \times \mathscr{D}(V^{prov})$, relating the two state spaces together with a provided data state.

Before we formally define when S is a strong modal refinement of T, we give an intuitive explanation. We first discuss the easier direction from concrete to abstract, see Figure 4.4: Any may-transition in S, for which there are data states that satisfy both pre- and postcondition, i.e. $\varphi_S \wedge \pi_S$, must be simulated in T by a may-transition such that the *same* data states also satisfy the respective pre- and postcondition, i.e. $\varphi_T \wedge \pi_T$. In short, any state change allowed in S must be simulated by T.

$$\underbrace{t \overset{[\varphi_T]\alpha[\pi_T]}{\dashrightarrow}_T t' \text{ with } (\delta \cdot \nu, \delta'; \rho) \models \varphi_T \wedge \pi_T}$$

$$\Big\Uparrow \text{ for all } \delta,\ \nu,\ \delta',\ \rho$$

$$\overbrace{s \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow}_S s' \text{ with } (\delta \cdot \nu, \delta'; \rho) \models \varphi_S \wedge \pi_S}$$

Figure 4.4: Illustration of strong modal refinement, from concrete to abstract

The other direction from abstract to concrete concerns must-transitions and is more involved. Figure 4.5 illustrates the situation. Consider a must-transition in T such that the precondition φ_T is satisfied. Then there must be a must-transition in S such that its precondition φ_S is satisfied by the *same* data states, see (1) in Figure 4.5. Now the postcondition π_T on the transition in T specifies possible next provided data states the transition is allowed to end up with. Since refinement must preserve this guarantee to end up in a provided data state satisfying π_T, the chosen transition in S and its respective postcondition π_S should allow *not more* provided data states than π_T; this is illustrated in Figure 4.5 by the implication (2). The pre data state δ as well as the required data state ν and parameter data state ρ (which satisfy both preconditions φ_T and φ_S) are fixed, and then π_S should be stronger than π_T, admitting at most the next provided data states that π_T admits.

$$\begin{array}{ccc}
\underbrace{t \xrightarrow{[\varphi_T]\alpha[\pi_T]}_T t' \text{ with } (\delta\cdot\nu;\rho)\models\varphi_T} & \text{and} & \underbrace{(\delta\cdot\nu,\delta';\rho)\models\pi_T} \\
(1) \Big\Downarrow \text{ for all } \delta,\, \nu,\, \rho & & (2) \Big\Uparrow \text{ for all } \delta' \\
\overbrace{s \xrightarrow{[\varphi_S]\alpha[\pi_S]}_S s' \text{ with } (\delta\cdot\nu;\rho)\models\varphi_S} & \text{and} & \overbrace{(\delta\cdot\nu,\delta';\rho)\models\pi_S}
\end{array}$$

Figure 4.5: Illustration of strong modal refinement, from abstract to concrete

Definition 4.3.1 (Strong modal refinement)
Let $S,T \in \mathbb{MOD}$ with the same extended action signature Σ and the same state signature V. S is a strong modal refinement of T, *denoted $S \leq_s^d T$, if $\models_\forall \varphi_{0,S} \Rightarrow \varphi_{0,T}$ and if there exists a refinement relation*

$$R \subseteq St_S \times St_T \times \mathscr{D}(V^{prov})$$

such that $(s_0,t_0,\delta_0) \in R$ for all $\delta_0 \in \mathscr{D}(\varphi_{0,S})$, and for all $(s,t,\delta) \in R$:

1. *For all $s' \in St_S$, $[\varphi_S]\alpha[\pi_S] \in \mathscr{L}(\Sigma,V)$, $\nu \in \mathscr{D}(V^{req})$, $\rho \in \mathscr{D}(par(\alpha))$ and $\delta' \in \mathscr{D}(V^{prov})$, if*

   $$s \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow}_S s' \text{ and } (\delta\cdot\nu,\delta';\rho)\models\varphi_S \wedge \pi_S$$

   then there exist $t' \in St_T$, $[\varphi_T]\alpha[\pi_T] \in \mathscr{L}(\Sigma,V)$ such that

   $$t \overset{[\varphi_T]\alpha[\pi_T]}{\dashrightarrow}_T t' \text{ and } (\delta\cdot\nu,\delta';\rho)\models\varphi_T \wedge \pi_T$$

   and $(s',t',\delta') \in R$.

2. *For all $t' \in St_T$, $[\varphi_T]\alpha[\pi_T] \in \mathscr{L}(\Sigma,V)$, $\nu \in \mathscr{D}(V^{req})$ and $\rho \in \mathscr{D}(par(\alpha))$, if*

   $$t \xrightarrow{[\varphi_T]\alpha[\pi_T]}_T t' \text{ and } (\delta\cdot\nu;\rho)\models\varphi_T$$

   then there exist $s' \in St_S$, $[\varphi_S]\alpha[\pi_S] \in \mathscr{L}(\Sigma,V)$ such that

   $$s \xrightarrow{[\varphi_S]\alpha[\pi_S]}_S s' \text{ and } (\delta\cdot\nu;\rho)\models\varphi_S$$

   and for all $\delta' \in \mathscr{D}(V^{prov})$,

   $$\text{if } (\delta\cdot\nu,\delta';\rho)\models\pi_S \text{ then } (\delta\cdot\nu,\delta';\rho)\models\pi_T \text{ and } (s',t',\delta') \in R.$$

We show that strong modal refinement is a preorder.

Lemma 4.3.2
Strong modal refinement $\leq_s^d$ is a reflexive and transitive relation.

Proof. Reflexivity is easy to see. To show transitivity, let $S_1, S_2, S_3 \in \mathbb{MIOD}$ with the same action signature Σ and state signature V. Assume that $S_1 \leq^d_s S_2$ and $S_2 \leq^d_s S_3$, witnessed by refinement relations R_{12} and R_{23}, respectively. We define a relation

$$R = \{(s_1, s_3, \delta) \mid \exists s_2 \in St_2 : (s_1, s_2, \delta) \in R_{12}, (s_2, s_3, \delta) \in R_{23}\} \subseteq St_1 \times St_3 \times \mathscr{D}(V^{prov})$$

and show that R is a refinement relation for $S_1 \leq^d_s S_3$. The interesting direction is from abstract to concrete: assume that

$$s_3 \xrightarrow{[\varphi_3]\alpha[\pi_3]}_3 s'_3$$

such that $(\delta \cdot v; \rho) \models \varphi_3$ for some $v \in \mathscr{D}(V^{req})$ and some $\rho \in \mathscr{D}(par(\alpha))$. From $(s_2, s_3, \delta) \in R_{23}$ it follows that there exists a transition

$$s_2 \xrightarrow{[\varphi_2]\alpha[\pi_2]}_2 s'_2$$

such that $(\delta \cdot v; \rho) \models \varphi_2$. From $(s_1, s_2, \delta) \in R_{12}$ it follows that there exists a transition

$$s_1 \xrightarrow{[\varphi_1]\alpha[\pi_1]}_1 s'_1$$

such that $(\delta \cdot v; \rho) \models \varphi_1$. Now, for all $\delta' \in \mathscr{D}(V^{prov})$, if $(\delta \cdot v, \delta'; \rho) \models \pi_1$, then $(\delta \cdot v, \delta'; \rho) \models \pi_2$ and $(s'_1, s'_2, \delta') \in R_{12}$; then also $(\delta \cdot v, \delta'; \rho) \models \pi_3$ and $(s'_2, s'_3, \delta') \in R_{23}$. Altogether, we get that $(\delta \cdot v, \delta'; \rho) \models \pi_1$ implies $(\delta \cdot v, \delta'; \rho) \models \pi_3$ and $(s'_1, s'_3, \delta') \in R$.□

A sufficient condition for strong modal refinement is the following.[2]

Lemma 4.3.3
Let $S, T \in \mathbb{MIOD}$ with the same extended action signature and the same state signature. If $\models_\forall \varphi_{0,S} \Rightarrow \varphi_{0,T}$ and there exists a relation $R \subseteq St_S \times St_T$ such that $(s_0, t_0) \in R$, and for all $(s,t) \in R$,

1. *whenever $t \xrightarrow{[\varphi_T]\alpha[\pi_T]}_T t'$, then there exists $s \xrightarrow{[\varphi_S]\alpha[\pi_S]}_S s'$ such that $\models_\forall \varphi_T \Rightarrow \varphi_S$, $\models_\forall \varphi_T \wedge \pi_S \Rightarrow \pi_T$ and $(s', t') \in R$,*
2. *whenever $s \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow}_S s'$ then there exists $t \overset{[\varphi_T]\alpha[\pi_T]}{\dashrightarrow}_T t'$ such that $\models_\forall (\varphi_S \wedge \pi_S) \Rightarrow (\varphi_T \wedge \pi_T)$ and $(s', t') \in R$.*

Then $S \leq^d_s T$.

[2] Lemma 4.3.3 is also a sufficient condition for strong modal refinement in the sense of [22] if only input actions are considered.

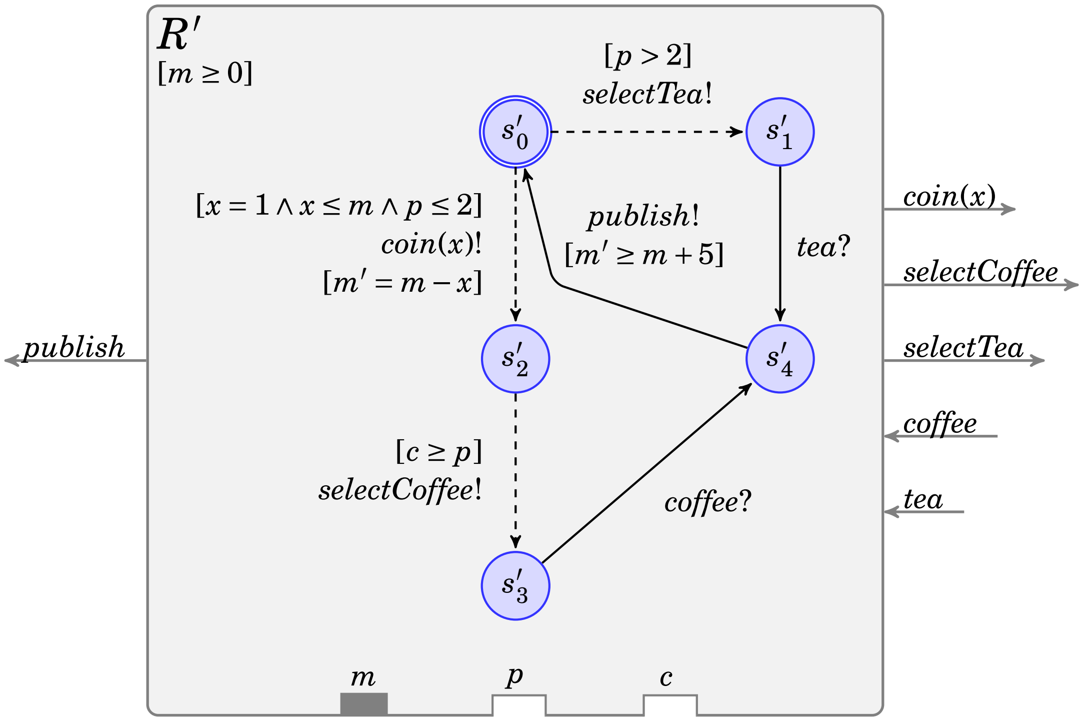


Figure 4.6: MIOD R' specifying a refined behaviour of the researcher

Proof. The refinement relation R' demonstrating $S \leq_s^d T$ can be easily defined by

$$R' = \left\{(s,t,\delta) \,\middle|\, (s,t) \in R, \delta \in \mathscr{D}(V^{prov})\right\}.$$

It is straightforward to show that R' is indeed a refinement relation such that $(s_0, t_0, \delta_0) \in R'$ for all $\delta_0 \in \mathscr{D}(\varphi_{0,S})$. □

Note that if predicates are decidable and the well-formed MIODs S and T are finite,i.e. have finitely many states and transitions, then Lemma 4.3.3 provides a *decidable*, sufficient condition for $S \leq_s^d T$.

Example 4.3.4

The abstract specifications (see Figures 4.1 and 4.2) of the researcher and the vending machine are now refined as shown in Figures 4.6 and 4.7, respectively.

The refined researcher R' may only throw a coin into the machine's slot if she can observe the coffee price to be less or equal than 2€. In this case, she may select and drink a coffee. If the coffee price is greater than 2€, she prefers to drink a tea (free of cost). In any case, after drinking either tea or coffee, she publishes for which she receives new money, at least 5€. To show the refinement $R' \leq_s^d R$ we can make use of Lemma 4.3.3 and propose a relation in $St_{R'} \times St_R$ given by

$$\{(s'_0, s_0), (s'_1, s_1), (s'_4, s_3), (s'_2, s_0), (s'_3, s_2)\}.$$

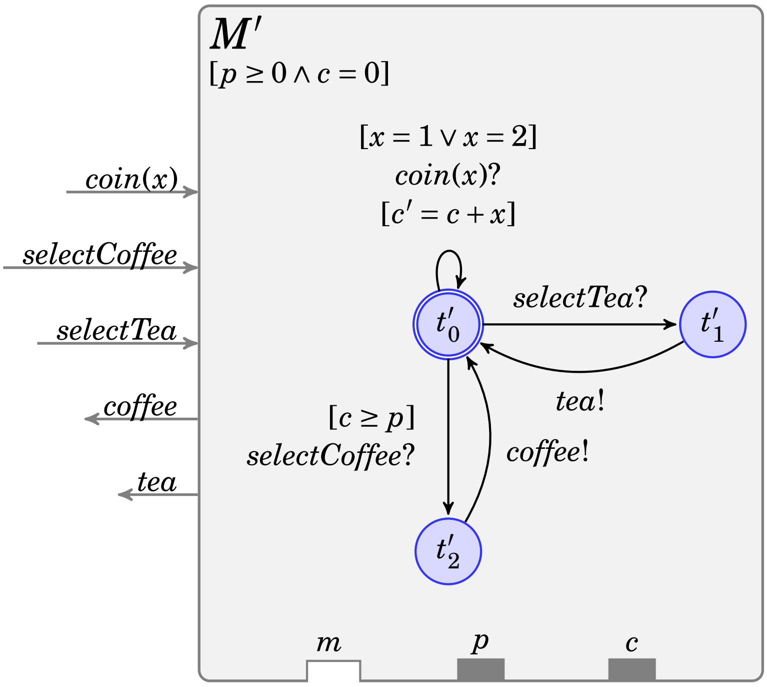


Figure 4.7: MIOD M' specifying a refined behaviour of the vending machine

This relation satisfies all the conditions in Lemma 4.3.3, hence $R' \leq^d_s R$.

The refined vending machine specification M', see Figure 4.7, is strengthening the single may-transition to a must-transition. Again using Lemma 4.3.3, the relation

$$Q = \{(t'_0, t_0), (t'_1, t_1), (t'_2, t_2)\}$$

witnesses $M' \leq^d_s M$. As an example, for the pair $(t'_0, t_0) \in Q$ let us check whether the conditions of Lemma 4.3.3 are satisfied, for instance, consider the must-transition in M

$$t_0 \xrightarrow{[x=1]\ coin(x)?\ [c' \geq c+x]}_M t_0.$$

Then there is a must-transition in M'

$$t'_0 \xrightarrow{[x=1 \vee x=2]\ coin(x)?\ [c'=c+x]}_{M'} t'_0$$

such that

- $\models_\forall x = 1 \Rightarrow (x = 1 \vee x = 2)$,
- $\models_\forall (x = 1 \wedge c' = c + x) \Rightarrow c' \geq c + x$, *and*
- $(t'_0, t_0) \in Q$.

All other state pairs in Q can be verified analogously.

We show that any implementation in $\mathbb{MIOD}^i$ is a final element with respect to strong modal refinement.

Lemma 4.3.5
Let $I \in \mathbb{MIOD}^i$ and $S \in \mathbb{MIOD}$. Then $S \leq_s^d I$ implies $I \leq_s^d S$.

Proof. Assume that $S \leq_s^d I$ is demonstrated by a refinement relation R. Then it is straightforward to prove that $R^{-1} = \{(i,s,\delta) \mid (s,i,\delta) \in R\}$ is a refinement relation demonstrating $I \leq_s^d S$. □

The implementation semantics of $S \in \mathbb{MIOD}$ induced by strong modal refinement is denoted by

$$[\![S]\!]_s^d \triangleq \left\{ I \in \mathbb{MIOD}^i \,\middle|\, I \leq_s^d S \right\}.$$

It is straightforward to prove that $[\![S]\!]_s^d \neq \emptyset$ if and only if S is well-formed.

In Theorem 2.1.2 we have already shown on the abstract level of specification theories that refinement implies thorough refinement which is defined by inclusion of implementation semantics. The other direction only holds if the MIOD on the right hand side is deterministic, which is analogous to the situation found for MIOs. We first define what determinism means for MIODs.

Definition 4.3.6 (Determinism)
A MIOD S is deterministic, if for every reachable state (s,δ), for all $\alpha \in \bigcup\Sigma$, $v \in \mathscr{D}(V^{req})$ and $\rho \in par(\alpha)$,

1. *if there is $s \overset{[\varphi]\alpha[\pi]}{\dashrightarrow} s'$ and $s \overset{[\varphi']\alpha[\pi']}{\dashrightarrow} s''$ such that $(\delta \cdot v;\rho) \models \varphi \wedge \varphi'$, then $s' = s''$,*
2. *if there is $s \xrightarrow{[\varphi]\alpha[\pi]} s'$ and $s \xrightarrow{[\varphi']\alpha[\pi']} s''$ such that $(\delta \cdot v;\rho) \models \varphi \wedge \varphi'$, then $s' = s''$ and for all $\delta' \in \mathscr{D}(V^{prov})$, $(\delta \cdot v, \delta';\rho) \models \pi$ if and only if $(\delta \cdot v, \delta';\rho) \models \pi'$.*

Note that determinism requires for may-transitions only that the next control state is unique. However, for simultaneously enabled must-transitions, we require that they lead to the same next control state and their postconditions are equivalent. Intuitively, the reason for the weak condition on the may-transition relation is that data states are observable and are distinguished by refinement, i.e. two states (s,δ) and (t,δ') of two MIODs S and T, respectively, can only be related if $\delta = \delta'$.

The set of the isomorphism classes (w.r.t. bijections on states) of deterministic, well-formed MIODs is denoted by $d\mathbb{MIOD}$, the set of isomorphism classes for deterministic, well-formed implementations of MIODs is denoted by $d\mathbb{MIOD}^i$.

The following theorem shows that completeness of refinement is achieved once the right hand side is assumed to be deterministic. The proof is an adaptation of the proof for completeness of refinement for (usual) modal transition systems in [32].

Theorem 4.3.7 (Relative completeness of strong modal refinement)
Let $S,T \in \mathbb{MIOD}$, and let T be deterministic. Then $[\![S]\!]^d_s \subseteq [\![T]\!]^d_s$ implies $S \leq^d_s T$.

Proof. In this proof, we write (S,s,φ) for S where the initial location is replaced with $s \in S$, and the initial state predicate by $\varphi \in \mathcal{S}(V^{prov},\emptyset)$. Observe that the assumption $[\![S]\!]^d_s \subseteq [\![T]\!]^d_s$ then means more precisely[3]

$$[\![(S,s_0,\varphi_{\delta_0})]\!]^d_s \subseteq [\![(T,t_0,\varphi_{\delta_0})]\!]^d_s$$

for all $\delta_0 \in \mathcal{D}(V^{prov})$ with $\delta_0 \models \varphi_{0,S}$.

Let $R \subseteq St_S \times St_T \times \mathcal{D}(V^{prov})$ be the smallest relation satisfying

- for all $\delta_0 \in \mathcal{D}(V^{prov})$, if $\delta_0 \models \varphi_{0,S}$, then $(s_0,t_0,\delta_0) \in R$,
- for all $(s,t,\delta) \in R$, $s \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow}_S s'$, $t \overset{[\varphi_T]\alpha[\pi_T]}{\dashrightarrow}_T t'$, $\nu \in \mathcal{D}(V^{req})$, $\rho \in \mathcal{D}(par(\alpha))$, $\delta' \in \mathcal{D}(V^{prov})$, if $(\delta \cdot \nu, \delta'; \rho) \models \varphi_S \wedge \pi_S \wedge \varphi_T \wedge \pi_T$, then $(s',t',\delta') \in R$.

We will show that R is a relation witnessing $S \leq^d_s T$.

First, we prove that $(s,t,\delta) \in R$ implies

$$[\![(S,s,\varphi_\delta)]\!]^d_s \subseteq [\![(T,t,\varphi_\delta)]\!]^d_s. \qquad (\star)$$

For $(s_0,t_0,\delta_0) \in R$ with $\delta_0 \models \varphi_{0,S}$, this holds by assumption. Now, let $(s,t,\delta) \in R$ and assume that all the assumptions of the above second condition hold. Let $I' \in [\![(S,s',\varphi_{\delta'})]\!]^d_s$ with initial state i'_0. Then, since S is well-formed and hence consistent, there exists $I \in [\![(S,s,\varphi_\delta)]\!]^d_s$ with initial state i_0 such that

$$i_0 \xrightarrow{[\varphi_\delta \wedge \varphi_\nu \wedge \varphi_\rho]\alpha[(\varphi_{\delta'})']}_I i$$

and the reachable part from i on is the same as I' (from the initial state i'_0 on), i.e. $(I,i,\varphi_{\delta'}) = I'$. From $(\star)$ it follows that $I \in [\![(T,t,\varphi_\delta)]\!]^d_s$, and since T is deterministic we can conclude that $(I,i,\varphi_{\delta'}) \in [\![(T,t',\varphi_{\delta'})]\!]^d_s$, and then $I' \in [\![(T,t',\varphi_{\delta'})]\!]^d_s$.

We now show that R is a relation witnessing $S \leq^d_s T$. By definition of R, we know that $(s_0,t_0,\delta_0) \in R$ for all $\delta_0 \models \varphi_{0,S}$. Let $(s,t,\delta) \in R$.

1. Assume $s \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow} s'$ and $(\delta \cdot \nu, \delta'; \rho) \models \varphi_S \wedge \pi_S$. Then there exists an implementation
$$I \in [\![(S,s,\varphi_\delta)]\!]^d_s$$
such that
$$i_0 \xrightarrow{[\varphi_\delta \wedge \varphi_\nu \wedge \varphi_\rho]\alpha[(\varphi_{\delta'})']}_I i.$$

[3] Recall that, for any $W \subseteq SV$, we assume that every $\delta \in \mathcal{D}(W)$ can be characterized by a predicate $\varphi_\delta \in \mathcal{S}(W,\emptyset)$, i.e. $\delta \models \varphi_\delta$ and for any $\delta' \in \mathcal{D}(W)$, $\delta' \models \varphi_\delta$ implies $\delta' = \delta$.

By the assertion above, we know that

$$I \in [\![(T,t,\varphi_\delta)]\!]_s^d,$$

hence there exists $t \xrightarrow{[\varphi_T]\alpha[\pi_T]}_T t'$ such that $(\delta\cdot\nu,\delta';\rho)\vDash\varphi_T\wedge\pi_T$. By definition of R, we finally get $(s',t',\delta')\in R$.

2. Assume $t \xrightarrow{[\varphi_T]\alpha[\pi_T]}_T t'$ such that $(\delta\cdot\nu;\rho)\vDash\varphi_T$. Then, for all $I\in[\![(T,t,\varphi_\delta)]\!]_s^d$ there exists
$$i_0 \xrightarrow{[\varphi_\delta\wedge\varphi_\nu\wedge\varphi_\rho]\alpha[(\varphi_{\delta'})']}_I i$$
for some
$$\delta'\in P \triangleq \{\delta'\in\mathscr{D}(V^{prov}) \mid (\delta\cdot\nu,\delta';\rho)\vDash\pi_T\}.$$
By this observation and $[\![(S,s,\delta)]\!]_s^d\subseteq[\![(T,t,\delta)]\!]_s^d$ we know that every implementation of (S,s,φ_δ) has such a transition for some $\delta'\in P$. This can only be the case if there is a must-transition
$$s\xrightarrow{[\varphi_S]\alpha[\pi_S]} s'$$
such that $(\delta\cdot\nu;\rho)\vDash\varphi_S$ and for all $\delta'\in\mathscr{D}(V^{req})$, $(\delta\cdot\nu,\delta';\rho)\vDash\pi_S$ implies $\delta'\in P$. By the definition of R, we can infer $(s',t',\delta')\in R$; this follows from the fact that there exist underlying may-transitions in S and T. □

The next theorem shows that strong modal refinement is a precongruence with respect to modal synchronous composition of MIODs.

Theorem 4.3.8 (Compositional refinement)
Let $S,S',T,T'\in\mathbb{MIOD}$. If $S'\le_s^d S$ and $T'\le_s^d T$ and S, T are composable, then S', T' are composable and $S'\otimes^d T'\le_s^d S\otimes^d T$.

Proof. It suffices to prove the case $T'=T$ since composition is commutative. Definedness of $S'\otimes^d T$ follows from $S\otimes^d T$ and $S'\le_s^d S$ since strong modal refinement does not change action signatures or state signatures. Let $V=(V^{prov},V^{req})=V_S\otimes V_T$. To show $S'\otimes^d T\le_s^d S\otimes^d T$ we define a refinement relation $R\subseteq(St_{S'}\times St_T)\times(St_S\times St_T)\times\mathscr{D}(V^{prov})$ by

$$R=\{(((s',t),(s,t),\delta_S\cdot\delta_T) \mid (s',s,\delta_S)\in R_S\}$$

where R_S witness $S'\le_s^d S$. We show that R proves $S'\otimes^d T\le_s^d S\otimes^d T$.

$((s',t),(s,t),(\delta_{0,S'}\cdot\delta_{0,T}))$ clearly holds for any $\delta_{0,S'}\in\mathscr{D}(\varphi_{0,S'})$ and any $\delta_{0,T}\in\mathscr{D}(\varphi_{0,T})$, because $\vDash_\forall\varphi_{0,S'}\Rightarrow\varphi_{0,S}$. Let

$$((s',t),(s,t),(\delta_S\cdot\delta_T))\in R,$$

so we can assume that $(s', s, \delta_S) \in R_S$. Let

$$(s,t) \xrightarrow{[\varphi]\alpha[\pi]} (\hat{s}, \hat{t}) \tag{1}$$

be a must-transition in $S \otimes^d T$. The only interesting case is when α is a shared action of S and T, i.e. $\alpha \in \Sigma_S^{ext} \cap \Sigma_T^{ext}$. Assume that there is $\nu \in \mathscr{D}(V^{req})$, $\rho \in \mathscr{D}(par(\alpha))$ such that

$$(\delta_S \cdot \delta_T \cdot \nu; \rho) \models \varphi.$$

From (1) and the rules of composition it follows that there exist

$$s \xrightarrow{[\varphi_S]\alpha[\pi_S]} \hat{s} \text{ and } t \xrightarrow{[\varphi_T]\alpha[\pi_T]} \hat{t}$$

such that $\varphi = \varphi_S \wedge \varphi_T$ and $\pi = \pi_S \wedge \pi_T$. From $(s', s, \delta_S) \in R_S$ we can conclude that there exists

$$s' \xrightarrow{[\varphi_{S'}]\alpha[\pi_{S'}]} \hat{s}'$$

such that $(\delta_S \cdot \delta_T \cdot \nu; \rho) \models \varphi_{S'} \wedge \varphi_T$, and for all $\delta'_S \in \mathscr{D}(V^{prov})$, if $(\delta_S \cdot \delta_T \cdot \nu, \delta'_S; \rho) \models \pi_{S'}$, then $(\delta_S \cdot \delta_T \cdot \nu, \delta'_S; \rho) \models \pi_S$ and $(\hat{s}', \hat{s}, \delta'_S) \in R_S$.

Then we have

$$(s', t) \xrightarrow{[\varphi_{S'} \wedge \varphi_T]\alpha[\pi_{S'} \wedge \pi_T]} (\hat{s}', \hat{t})$$

and the rest follows from the above satisfaction statements, including

$$((\hat{s}', \hat{t}), (\hat{s}, \hat{t}), \delta'_S \cdot \delta_T) \in R.$$

The other direction of modal refinement (condition 2 of Def. 4.3.1, from concrete to abstract) is very similar to the proof above. □

Example 4.3.9
Figure 4.8 shows the composition $R' \otimes^d M'$ of the refined system specifications R' and M' (see also Figures 4.6 and 4.7 for their individual specifications). Thanks to Theorem 4.3.8 we can infer $R' \otimes^d M' \leq_s^d R \otimes^d M$ just by verifying the refinements $R' \leq_s^d R$ and $M' \leq_s^d M$.

4.3.2 Strong Environment Correctness

The environment correctness notion that we introduce for MIODs in the following builds upon strong environment correctness as defined for MIOs in Section 3.3.2, Chapter 3. From the control point of view strong environment correctness for MIODs requires that in any reachable state of the product $S \otimes^d E$ of two MIODs S (the specification) and E (the environment), if S *may* issue an output (in its current control and data state) then E is in a control and data state where it *must* be able to take the corresponding input.

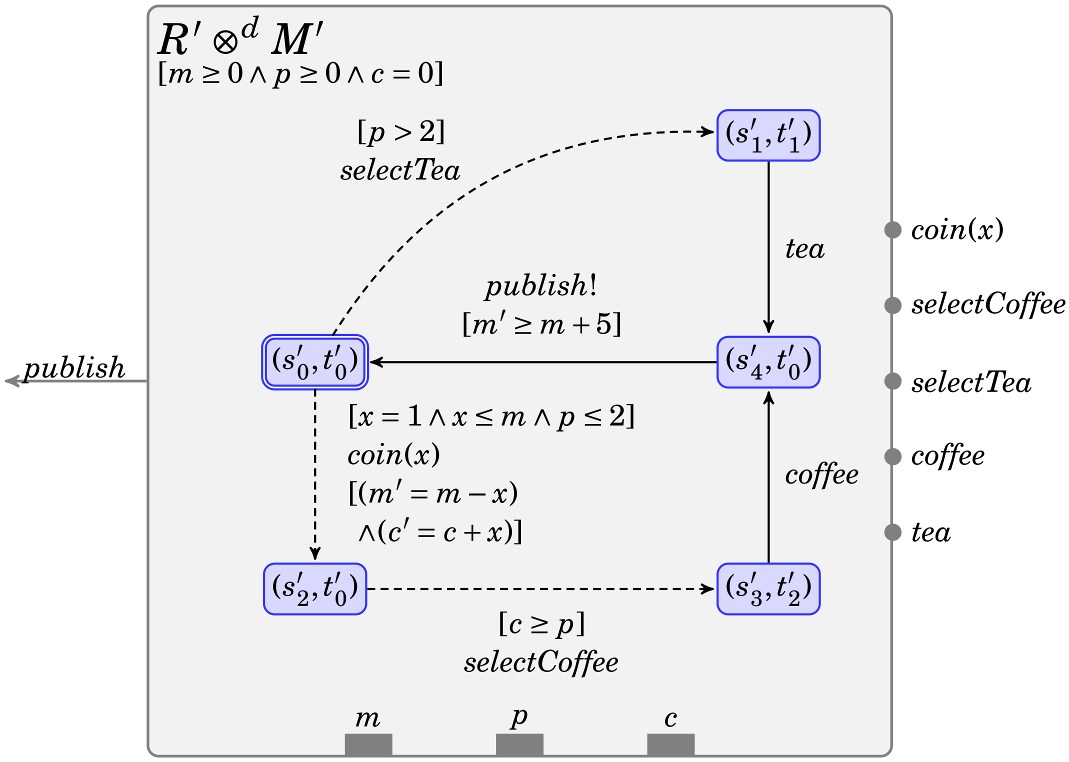


Figure 4.8: Modal synchronous composition of MIODs R' and M'

Definition 4.3.10 (Strong environment correctness)
Let $S,E \in \mathbb{MIOD}$ be composable. E is a strongly correct environment for S, *written $S \rightarrow^d_s E$, if for all reachable states*

$$((s,e),(\delta_S \cdot \delta_E)) \textit{ in } S \otimes^d E,$$

for all $\alpha \in \Sigma^{out}_S \cap \Sigma^{in}_E$, whenever

$$s \overset{[\varphi_S]\alpha![\pi_S]}{\dashrightarrow}_S$$

and $(\delta_S \cdot \delta_E \cdot v;\rho) \models \varphi_S$ for some $v \in \mathscr{D}(V^{req}_{S\otimes^d E})$ and some $\rho \in \mathscr{D}(par(\alpha))$, then

$$e \xrightarrow{[\varphi_E]\alpha?[\pi_E]}_E$$

such that $(\delta_S \cdot \delta_E \cdot v;\rho) \models \varphi_E$.

Recall that we write $S \leftrightarrows^d_s E$ ("S and E are strongly compatible") if $S \rightarrow^d_s E$ and $E \rightarrow^d_s S$.

Example 4.3.11
Consider the MIODs shown in Figures 4.1, 4.2 and their composition shown in Figure 4.3. Strong compatibility of the specifications R and M means that, for instance, in the reachable state $(s_0,t_0,[m \mapsto 1,\ldots])$, in s_0 the output $coin(x)!$ is possible in R with parameter valuation $[x \mapsto 1]$ since we have

$$([m \mapsto 1,\ldots];[x \mapsto 1]) \models x = 1 \wedge x \leq m \ .$$

For R and M to be strongly compatible, we have to check that M is ready to take that output. And indeed, there is a must-transition for $coin(x)?$ which must be enabled whenever the parameter valuation is $[x \mapsto 1]$ (independent of the provided data state of M), hence the output of R can be taken by M for sure. The reader may easily verify the other outputs sent from R to M (and the other way round), coming to the conclusion that R and M are strongly compatible, i.e. $R \leftrightarrows^d_s M$.

Strong environment correctness of MIODs is preserved by refinement:

Theorem 4.3.12 (Preservation of strong environment correctness)
Let $S,S',E,E' \in \mathbb{MIOD}$. If $S \rightarrow^d_s E$ and $S' \leq^d_s S$ and $E' \leq^d_s E$, then $S' \rightarrow^d_s E'$.

Proof. Let $((s',e'),\delta_S \cdot \delta_E)$ be a reachable state in $S \otimes^d E$. From reachability of s' in S', e' in E' and strong modal refinements $S' \leq^d_s S$, $E' \leq^d_s E$ we can conclude that there exist related states $s \in St_S$, $e \in St_E$ such that (s,e) is reachable in $S \otimes^d E$.

Assume that

$$s' \overset{[\varphi_{S'}]\alpha![\pi_{S'}]}{\dashrightarrow}_{S'}$$

with $\alpha \in \Sigma_S^{out} \cap \Sigma_E^{in}$, and

$$(\delta_S \cdot \delta_E \cdot \nu; \rho) \vDash \varphi_{S'}$$

for some $\nu \in \mathscr{D}(V^{req}_{S \otimes^d E})$ and some $\rho \in \mathscr{D}(par(\alpha))$. Then

$$s \overset{[\varphi_S]\alpha![\pi_S]}{\dashrightarrow}_S$$

such that $(\delta_S \cdot \delta_E \cdot \nu; \rho) \vDash \varphi_S$. Since $((s,e), \delta_S \cdot \delta_E)$ is reachable in $S \otimes^d E$, we can conclude that

$$e \xrightarrow{[\varphi_E]\alpha?[\pi_E]}_E$$

with $(\delta_S \cdot \delta_E \cdot \nu; \rho) \vDash \varphi_E$. Again, by strong modal refinement $E' \leq_s^d E$, we can infer that

$$e' \xrightarrow{[\varphi_{E'}]\alpha?[\pi_{E'}]}_{E'}$$

with $(\delta_S \cdot \delta_E \cdot \nu; \rho) \vDash \varphi_{E'}$ which was to be shown. □

Example 4.3.13

This result allows us to infer strong environment correctness of the refined MIODs R' and M' in both directions, see Figures 4.6,4.7 for the individual MIODs and Figure 4.8 for their composition. As we have proven strong environment correctness of R and M in both directions as well as the refinements $R' \leq_s^d R$ and $M' \leq_s^d M$, Theorem 4.3.12 allows us to conclude that both $R' \rightarrow_s^d M'$ and $M' \rightarrow_s^d R'$ hold.

4.3.3 Definition of $Th^{\mathbb{MIOD}}_{strong}$ and $Th^{d\mathbb{MIOD}}_{strong}$

We can now summarize our results for MIODs by stating that MIODs as well as deterministic MIODs form a specification theory.

Corollary 4.3.14

$Th^{\mathbb{MIOD}}_{strong} \triangleq (\mathbb{MIOD}, \mathbb{MIOD}^i, \leq_s^d, \otimes^d, \rightarrow_s^d)$ is a specification theory.
$Th^{d\mathbb{MIOD}}_{strong} \triangleq (d\mathbb{MIOD}, d\mathbb{MIOD}^i, \leq_s^d, \otimes^d, \rightarrow_s^d)$ is a specification theory.

Proof. Compositionality of strong modal refinement follows from Theorem 4.3.8. Preservation of strong environment correctness is shown in Theorem 4.3.12. Finality of implementations is shown in Lemma 4.3.5. All theorems also hold for deterministic MIOs. □

We can obtain a reflective embedding of $Th^{d\mathbb{MIOD}}_{strong}$ in $Th^{\mathbb{MIOD}}_{strong}$ by taking the identity function. The proof is straightforward and omitted.

Corollary 4.3.15

The identity function $id : d\mathbb{MIOD} \rightarrow \mathbb{MIOD}$ is a reflective embedding of $Th^{d\mathbb{MIOD}}_{strong}$ in $Th^{\mathbb{MIOD}}_{strong}$.

Furthermore, comparing the specification theories $Th^{\mathbb{MIO}}_{strong}$ and $Th^{\mathbb{MIOD}}_{strong}$, we can observe that the latter is clearly more expressive than the former. In fact we can define an embedding f of $Th^{\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIOD}}_{strong}$ in the following way. f transforms every MIO to a MIOD with no provided and required state variables, no action parameters, and pre- and postconditions being the universally valid predicate *true*. More formally, we can define $f : \mathbb{MIO} \to \mathbb{MIOD}$ by the function that maps any MIO $S = (St_S, s_0, \Sigma_S, \dashrightarrow_S, \longrightarrow_S)$ to the well-formed MIOD $f(S) = (St_S, s_0, \varphi_0, \Sigma_S, V, \dashrightarrow_{f(S)}, \longrightarrow_{f(S)})$ with $\varphi_0 = true$, $V = (V^{prov}, V^{req})$, $V^{prov} = V^{req} = \emptyset$, and

$$s \overset{[true]\alpha[true]}{\dashrightarrow}_{f(S)} s' \text{ if and only if } s \overset{\alpha}{\dashrightarrow}_S s' ,$$

$$s \xrightarrow{[true]\alpha[true]}_{f(S)} s' \text{ if and only if } s \xrightarrow{\alpha}_S s' .$$

Lemma 4.3.16
The function f as defined above is a reflective embedding of $Th^{\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIOD}}_{strong}$, and f restricted to deterministic MIOs is a reflective embedding of $Th^{d\mathbb{MIO}}_{strong}$ in $Th^{d\mathbb{MIOD}}_{strong}$.

Proof. The claim can be easily verified by looking at the definitions of composition, refinement and environment correctness: they all reduce to the corresponding definitions on MIOs as soon as all predicates are of the form *true*. For instance, it is easy to verify that $S \leq_s T$ if and only if $f(S) \leq^d_s f(T)$ for all $S, T \in \mathbb{MIO}$. □

Integrating the modal specification theories $Th^{\mathbb{MIOD}}_{strong}$ and $Th^{d\mathbb{MIOD}}_{strong}$ introduced in this section as well as the three reflective embeddings (denoted by $\hookrightarrow\!\!\!\triangleright$), we obtain the diagram shown in Figure 4.9 that extends the previous diagram in Figure 3.9.

4.4 Predicate Abstraction

We now switch our focus to the problem of deciding strong modal refinement for finite, well-formed MIODs. The problem $S \leq^d_s T$ for finite $S, T \in \mathbb{MIOD}$ but with *infinite* variable domains is known to be undecidable in general as soon as the assertion language can express two or more counters for (unbounded) integer variables. In this case the refinement problem can be reduced to the halting problem for Minsky machines [143] which is undecidable, see [111] for an overview of general techniques how to prove undecidability of bisimilarity.

In [22] we have proposed strong modal refinement for MIODs possibly with infinite variable domains. The naive technique in [22] relies on syntactically reachable states rather than considering a current data state δ being part of

$$\begin{tikzcd}
 & Th^{\mathbb{MIO}}_{weak} \\
Th^{\mathbb{MIOD}}_{strong} & Th^{\mathbb{MIO}}_{strong} \arrow[u, hook] \arrow[l, hook, -{Triangle[open]}] \\
Th^{d\mathbb{MIOD}}_{strong} \arrow[u, hook, -{Triangle[open]}] & Th^{d\mathbb{MIO}}_{strong} \arrow[u, hook, -{Triangle[open]}] \arrow[l, hook, -{Triangle[open]}]
\end{tikzcd}$$

Figure 4.9: Modal specification theories and embeddings

the refinement relation, similar but slightly more flexible than the conditions in Lemma 4.3.3 above. This refinement notion corresponds to the usual refinement of operation specifications, e.g. in behavioural subtyping for object-oriented programs [131], where preconditions can be weakened and postconditions can be strengthened. As this refinement notion only takes into account the control states, strong modal refinement can be decided whenever the set of control states is finite and the satisfaction relation can be decided. However, better approximations are clearly desirable, both from a theoretical and from a practical point of view.

Remark 4.4.1
We do not explicitly study the problem of verifying strong environment correctness for which, however, the predicate abstraction technique would also work. Whether predicate abstraction can be extended to the verification of weak strong modal refinement and weak environment correctness is out of scope and left for future work.

Example 4.4.2
We illustrate predicate abstraction with the following running example, slightly more technical and abstract than the previous examples but suited to illustrate our technique. For the rest of this section, we define the data universe $\mathcal{U}$ by the natural numbers, i.e. $\mathcal{U} \triangleq \mathbb{N}$. Figure 4.10 is an abstract specification T of a component with two provided state variables x and y of non-negative integers, and there is a single internal action. T expresses that performing a continuously alternates the data state between "the sum of $x + y$ is even" and "the sum of $x + y$ is odd".

Consider the MIOD S in Figure 4.11. Our goal is to show that S is a strong

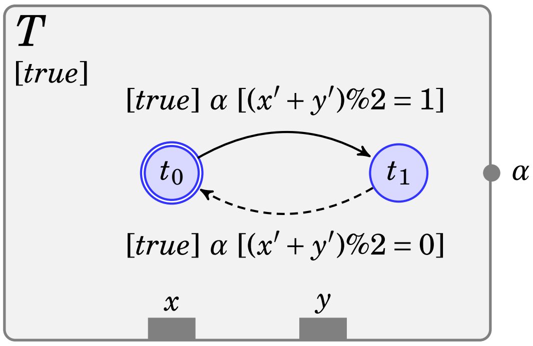

T
$[true]$
$[true]$ α $[(x'+y')\%2=1]$
t_0
t_1
α
$[true]$ α $[(x'+y')\%2=0]$
x
y

Figure 4.10: Abstract specification T

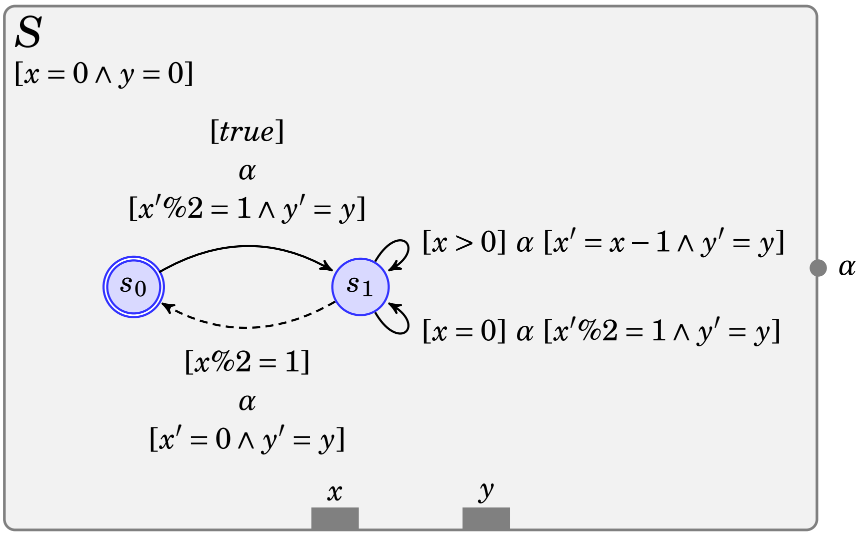


Figure 4.11: Concrete specification S

modal refinement of T. However, Lemma 4.3.3 does not work here – for instance, the state pair (s_0, t_0) fails the check for the must-transition because of the postconditions, i.e.

$$\not\models_{\forall} (x'\%2=1 \wedge y'=y) \Rightarrow (x'+y')\%2=1.$$

In this case, we propose to resort to predicate abstraction techniques [93]. Given two *finite*, well-formed MIODs S and T we derive over- and under-approximations S^o and T^u, respectively, using a finite number of predicates partitioning the provided and required data state space. Importantly, if predicates are decidable then we give algorithms how to derive the over- and under-approximation S^o and T^u, respectively. Since S^o and T^u are finite MIODs and the finitely many predicates can be encoded by Boolean variables, $S^o \leq_s^d T^u$ becomes decidable. As the main result of this approach to the verification of strong modal refinement, we show that $S^o \leq_s^d T^u$ implies $S \leq_s^d T$. We describe the predicate abstraction technique for MIODs without action parameters, however, they can be easily integrated with some more technical effort. Technical details about the encoding of predicates with Boolean variables can be found in [25].

Given a MIOD $S = (S, s_0, \varphi_0, \Sigma, V, \dashrightarrow, \longrightarrow)$, we partition the local data state space and the uncontrolled data state space using finitely many predicates

$$\Phi_1, \Phi_2, \dots, \Phi_N \in \mathit{Pred}(V^{prov}) \quad \text{and} \quad \Psi_1, \Psi_2, \dots, \Psi_M \in \mathit{Pred}(V^{req}).$$

It is crucial that the predicates form a partition, i.e. for $\Phi_1, \Phi_2, \dots, \Phi_N \in \mathit{Pred}(V^{prov})$ this is means

- $\not\models_\exists \Phi_i \wedge \Phi_j$ for all $1 \leq i < j \leq N$ and
- $\models_\forall \Phi_1 \vee \ldots \vee \Phi_N$;

similar for $\Psi_1, \Psi_2, \dots, \Psi_M \in \mathit{Pred}(V^{req})$. We fix these predicates in the following to simplify the presentation.

Example 4.4.3

For our running example, we fix the following predicates. Note that we do not have any required state variables.

$$\begin{aligned}
\Phi_1 &\triangleq y = 0 \wedge x = 0 \\
\Phi_2 &\triangleq y = 0 \wedge x > 0 \wedge x\%2 = 1 \\
\Phi_3 &\triangleq y = 0 \wedge x > 0 \wedge x\%2 = 0 \\
\Phi_4 &\triangleq y > 0 \wedge (x + y)\%2 = 1 \\
\Phi_5 &\triangleq y > 0 \wedge (x + y)\%2 = 0
\end{aligned}$$

These predicates are carefully chosen such that the occurring pre- and postconditions in S and T can be "best" approximated; automatic procedures obtaining such predicates is subject of future work.

The transition relation of the over-approximation expands the allowed behaviours and limits the required behaviours. Dually, the under-approximation will further restrict the allowed behaviour and add more required transitions. In other words, over-approximation is an *existential* abstraction on may-transitions and *universal* abstraction on must-transitions. Conversely, under-approximation is a *universal* abstraction on may-transitions and *existential* abstraction on must-transitions.

Let us define, for any $S \in \mathbb{MIOD}$, states $s, s' \in St_S$, and actions $\alpha \in \bigcup \Sigma$,

$$\begin{aligned}
may_S(s, \alpha, s') &\triangleq \{(\varphi, \pi) \mid s \overset{[\varphi]\alpha[\pi]}{\dashrightarrow}_S s'\}, \\
must_S(s, \alpha, s') &\triangleq \{(\varphi, \pi) \mid s \xrightarrow{[\varphi]\alpha[\pi]}_S s'\}.
\end{aligned}$$

Definition 4.4.4 (Over-approximation)
Let S and S^o be two MIODs with the same extended action signature and state signature. S^o is an over-approximation *of S w.r.t. $(\Phi_i)_{1\le i\le N}$, and $(\Psi_j)_{1\le j\le M}$, if $S \le_s^d S^o$ and for S^o it is satisfied that*

1. *the initial state predicate of S^o is of the form $\bigvee_{i\in I}\Phi_i$ for some $\emptyset \neq I \subseteq \{1,\dots,N\}$,*
2. *preconditions in S^o are of the form $(\bigvee_{i\in I}\Phi_i)\wedge(\bigvee_{j\in J}\Psi_j)$ for some $\emptyset \neq I \subseteq \{1,\dots,N\}$ and $\emptyset \neq J \subseteq \{1,\dots,M\}$,*
3. *postconditions in S^o are of the form $\bigvee_{i\in I}(\Phi_i)'$ for some $\emptyset \neq I \subseteq \{1,\dots,N\}$.*

We now describe a simple algorithm how an over-approximation can be obtained. Minimality requirements are used to ensure that the over-approximation is as close as possible to the original specification.

Algorithm 1 (A-O). *Let $S = (St_S, s_0, \varphi_0, \Sigma, V, \dashrightarrow, \longrightarrow) \in \mathbb{MIOD}$. The MIOD S^o is constructed as follows.*

- *The control state space of the over-approximation S^o is given by*

  $$St_{S^o} \triangleq St_S \times \left\{ \bigvee_{i\in I}\Phi_i \;\middle|\; \emptyset \neq I \subseteq \{1,\dots,N\} \right\}.$$

  The state predicate is added to over-approximate the data states which are actually reachable for this control state in S.

- *The initial state is $(s_0, \bigvee_{i\in I}\Phi_i)$ where $i \in I$ if and only if $\models_\exists \varphi_0 \wedge \Phi_i$.*

- *For a control state $(s,\Gamma) \in St_{S^o}$ and some $1 \le i \le N$, $1 \le j \le M$, the following may- and must-transitions are added:*

  - *If $may_S(s,\alpha,s') \neq \emptyset$ and*

    $$\models_\exists \Gamma \wedge \Phi_i \wedge \Psi_j \wedge \left(\bigvee_{(\varphi,\pi)\in may_S(s,\alpha,s')} \varphi \wedge \pi \right)$$

    then

    $$(s,\Gamma) \overset{[\Phi_i\wedge\Psi_j]\alpha[\bigvee_{i\in I}(\Phi_i)']}{\dashrightarrow}_{S^o} (s', \bigvee_{i\in I}\Phi_i)$$

    where $\emptyset \neq I \subseteq \{1,\dots,N\}$ is a minimal set satisfying

    $$\models_\forall \Gamma \wedge \Phi_i \wedge \Psi_j \wedge \left(\bigvee_{(\varphi,\pi)\in may_S(s,\alpha,s')} \varphi \wedge \pi \right) \Rightarrow \bigvee_{i\in I}(\Phi_i)'.$$

- *If $must_S(s,\alpha,s') \neq \emptyset$ and*

$$\models_\forall \Gamma \wedge \Phi_i \wedge \Psi_j \Rightarrow \left(\bigvee_{(\varphi,\pi)\in must_S(s,\alpha,s')} \varphi \right),$$

then

$$(s,\Gamma) \xrightarrow{[\Phi_i \wedge \Psi_j]\alpha[\bigvee_{i\in I}(\Phi_i)']}_{S^o} (s', \bigvee_{i\in I}(\Phi_i)')$$

where $\emptyset \neq I \subseteq \{1,\ldots,N\}$ is a minimal set satisfying

$$\models_\forall \Gamma \wedge \Phi_i \wedge \Psi_j \Rightarrow \left(\bigvee_{(\varphi,\pi)\in must_S(s,\alpha,s')} \varphi \wedge \left(\pi \Rightarrow \bigvee_{i\in I}(\Phi_i)'\right)\right).$$

The minimality of the postcondition ensures that the effect of this must-transition is best described (for the fixed set of predicates).

Theorem 4.4.5
Let S be a well-formed MIOD. If a well-formed MIOD S^o is obtained by Algorithm (A-O), then S^o is an over-approximation of S, in particular, it holds that $S \leq_s^d S^o$.

Proof. Observe that S^o adheres to the required format of initial state predicate, preconditions and postconditions. We only have to show that $S \leq_s^d S^o$. We define a relation $R \subseteq St_S \times St_{S^o} \times \mathscr{D}(V^{prov})$ by

$$R = \{(s,(s,\Gamma),\delta) \mid \delta \models \Gamma\}.$$

We show that R proves the refinement $S \leq_s^d S^o$. Assume that the initial state predicate of S^o is $\bigvee_{i\in I_0}\Phi_i$ for some $\emptyset \neq I_0 \subseteq \{1,\ldots,N\}$. We can infer that

$$(s_0,(s_0,\bigvee_{i\in I_0}\Phi_i),\delta_0) \in R$$

for all $\delta_0 \models \varphi_{0,S}$, since for every such δ_0 there must exist some $i \in I_0$ such that $\delta_0 \models \varphi_0 \wedge \Phi_i$.

Let us consider a triple $(s,(s,\Gamma),\delta) \in R$. Assume a must-transition

$$(s,\Gamma) \xrightarrow{[\Phi_i \wedge \Psi_j]\alpha[\bigvee_{k\in I}(\Phi_k)']}_{S^o} (s', \bigvee_{k\in I}\Phi_k)$$

such that $(\delta\cdot\nu) \models \Phi_i \wedge \Psi_j$ for some $\nu \in \mathscr{D}(V^{req})$. By the above rules for constructing S^o and by $\delta \models \Gamma$ (which we know by assumption $(s,(s,\Gamma),\delta) \in R$), it follows that there exists a must-transition

$$s \xrightarrow{[\varphi]\alpha[\pi]}_S s'$$

such that $(\delta\cdot\nu,\delta') \models \varphi \wedge \pi \Rightarrow \bigvee_{k\in I}(\Phi_k)'$ for all $\delta' \in \mathscr{D}(V^{prov})$. Hence we have that $(s',(s',\bigvee_{k\in I}(\Phi_k)'),\delta') \in R$ for all $\delta' \in \mathscr{D}(V^{prov})$ such that $(\delta\cdot\nu,\delta') \models \pi$.

The direction for the may-transition is similar and less difficult. □

Example 4.4.6
The over-approximation of S, see Figure 4.11, constructed according to Algorithm (A-O) using the fixed predicates $\Phi_1,\ldots,\Phi_5$, as chosen in Example 4.4.3, is shown in Figure 4.12. Strictly applying Algorithm (A-O) would result in two transitions from the state $(s_1, \Phi_1 \vee \Phi_3)$ to the state (s_1, Φ_2) which have been gathered to a single transition in Figure 4.12. Let us explain the construction by looking at two particular transitions. For instance, there is a may-transition

$$(s_0,\Phi_1) \xrightarrow{[\Phi_1]\alpha[(\Phi_2)']}_{S^o} (s_1,\Phi_2)$$

because there is a may-transition

$$s_0 \xrightarrow{[true]\alpha[x'\%2=1\wedge y'=y]}_{S} s_1$$

in S such that $\models_\exists \Phi_1 \wedge true \wedge x'\%2 = 1 \wedge y' = y$ and $I = \{2\}$ is the minimal set such that $\models_\forall \Phi_1 \wedge x'\%2 = 1 \wedge y' = y \Rightarrow (\Phi_2)'$. For an example of a must-transition in S^o, we consider

$$(s_0,\Phi_1) \xrightarrow{[\Phi_1]\alpha[(\Phi_2)']}_{S^o} (s_1,\Phi_2).$$

This transition is present in S^o because there is

$$s_0 \xrightarrow{[true]\alpha[x'\%2=1\wedge y'=y]}_{S} s_1$$

in S such that $\models_\forall \Phi_1 \Rightarrow true$ and $I = \{2\}$ is a minimal set satisfying $\models_\forall \Phi_1 \Rightarrow (x'\%2 = 1 \wedge y' = y \Rightarrow (\Phi_2)')$.

We now turn to the definition of an *under-approximation* T^u of T. Recall that an under-approximation is a *universal* abstraction on may-transitions and *existential* abstraction on must-transitions.

Definition 4.4.7 (Under-approximation)
Let T and T^u be two MIODs with the same extended action signature and state signature. T^u is an under-approximation *of T w.r.t. $(\Phi_i)_{1\leq i\leq N}$, and $(\Psi_j)_{1\leq j\leq M}$, if $T^u \leq_s^d T$ and for T^u it is satisfied that*

1. *the initial state predicate of T^u is of the form $\bigvee_{i\in I}\Phi_i$ for some $\emptyset \neq I \subseteq \{1,\ldots,N\}$,*
2. *preconditions in T^u are of the form $(\bigvee_{i\in I}\Phi_i) \wedge (\bigvee_{j\in J}\Psi_j)$ for some $\emptyset \neq I \subseteq \{1,\ldots,N\}$ and $\emptyset \neq J \subseteq \{1,\ldots,M\}$,*
3. *postconditions in T^u are of the form $\bigvee_{i\in I}(\Phi_i)'$ for some $\emptyset \neq I \subseteq \{1,\ldots,N\}$.*

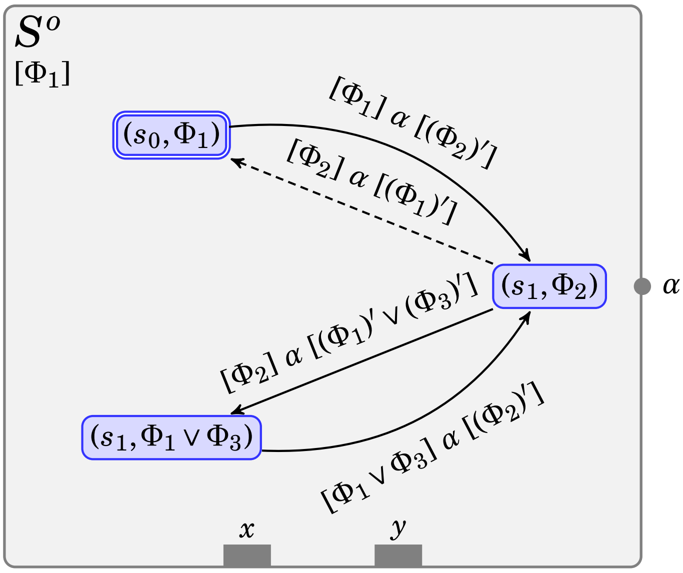

S^o
$[\Phi_1]$
(s_0,Φ_1)
$[\Phi_1]\ \alpha\ [(\Phi_2)']$
$[\Phi_2]\ \alpha\ [(\Phi_1)']$
(s_1,Φ_2)
α
$[\Phi_2]\ \alpha\ [(\Phi_1)' \vee (\Phi_3)']$
$(s_1,\Phi_1 \vee \Phi_3)$
$[\Phi_1 \vee \Phi_3]\ \alpha\ [(\Phi_2)']$
x
y

Figure 4.12: Over-approximation of S

We now describe a simple construction of an under-approximation. For this construction to work, we have to require that every postcondition does not mention unprimed variables, i.e. postconditions must be independent from the previous data state, and all postconditions must be equivalent to a disjunction of a selection of the predicates $(\Phi_1)',\dots,(\Phi_N)'$. More precisely, for every postcondition we stipulate the existence of a set $\emptyset \neq I \subseteq \{1,\dots,N\}$ such that $\models_\forall \pi \Leftrightarrow \bigvee_{k\in I}(\Phi_i)'$.

Algorithm 2 (A-U). *Let $T = (St_T, t_0, \varphi_0, \Sigma, V, \dashrightarrow, \longrightarrow)$ be a MIOD. The MIOD T^u is constructed as follows.*

- *The control state space of the under-approximation T^u is given by*

$$St_{T^u} \triangleq St_T \times \left\{ \bigvee_{i\in I} \Phi_i \;\middle|\; \emptyset \neq I \subseteq \{1,\dots,N\} \right\}.$$

  The state predicate is added to under-approximate the data states which are actually reachable in T for this control state.

- *The initial state is $(t_0, \bigvee_{i\in I}\Phi_i)$ where $i \in I$ if and only if $\models_\forall \Phi_i \Rightarrow \varphi_0$.*

- *For a control state $(t,\Gamma) \in St_{T^u}$, the following may- and must-transitions are added:*

  - *If $may_T(t,\alpha,t') \neq \emptyset$ and there exists a maximal set $\emptyset \neq I \subseteq \{1,\dots,N\}$ satisfying*

$$\models_\forall \Gamma \wedge \Phi_i \wedge \Psi_j \wedge \bigvee_{k\in I}(\Phi_k)' \Rightarrow \left(\bigvee_{(\varphi,\pi)\in may_T(t,\alpha,t')} \varphi \wedge \pi \right)$$

then

$$(t,\Gamma) \overset{[\Phi_i \wedge \Psi_j]\alpha[\bigvee_{k\in I}(\Phi_k)']}{\dashrightarrow}_{T^u} (t', \bigvee_{k\in I} \Phi_k).$$

- *For every* $(\varphi,\pi) \in must_T(t,\alpha,t')$*, if* $\models_\exists \Gamma \wedge \Phi_i \wedge \Psi_j \wedge \varphi$*, then there is*

$$(t,\Gamma) \xrightarrow{[\Phi_i \wedge \Psi_j]\alpha[\bigvee_{k\in I}(\Phi_k)']}_{T^u} (t', \bigvee_{k\in I}(\Phi_k)')$$

where $\emptyset \neq I \subseteq \{1,\dots,N\}$ *is a set satisfying* $\models_\forall \pi \Leftrightarrow \bigvee_{k\in I}(\Phi_k)'$*.*

Technically, under-approximation may yield a MIOD which is not well-formed: states may be reachable in which must-transitions are enabled which are not reachable by may-transitions. However, this does not affect the following result, as we only use transitivity of refinement which continue to hold for these MIODs. Note that postconditions on must-transitions are still satisfiable since they are of the form $\bigvee_{k\in I}(\Phi_i)'$ for some set $\emptyset \neq I \subseteq \{1,\dots,N\}$.

Theorem 4.4.8
Let T *be a well-formed MIOD. If* T^u *is the MIOD obtained according to Algorithm (A-U), then* T^u *is an under-approximation of* T*, in particular, it holds that* $T^u \leq_s^d T$*.*

Proof. Observe that T^u adheres to the required format of initial state predicate, preconditions and postconditions. We only have to show that $T^u \leq_s^d T$. We define a relation $R \subseteq St_{T^u} \times St_T \times \mathscr{D}(V^{prov})$ by

$$R \triangleq \{((t,\Gamma),t,\delta) \mid \delta \models \Gamma\}.$$

We show that R proves the refinement $T^u \leq_s^d T$. Assume that the initial state predicate of T^u is $\bigvee_{i\in I_0}\Phi_i$ for some $I_0 \subseteq \{1,\dots,N\}$. We can infer that

$$((t_0, \bigvee_{i\in I_0} \Phi_i), t_0, \delta_0) \in R$$

for all $\delta_0 \models \bigvee_{i\in I_0}\Phi_i$, since for every such δ_0 we have that $\delta_0 \models \bigvee_{i\in I_0}\Phi_i \wedge \varphi_{0,T}$.

Let us consider a triple $((t,\Gamma),t,\delta) \in R$. Assume a must-transition

$$t \xrightarrow{[\varphi]\alpha[\pi]}_T t'$$

such that $(\delta \cdot v) \models \varphi$ for some $v \in \mathscr{D}(V^{req})$. By assumption $(s,(s,\Gamma),\delta) \in R$, we know that $(\delta \cdot v) \models \Gamma \wedge \Phi_i \wedge \Psi_j \wedge \varphi$ for some $i \in \{1,\dots,N\}$ and some $j \in \{1,\dots,M\}$. Then by the rules for constructing T^u, there is

$$(t,\Gamma) \xrightarrow{[\Phi_i \wedge \Psi_j]\alpha[\bigvee_{k\in I}(\Phi_k)']}_{T^u} (t', \bigvee_{k\in I}(\Phi_k)')$$

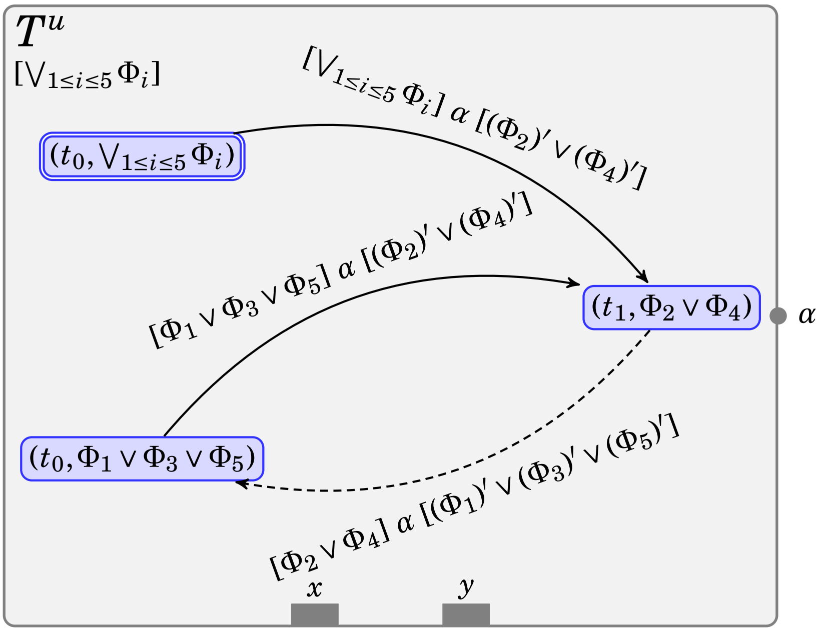

T^u
$[\bigvee_{1\leq i\leq 5}\Phi_i]$
$(t_0,\bigvee_{1\leq i\leq 5}\Phi_i)$
$[\bigvee_{1\leq i\leq 5}\Phi_i]\ \alpha\ [(\Phi_2)'\vee(\Phi_4)']$
$[\Phi_1\vee\Phi_3\vee\Phi_5]\ \alpha\ [(\Phi_2)'\vee(\Phi_4)']$
$(t_1,\Phi_2\vee\Phi_4)$
α
$(t_0,\Phi_1\vee\Phi_3\vee\Phi_5)$
$[\Phi_2\vee\Phi_4]\ \alpha\ [(\Phi_1)'\vee(\Phi_3)'\vee(\Phi_5)']$
x
y

Figure 4.13: Under-approximation T^u of T

where $I \subseteq \{1,\dots,N\}$ is a non-empty set satisfying $\models_\forall \pi \Leftrightarrow \bigvee_{k\in I}(\Phi_k)'$. It follows that for all $\delta' \in \mathcal{D}(V^{prov})$, whenever $(\delta\cdot v,\delta') \models \bigvee_{k\in I}\Phi_k$, then $(\delta\cdot v,\delta')\models\pi$ and $((t',\bigvee_{k\in I}\Phi_k),t',\delta')\in R$.

The direction for the may-transition is similar and less difficult. □

Example 4.4.9

Figure 4.13 shows the under-approximation T^u of T, obtained by Algorithm (A-U) except that, for presentation issues and similarly to before, transitions with the same source and target state and the same modality and postcondition have been gathered by drawing a single transition with disjoint preconditions.

Corollary 4.4.10

Let S and T be well-formed MIODs with the same extended action signature and the same state signature. Let $(\Phi_i)_{1\leq i\leq N}$ and $(\Psi_j)_{1\leq j\leq M}$ be predicates that partition the data state spaces $\mathcal{D}(V^{prov})$ and $\mathcal{D}(V^{req})$, respectively.

- *Let S^o be an over-approximation w.r.t. $(\Phi_i)_{1\leq i\leq N}$ and $(\Psi_j)_{1\leq j\leq M}$ obtained according to Algorithm (A-O).*

- *Let T^u be an under-approximation w.r.t. $(\Phi_i)_{1\leq i\leq N}$ and $(\Psi_j)_{1\leq j\leq M}$ obtained according to Algorithm (A-U).*

Then $S^o \leq_s^d T^u$ implies $S \leq_s^d T$.

Proof. This simply follows from transitivity of $\leq_s^d$ and

$$S \leq_s^d S^o \leq_s^d T^u \leq_s^d T$$

where $S \leq_s^d S^o$ is proven in Theorem 4.4.5 and $T^u \leq_s^d T$ is proven in Theorem 4.4.8. □

Example 4.4.11
It can be easily verified that S^o, shown in Figure 4.12, is a strong modal refinement of T^u, shown in Figure 4.13. Using Lemma 4.3.3 it sufficies to define the relation

$$\begin{aligned} Q = \Big\{ & \big((s_0, \Phi_1), (t_0, \bigvee_{1 \leq i \leq 5} \Phi_i)\big), \\ & \big((s_1, \Phi_2), (t_1, \Phi_2 \vee \Phi_4)\big), \\ & \big((s_1, \Phi_1 \vee \Phi_3), (t_0, \Phi_1 \vee \Phi_3 \vee \Phi_5)\big), \\ & \big((s_0, \Phi_1), (t_0, \Phi_1 \vee \Phi_3 \vee \Phi_5)\big)\Big\} \end{aligned}$$

and all state pairs in Q satisfy the conditions of Lemma 4.3.3. Thus we have shown that $S^o \leq_s^d T^u$.

Finally, our abstraction also supports compositional reasoning about composition in the following sense:

Theorem 4.4.12
Let S and T be two composable MIODs. Let $\Phi_i \in \mathscr{S}(V_S^{prov}), 1 \leq i \leq N$, $\Psi_j \in \mathscr{S}(V_T^{prov}), 1 \leq j \leq M$, and $\Omega_k \in \mathscr{S}(V_{S \otimes^d T}^{req})$, $1 \leq k \leq P$ be predicates partitioning the respective set of data states. Assume that

- *S^o and S^u are over- and under-approximation of S w.r.t. $(\Phi_i)_{1 \leq i \leq N}$ and $(\Psi_j \wedge \Omega_k)_{1 \leq j \leq M, 1 \leq k \leq P}$, obtained according to Algorithms (A-O) and (A-U), respectively,*
- *T^o and T^u are over- and under-approximation of T w.r.t. $(\Psi_j)_{1 \leq j \leq M}$ and $(\Phi_i \wedge \Omega_k)_{1 \leq i \leq N, 1 \leq k \leq P}$, obtained according to Algorithms (A-O) and (A-U), respectively,*
- *$(S \otimes^d T)^o$ and $(S \otimes^d T)^u$ are over- and under-approximation of $S \otimes^d T$ w.r.t. $(\Phi_i \wedge \Psi_j)_{1 \leq i \leq N, 1 \leq j \leq M}$ and $(\Omega_k)_{1 \leq k \leq P}$, obtained according to Algorithms (A-O) and (A-U), respectively.*

Then it is satisfied:

1. *$(S \otimes^d T)^o \leq_s^d S^o \otimes^d T^o$,*
2. *$S^u \otimes^d T^u \leq_s^d (S \otimes^d T)^u$.*

Proof. We prove that $(S \otimes^d T)^o \leq^d_s S^o \otimes^d T^o$. For this to show, we define a relation

$$R = \{(((s,t),\Gamma),((s,\Gamma_S),(t,\Gamma_T)),\delta) \mid \delta \models \Gamma \text{ and } \models_\forall \Gamma \Rightarrow \Gamma_S \wedge \Gamma_T\}$$

and show that R is a refinement relation for the above refinement statement.

First, let us prove that for all $\delta_0 \in \mathscr{D}(V^{prov}_{S\otimes^d T})$ such that $\delta_0 \models \varphi_{(S\otimes^d T)^o}$ we have that

$$(((s,t),\varphi_{(S\otimes^d T)^o}),((s,\varphi_{S^o}),(t,\varphi_{T^o})),\delta_0) \in R.$$

We only have to show that $\models_\forall \varphi_{(S\otimes^d T)^o} \Rightarrow \varphi_{S^o} \wedge \varphi_{T^o}$. Assume $\delta_S \cdot \delta_T \models \varphi_{(S\otimes^d T)^o}$. Then we must have Φ_i and Ψ_j such that $\delta_S \cdot \delta_T \models \Phi_i \wedge \Psi_j$. By construction of $\varphi_{(S\otimes^d T)^o}$ it must hold that

$$\models_\exists \varphi_{0,S} \wedge \varphi_{0,T} \wedge \Phi_i \wedge \Psi_j$$

which implies

$$\models_\exists \varphi_{0,S} \wedge \Phi_i \text{ and } \models_\exists \varphi_{0,T} \wedge \Psi_j.$$

Thus

$$\delta_S \models \varphi_{S^o} \text{ and } \delta_T \models \varphi_{T^o}$$

which is equivalent to

$$\delta_S \cdot \delta_T \models \varphi_{S^o} \wedge \varphi_{T^o}.$$

Second, consider a triple $(((s,t),\Gamma),((s,\Gamma_S),(t,\Gamma_T)),\delta_S \cdot \delta_T) \in R$.

- Assume that $((s,t),\Gamma) \xrightarrow{[\varphi]\alpha[\pi]} ((s',t'),\Gamma')$ in $(S \otimes^d T)^o$, and $(\delta_S \cdot \delta_T \cdot \nu, \delta'_S \cdot \delta'_T) \models \varphi \wedge \pi$ for some $\delta'_S \in \mathscr{D}(V^{prov}_S)$, $\delta'_T \in \mathscr{D}(V^{prov}_T)$ and $\nu \in \mathscr{D}(V^{req}_{S\otimes^d T})$. The interesting case is when $\alpha \in \Sigma^{ext}_S \cap \Sigma^{ext}_T$. By construction of $(S \otimes^d T)^o$, we know that

  $$\varphi = (\Phi_i \wedge \Psi_j) \wedge \Omega_k \text{ and } \pi = \bigvee_{(i,j)\in K} ((\Phi_i)' \wedge (\Psi_j)')$$

  for some $\emptyset \neq K \subseteq \{1,\ldots,N\} \times \{1,\ldots,M\}$ and

  $$\models_\forall \Gamma \wedge (\Phi_i \wedge \Psi_j) \wedge \Omega_k \wedge \left(\bigvee_{(\varphi,\pi)\in may_{S\otimes^d T}((s,t),\alpha,(s',t'))} \varphi \wedge \pi \right) \Rightarrow \bigvee_{(i,j)\in K} ((\Phi_i)' \wedge (\Psi_j)').$$

  Then we get

  $$\models_\exists \Gamma_S \wedge \Phi_i \wedge (\Psi_j \wedge \Omega_k) \wedge \left(\bigvee_{(\varphi,\pi)\in may_S(s,\alpha,s')} \varphi \wedge \pi \right)$$

  and

  $$\models_\exists \Gamma_T \wedge \Psi_j \wedge (\Phi_i \wedge \Omega_k) \wedge \left(\bigvee_{(\varphi,\pi)\in may_T(t,\alpha,t')} \varphi \wedge \pi \right).$$

Thus, there exist

$$(s,\Gamma_S) \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow} (s', \bigvee_{i\in I} \Phi_i) \text{ in } S^o$$

and

$$(t,\Gamma_T) \overset{[\varphi_T]\alpha[\pi_T]}{\dashrightarrow} (t', \bigvee_{j\in J} \Psi_j) \text{ in } T^o$$

such that I and J are the minimal (and non-empty) sets such that $\bigvee_{i\in I}\Phi_i$ and $\bigvee_{j\in J}\Psi_j$ over-approximate the next possible data states. Finally, we get that

$$(((s',t'), \bigvee_{(i,j)\in K} (\Phi_i \wedge \Psi_j)),((s,\bigvee_{i\in I}\Phi_i),(t,\bigvee_{j\in J}\Psi_j)),\delta'_S \cdot \delta'_T) \in R.$$

- Assume that

$$((s,\bigvee_{i\in I}\Phi_i),(t,\bigvee_{j\in J}\Psi_j)) \xrightarrow{[\Phi_i\wedge\Psi_j\wedge\Omega_k]\alpha[\bigvee_{i\in I'}(\Phi_i)'\wedge\bigvee_{j\in J'}(\Psi_j)']} ((s',\bigvee_{i\in I'}\Phi_i),(t',\bigvee_{j\in J'}\Psi_j))$$

in $S^o \otimes^d T^o$ and $\delta_S \cdot \delta_T \cdot \nu \vDash \Phi_i \wedge \Psi_j \wedge \Omega_k$ for some $\nu \in \mathscr{D}(V^{req}_{S\otimes^d T})$. The interesting case is again $\alpha \in \Sigma^{ext}_S \cap \Sigma^{ext}_T$. Then there exist transitions

$$(s,\bigvee_{i\in I}\Phi_i) \xrightarrow{[\Phi_i\wedge(\Psi_j\wedge\Omega_k)]\alpha[\bigvee_{i\in I'}(\Phi_i)']} (s',\bigvee_{i\in I'}\Phi_i)$$

and

$$(t,\bigvee_{j\in J}\Psi_j) \xrightarrow{[\Psi_j\wedge(\Phi_i\wedge\Omega_k)]\alpha[\bigvee_{j\in J'}(\Psi_j)']} (t',\bigvee_{j\in J'}\Psi_j).$$

Then we can infer that there exists

$$((s,t),\bigvee_{(i,j)\in K}(\Phi_i\wedge\Psi_j)) \xrightarrow{[\Phi_i\wedge\Psi_j\wedge\Omega_k]\alpha[\bigvee_{(i,j)\in K'}((\Phi_i)'\wedge(\Psi_j)')]} ((s',t'),\bigvee_{(i,j)\in K'}(\Phi_i\wedge\Psi_j))$$

in $S \otimes^d T^o$. We still have to show that

$$\vDash_\forall \left(\bigvee_{(i,j)\in K'}(\Phi_i\wedge\Psi_j)\right) \Rightarrow \left(\bigvee_{i\in I'}\Phi_i \wedge \bigvee_{j\in J'}\Psi_j\right).$$

But this follows from the rules of contructing over-approximations and the rules of composition.

The other claim $S^u \otimes^d T^u \leq^d_s (S \otimes^d T)^u$ can be shown in a similar way. □

This result allows reusing abstractions of individual components in a continued development and verification process. For instance, if we want to verify $S \otimes^d T \leq^d_s U$ then we can compute (or reuse) the less complex abstractions S^o and T^o. Theorem 4.4.12 implies then that from $S^o \otimes^d T^o \leq^d_s U^u$ we can infer $S \otimes^d T \leq^d_s (S \otimes^d T)^o \leq^d_s S^o \otimes^d T^o \leq^d_s U^u \leq^d_s U$ and hence $S \otimes^d T \leq^d_s U$.

4.5 Denotational Semantics

In this section we propose a formal denotational semantics of implementations of MIODs which assigns, to any implementation $I \in \mathbb{MIOD}^i$, its denotational semantics dsem(I) given as *implementation model* of I. For the definition of implementation models we use input/output automata with data states (IODs) which provide a suitable semantic formalization for the behaviour of components implemented on the basis of concrete data states and control states determining the current execution points of an implementation model. The state space of an implementation model is given by the Cartesian product of the set St_I of control states and the set $\mathscr{D}(V_I^{prov})$ of provided data states of I, i.e. any state $(i,\delta) \in St \times \mathscr{D}(V^{prov})$ of an implementation model is determined by a control state $i \in St_I$ and a provided data state $\delta \in \mathscr{D}(V_I^{prov})$. The set of implementation labels for the given extended action signature Σ_I and state signature V_I of I is defined by the set $\mathscr{L}^{impl}(\Sigma_I, V_I)$ which consists of all expressions of the form

$$[v]\alpha(\rho)$$

where $v \in \mathscr{D}(V_I^{req})$ represents a visible data state of the environment which is a guard for that transition, $\alpha \in \bigcup \Sigma_I$ is the action, and $\rho \in \mathscr{D}(par(\alpha))$ is a valuation of the formal parameters of α. The guard v expresses that the implementation model will only execute the transition if the environment is in the data state determined by v. This will, of course, be crucial when we consider the composition of implementation models later on. In a concrete program the guard may require that the component performs in one atomic step a test on the visible data state of the environment and, depending on the result, performs the action (or, if $\alpha \in \Sigma_I^{ext}$, waits for a synchronization with a communication partner).

Definition 4.5.1 (Input/output automaton with data states (IOD))
An input/output automata with data states (IOD)

$$(St, i_0, \delta_0, \Sigma, V, \longrightarrow)$$

consists of a set of control states St, an initial control state i_0, an initial data state $\delta_0 \in \mathscr{D}(V^{prov})$, an extended action signature Σ, a state signature $V = (V^{prov}, V^{req})$ such that $V^{prov} \cup V^{req} = \mathrm{SV}$, and a transition relation

$$\longrightarrow \subseteq (St \times \mathscr{D}(V^{prov})) \times \mathscr{L}^{impl}(\Sigma, V) \times (St \times \mathscr{D}(V^{prov})).$$

We use IODs as representatives of their isomorphism classes w.r.t. bijections on states. The set of those isomorphism classes of all IODs is denoted by $\mathbb{IOD}$.

Let us now discuss denotational semantics of implementations: To any implementation $I \in \mathbb{MIOD}^i$we assign its denotational semantics dsem$(I) \in \mathbb{IOD}$. The initial provided data state of the IOD dsem(I) is the unique data state that satisfies

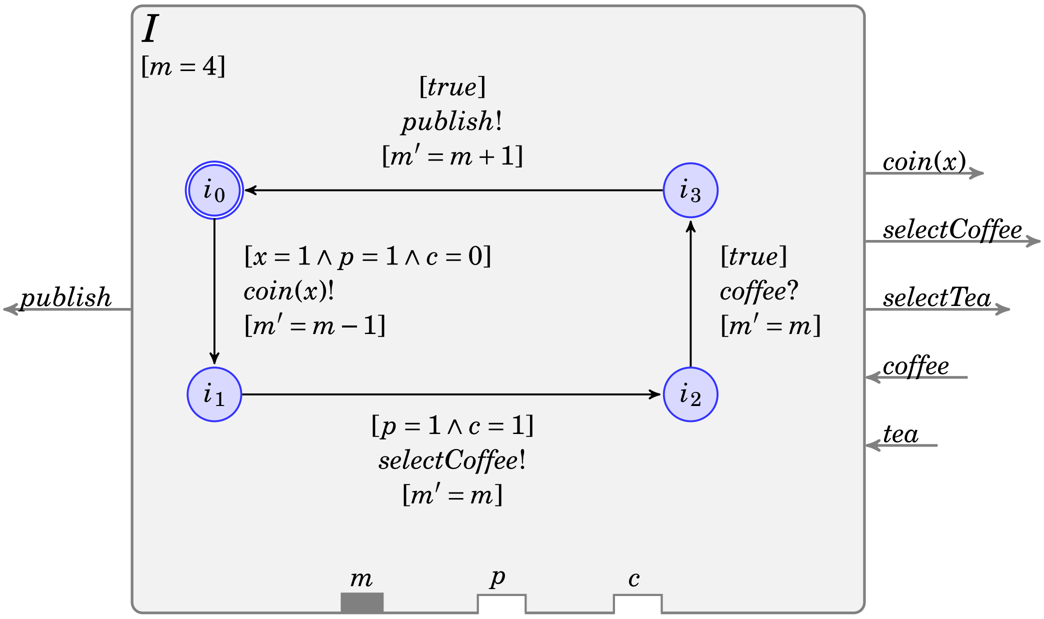


Figure 4.14: An implementation $I \in \mathbb{MIOD}^i$

the initial state predicate of I. The transition relation of $\text{dsem}(I)$ is determined by all transitions such that the involved data states satisfy the pre- and postcondition of a transition in I. The denotational semantics of implementations is formalized in the following definition.[4]

Definition 4.5.2 (Denotational semantics of implementations)
Let $I = (St_I, i_0, \varphi_{0,I}, \Sigma_I, V_I, \dashrightarrow_I, \longrightarrow_I) \in \mathbb{MIOD}^i$ be an implementation with extended action signature $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ and state signature $V = (V^{prov}, V^{req})$. The denotational semantics of I is defined by

$$\text{dsem}(I) \triangleq (St_I, i_0, \delta_0, \Sigma_I, V_I, \longrightarrow_{\text{dsem}(I)})$$

where $\delta_0 \in \mathcal{D}(V_I^{prov})$ is the unique data state satisfying $\delta_0 \models \varphi_{0,I}$, and the transition relation $\longrightarrow_{\text{dsem}(I)}$ is defined by the following rule:
for all $i, i' \in St_I$, $\delta, \delta' \in \mathcal{D}(V_I^{prov})$, $v \in \mathcal{D}(V_I^{req})$, $\alpha \in \Sigma_I$, $\rho \in \mathcal{D}(par(\alpha))$:

$$\frac{i \xrightarrow{[\varphi]\alpha[\pi]}_I i' \quad (\delta \cdot v, \delta'; \rho) \models \varphi \wedge \pi}{(i, \delta) \xrightarrow{[v]\alpha(\rho)}_{\text{dsem}(I)} (i', \delta')}$$

[4] Definition 4.5.2 coincides with the implementation relation defined in [22] if only input actions are considered.

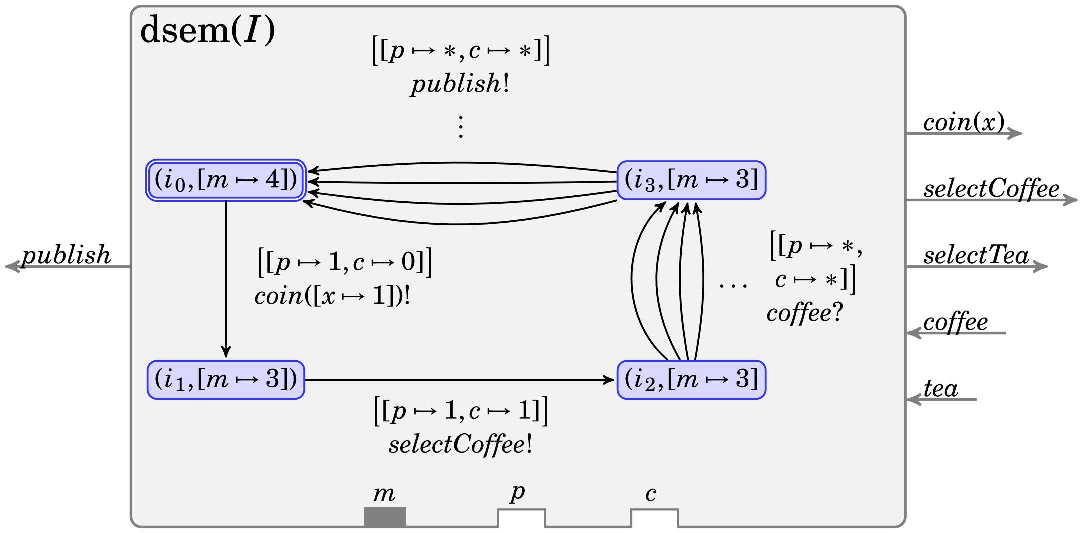


Figure 4.15: The denotational semantics dsem(I) of I, given as an IOD

Example 4.5.3
Figure 4.14 shows a implementation I which is a strong modal refinement of R (see Figure 4.1 in Section 4.1). The refinement can be easily verified by using Lemma 4.3.3 and the relation

$$\{(i_0, s_0), (i_1, s_0), (i_2, s_2), (i_3, s_3)\}.$$

Figure 4.15 shows the denotational semantics dsem(I) *of I. The transition labelled with $[p \mapsto *, c \mapsto *]$ coffee? stands for the set of transitions with labels from the set*

$$\{[p \mapsto a, c \mapsto b]\ \mathit{coffee?} \mid a, b \in \mathbb{Z}\},$$

and similar for the other transition using $*$ in its label.

Let us consider, for instance, the state $(i_2, [m \mapsto 3])$ in dsem(I)*, and the transition in I*

$$i_2 \xrightarrow{[\mathit{true}]\mathit{coffee?}[m'=m]}_I i_3.$$

This must-transition is implemented in dsem(I) *by the transitions labelled with $[p \mapsto a, c \mapsto b]$ coffee? for arbitrary $a, b \in \mathbb{Z}$, so for any required data state $\nu \in \mathcal{D}(\{p, c\})$ we have a transition labelled with $[\nu]$ coffee?. The postcondition $m' = m$ requires that the next provided data state is $[m \mapsto 3]$.*

Next we define a semantic composition operator for implementation models. Given two IODs G_1 and G_2, we say that they are *composable* if their action signatures and their state signatures are composable, respectively. If G_1 and G_2 are composable, then their composition $G_1 \otimes^d_{\mathrm{dsem}} G_2$ synchronizes transitions whose labels refer to shared actions: a transition with label $[\nu_1]\alpha(\rho)!$ of G_1 with current data state δ_1 is synchronized with a transition with label $[v_2]\alpha(\rho)?$ of G_2 with current data state δ_2 if[5]

- v_1 coincides with δ_2 on all provided state variables of G_2, i.e. $v_1(x) = \delta_2(x)$ for all $x \in V_2^{prov}$,
- v_2 coincides with δ_1 on all provided state variables of G_1, i.e. $v_2(x) = \delta_1(x)$ for all $x \in V_1^{prov}$, and
- v_1 and v_2 concide on all state variables that are required in both G_1 and G_2, i.e. $v_1(x) = v_2(x)$ for all $x \in (V_1^{req} \setminus V_2^{prov})$.

If the above conditions are met, then both transitions labelled with $[v_1]\alpha(\rho)!$ and $[v_2]\alpha(\rho)?$ are synchronized to a transition labelled by $[v]\alpha(\rho)$ where ρ is the unique required state variable such that $v(x) = v_1(x) = v_2(x)$ for all $x \in (V_1^{req} \setminus V_2^{prov})$.

If the above conditions for synchronization are not met then no synchronization happens and both transitions from G_1 and G_2 are dropped in the composition. As usual, all other transitions with non-shared actions are interleaved in the composition.

Definition 4.5.4 (Synchronous composition)
Let $G_j = (St_j, g_{0,j}, \delta_{0,j}, \Sigma_j, V_j, \rightarrow_j) \in \mathbb{IOD}$, $j \in \{1,2\}$ be two composable IODs. The synchronous composition *of G_1 and G_2 is defined by the IOD*

$$G_1 \otimes^d_{\mathrm{dsem}} G_2 \triangleq \left(St_1 \times St_2, (g_{0,1}, g_{0,2}), (\delta_{0,1} \cdot \delta_{0,2}), \Sigma_1 \otimes \Sigma_2, V_1 \otimes V_2, \rightarrow\right)$$

where the transition relation $\rightarrow$ is defined by the following rules:

$$\frac{(g_1,\delta_1) \xrightarrow{[v\cdot\delta_2]\alpha(\rho)}_1 (g_1',\delta_1') \quad \alpha \notin \Sigma_2^{ext}}{((g_1,g_2),\delta_1\cdot\delta_2) \xrightarrow{[v]\alpha(\rho)} ((g_1',g_2),\delta_1'\cdot\delta_2)} \qquad \frac{(g_2,\delta_2) \xrightarrow{[v\cdot\delta_1]\alpha(\rho)}_2 (g_2',\delta_2') \quad \alpha \notin \Sigma_1^{ext}}{((g_1,g_2),\delta_1\cdot\delta_2) \xrightarrow{[v]\alpha(\rho)} ((g_1,g_2'),\delta_1\cdot\delta_2')}$$

$$\frac{(g_1,\delta_1) \xrightarrow{[v\cdot\delta_2]\alpha(\rho)}_1 (g_1',\delta_1') \quad (g_2,\delta_2) \xrightarrow{[v\cdot\delta_1]\alpha(\rho)}_2 (g_2',\delta_2')}{((g_1,g_2),\delta_1\cdot\delta_2) \xrightarrow{[v]\alpha(\rho)} ((g_1',g_2'),\delta_1'\cdot\delta_2')}$$

The next result shows that our semantical composition operator $\otimes^d_{\mathrm{dsem}}$ preserves the syntactical composition operator $\otimes^d$ on the level of MIODs.

[5] Note that by definition of IODs and composability of G_1 and G_2, we have $V_i^{prov} \cup V_i^{req} = \mathrm{SV}$ for each $i \in \{1,2\}$, hence v_1 is defined for all state variables in V_2^{prov}, v_2 is defined for all state variables in V_1^{prov}, and $V_1^{req} \setminus V_2^{prov} = V_2^{req} \setminus V_1^{prov}$.

Theorem 4.5.5
Let $I_1, I_2 \in \mathbb{MIOD}^i$ be two composable implementations. Then dsem(I_1) *and* dsem(I_2) *are composable and*

$$\mathrm{dsem}(I_1) \otimes^d_{\mathrm{dsem}} \mathrm{dsem}(I_2) = \mathrm{dsem}(I_1 \otimes^d I_2).$$

Proof. Let $I_1 = (St_1, i_{0,1}, \delta_{0,1}, \Sigma_1, V_1, \dashrightarrow_1, \longrightarrow_1)$, $I_2 = (St_2, i_{0,2}, \delta_{0,2}, \Sigma_2, V_2, \dashrightarrow_2, \longrightarrow_2)$, and let $V = (V^{prov}, V^{req}) = V_1 \otimes V_2$.

Assume

$$((i_1, i_2), \delta_1 \cdot \delta_2) \xrightarrow{[v]\alpha(\rho)}_{\mathrm{dsem}(I_1) \otimes^d_{\mathrm{dsem}} \mathrm{dsem}(I_2)} ((i_1', i_2'), \delta_1' \cdot \delta_2').$$

Moreover, assume that $\alpha \in \Sigma_1^{ext} \cap \Sigma_2^{ext}$; the other cases are shown similarly. The transition must come from a synchronization of

$$(i_1, \delta_1) \xrightarrow{[v \cdot \delta_2]\alpha(\rho)}_{\mathrm{dsem}(I_1)} (i_1', \delta_1') \text{ and } (i_2, \delta_2) \xrightarrow{[v \cdot \delta_1]\alpha(\rho)}_{\mathrm{dsem}(I_2)} (i_2', \delta_2').$$

By the definition of the denotational semantics, we can infer that there exist transitions

$$i_1 \xrightarrow{[\varphi_1]\alpha[\pi_1]}_{I_1} i_1' \text{ and } i_2 \xrightarrow{[\varphi_2]\alpha[\pi_2]}_{I_2} i_2'$$

such that $(\delta_1 \cdot \delta_2 \cdot v, \delta_1' \cdot \delta_2'; \rho) \models \varphi_1 \wedge \varphi_2 \wedge \pi_1 \wedge \pi_2$. Hence there exists a transition

$$(i_1, i_2) \xrightarrow{[\varphi_1 \wedge \varphi_2]\alpha[\pi_1 \wedge \pi_2]}_{I_1 \otimes^d I_2} (i_1', i_2').$$

By the above satisfaction statements we can conclude that there is

$$((i_1, i_2), \delta_1 \cdot \delta_2) \xrightarrow{[v]\alpha(\rho)}_{\mathrm{dsem}(I_1 \otimes^d I_2)} ((i_1', i_2'), \delta_1' \cdot \delta_2')$$

which was to be shown. In fact all steps can be reversed in the proof, hence the desired equality of the transition relation is proven. □

We conclude this part on denotational semantics of implementations of MIODs by proposing a notion of strong environment correctness on the level of IODs that correspond to strong environment correctness on the level of MIODs.

Definition 4.5.6 (Strong environment correctness)
Let $G, H \in \mathbb{IOD}$ be two composable IODs. H is a strongly correct environment for G *if for every reachable state $((g,h), \delta_G \cdot \delta_H)$ in $G \otimes^d_{\mathrm{dsem}} H$, for all $\alpha \in \Sigma_G^{out} \cap \Sigma_H^{in}$,*

$$\textit{whenever } (g, \delta_G) \xrightarrow{[v \cdot \delta_H]\alpha(\rho)!}_G, \textit{ then } (h, \delta_H) \xrightarrow{[v \cdot \delta_G]\alpha(\rho)?}_H .$$

We show that strong environment correctness is preserved and reflected by the denotational semantics.

Theorem 4.5.7 (Preservation of strong environment correctness)
Let $I, J \in \mathbb{MIOD}$ be two MIODs. Then $I \rightarrow^d_s J$ if and only if $\mathrm{dsem}(J)$ *is a strongly correct environment for* $\mathrm{dsem}(I)$.

Proof. We only show the direction from left to right, the other direction can be shown by similar arguments. Let $((i,j),\delta_I \cdot \delta_J)$ be a reachable state in

$$\mathrm{dsem}(I) \otimes^d_{\mathrm{dsem}} \mathrm{dsem}(J).$$

Assume that there is

$$(i,\delta_I) \xrightarrow{[v\cdot\delta_J]\alpha(\rho)!}_{\mathrm{dsem}(I)},$$

then there exists

$$i \overset{[\varphi_I]\alpha![\pi_I]}{\dashrightarrow}_I$$

such that $(\delta_I \cdot \delta_J \cdot v;\rho) \models \varphi_I$. Then, from $I \rightarrow^d_s J$ it follows that there is

$$j \xrightarrow{[\varphi_J]\alpha?[\pi_J]}_J$$

such that $(\delta_I \cdot \delta_J \cdot v;\rho) \models \varphi_J$. This transition causes $\mathrm{dsem}(J)$ to have the transition

$$(j,\delta_J) \xrightarrow{[v\cdot\delta_I]\alpha(\rho)?}_{\mathrm{dsem}(J)}$$

which was to be shown. □

4.6 The Specification Theory $Th^{\mathbb{MIOD}}_{weak}$

So far, we have only looked at modal refinement and environment correctness for MIODs in their strong variant. In this section, we show how they can be extended to weak notions considering transitions labelled with internal actions as unobservable steps.

4.6.1 Weak Modal Refinement

Let us begin with proposing weak modal refinement for MIODs. In the case of MIOs (see Section 3.4.1), for a must-transition in the abstract specification, we have to simulate this must-transition in the concrete specification, however, we are allowed to add before and after the relevant transition some internal must-transitions. Obviously, those internal transitions are allowed to change the control states. When we take into account data states, we have to require that the internal must-transitions, that are introduced before and after a relevant transition, are all enabled under the same required data state and do not change the (visible) provided data state. This seems like an over-restriction, however, it is needed for compositionality reasons, see Example 4.6.4 below. For the may-transitions the situation is simpler: internal may-transitions surrounding the relevant transition just need to allow to stay in the same visible data state (under the same required data state).

To formalize that a must-transition shall not change the provided data state we introduce the notion of stability: a must-transition

$$s \xrightarrow{[\varphi]\alpha[\pi]} s'$$

of a MIOD S is (δ,ν)-*stable* for $\delta \in \mathscr{D}(V^{prov})$ and $\nu \in \mathscr{D}(V^{req})$ if there exists $\rho \in \mathscr{D}(par(\alpha))$ such that $(\delta\cdot\nu;\rho) \models \varphi$, and for all $\delta' \in \mathscr{D}(V^{prov})$, $(\delta\cdot\nu,\delta';\rho) \models \pi$ implies $\delta' = \delta$. A may-transition

$$s \overset{[\varphi]\alpha[\pi]}{\dashrightarrow} s'$$

is (δ,ν)-*stable* if there exists $\rho \in \mathscr{D}(par(\alpha))$ such that $(\delta\cdot\nu,\delta;\rho) \models \varphi \wedge \pi$.

Definition 4.6.1 (Weak modal refinement)
Let $S,T \in \mathbb{MIOD}$ with the same sets of input and output actions and the same state signature. S is a weak modal refinement of T, *denoted* $S \leq_w^d T$, *if* $\models \varphi_{0,S} \Rightarrow \varphi_{0,T}$ *and if there exists a refinement relation* $R \subseteq St_S \times St_T \times \mathscr{D}(V^{prov})$ *such that for all* $\delta_0 \in \mathscr{D}(\varphi_{0,S})$, $(s_0,t_0,\delta_0) \in R$, *and for all* $(s,t,\delta) \in R$:

1. ***(from concrete to abstract)***

   If

   $$s \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow} s'$$

   and $(\delta\cdot\nu,\delta';\rho) \models \varphi_S \wedge \pi_S$ *for some* $\nu \in \mathscr{D}(V^{req})$, $\rho \in \mathscr{D}(par(\alpha))$ *and* $\delta' \in \mathscr{D}(V^{prov})$, *then there exists a (possibly empty) sequence of* (δ,ν)*-stable internal may-transitions from* t *to some state* $t' \in St_T$*, and then*

   (a) *either* $\alpha \in \Sigma^{int}$, $\delta = \delta'$ *and* $(s',t',\delta) \in R$

   (b) *or there exists*

   $$t' \overset{[\varphi_T]\alpha[\pi_T]}{\dashrightarrow} t''$$

   such that

   i. $(\delta\cdot\nu,\delta';\rho) \models \varphi_T \wedge \pi_T$,

   ii. *there exist* (δ',ν)*-stable internal may-transitions from* t'' *to some state* $t''' \in St_T$ *such that* $(s',t''',\delta') \in R$.

2. ***(from abstract to concrete)***

   If

   $$t \xrightarrow{[\varphi_T]\alpha[\pi_T]} t'$$

such that $(\delta \cdot v; \rho) \models \varphi_T$ *for some* $v \in \mathscr{D}(V^{req})$ *and* $\rho \in \mathscr{D}(par(\alpha))$*, then there exists a (possibly empty) sequence of* (δ, v)*-stable internal must-transitions from* s *to some state* $s' \in St_S$*. Then*

(a) either $\alpha \in \Sigma^{int}$*,* $(\delta \cdot v, \delta; \rho) \models \pi_T$ *and* $(s', t', \delta) \in R$

(b) or there exists

$$s' \xrightarrow{[\varphi_S]\alpha[\pi_S]} s''$$

such that

i. $(\delta \cdot v; \rho) \models \varphi_S$*,*

ii. for all $\delta' \in \mathscr{D}(V^{prov})$*, if* $(\delta \cdot v, \delta'; \rho) \models \pi_S$*, then* $(\delta \cdot v, \delta'; \rho) \models \pi_T$*, and there exist* (δ', v)*-stable internal must-transitions from* s'' *to some state* $s''' \in St_S$ *such that* $(s''', t', \delta') \in R$*.*

Lemma 4.6.2
Weak modal refinement $\leq^d_w$ *is reflexive and transitive.*

Proof. The proof is along the lines of the proof of Lemma 4.3.2. □

Let us prove compositionality of weak modal refinement w.r.t. $\otimes^d$.

Theorem 4.6.3 (Compositional refinement)
Let $S, S', T, T' \in \mathbb{MIOD}$ *such that* S *and* T *are composable. If* $S' \leq^d_w S$ *and* $T' \leq^d_w T$ *then* S' *and* T' *are composable and* $S' \otimes^d T' \leq^d_w S \otimes^d T$*.*

Proof. It suffices to prove the case $T' = T$ since composition is commutative. Definedness of $S' \otimes^d T$ follows from $S \otimes^d T$ and $S' \leq^d_w S$ since weak modal refinement does not change external actions or state signatures. Let $V = (V^{prov}, V^{req}) = V_S \otimes V_T$. To show $S' \otimes^d T \leq^d_w S \otimes^d T$ we define a refinement relation $R \subseteq (St_{S'} \times St_T) \times (St_S \times St_T) \times \mathscr{D}(V^{prov})$ by

$$R = \{(((s', t), (s, t), \delta_S \cdot \delta_T) \mid (s', s, \delta_S) \in R_S, t \in St_T, \delta_T \in \mathscr{D}(V^{prov}_T)\}$$

where R_S witness $S' \leq^d_s S$. We show that R proves $S' \otimes^d T \leq^d_s S \otimes^d T$.

$((s', t), (s, t), (\delta_{0,S'} \cdot \delta_{0,T}))$ clearly holds for any $\delta_{0,S'} \in \mathscr{D}(\varphi_{0,S'})$ and any $\delta_{0,T} \in \mathscr{D}(\varphi_{0,T})$, because $\models_\forall \varphi_{0,S'} \Rightarrow \varphi_{0,S}$. Let

$$((s', t), (s, t), (\delta_S \cdot \delta_T)) \in R,$$

so we can assume that $(s', s, \delta_S) \in R_S$. Let

$$(s, t) \xrightarrow{[\varphi]\alpha[\pi]} (\hat{s}, \hat{t}) \tag{1}$$

be a must-transition in $S \otimes^d T$. The only interesting case is when α is a shared action of S and T, i.e. $\alpha \in \Sigma_S^{ext} \cap \Sigma_T^{ext}$. Assume that there is $v \in \mathscr{D}(V^{req})$, $\rho \in \mathscr{D}(par(\alpha))$ such that

$$(\delta_S \cdot \delta_T \cdot v; \rho) \models \varphi.$$

From (1) and the rules of composition it follows that there exist

$$s \xrightarrow{[\varphi_S]\alpha[\pi_S]} \hat{s} \text{ and } t \xrightarrow{[\varphi_T]\alpha[\pi_T]} \hat{t}$$

such that $\varphi = \varphi_S \wedge \varphi_T$ and $\pi = \pi_S \wedge \pi_T$. From $(s', s, \delta_S) \in R_S$ we can conclude that there exists a (possibly empty) sequence of $(\delta_S, v \cdot \delta_T)$-stable internal must-transitions from s' to s'', and

$$s'' \xrightarrow{[\varphi_{S'}]\alpha[\pi_{S'}]} \hat{s}''$$

such that for all $\delta'_S \in \mathscr{D}(V_S^{prov})$, if $(\delta_S \cdot \delta_T \cdot v, \delta'_S; \rho) \models \pi_{S'}$ then $(\delta_S \cdot \delta_T \cdot v, \delta'_S; \rho) \models \pi_S$, and there exist $(\delta'_S, v \cdot \delta_T)$-stable internal must-transitions from $\hat{s}''$ to $\hat{s}'''$ such that $(\hat{s}''', \hat{s}, \delta'_S) \in R_S$. It follows that there is a (possibly empty) sequence of $(\delta_S \cdot \delta_T, v)$-stable internal transitions from (s', t) to (s'', t), and

$$(s'', t) \xrightarrow{[\varphi_{S'} \wedge \varphi_T]\alpha[\pi_{S'} \wedge \pi_T]} (\hat{s}'', \hat{t})$$

such that

- $(\delta_S \cdot \delta_T \cdot v; \rho) \models \varphi_{S'}$,
- for all $\delta'_S \in \mathscr{D}(V_S^{prov})$ and all $\delta'_T \in \mathscr{D}(V_T^{prov})$, if $(\delta_S \cdot \delta_T \cdot v, \delta'_S \cdot \delta'_T; \rho) \models \pi_{S'} \wedge \pi_T$ then $(\delta_S \cdot \delta_T \cdot v, \delta'_S \cdot \delta'_T; \rho) \models \pi_S \wedge \pi_T$, and there exist $(\delta'_S \cdot \delta'_T, v)$-stable internal must-transitions from $(\hat{s}'', \hat{t})$ to $(\hat{s}''', \hat{t})$ such that $((\hat{s}''', \hat{t}), (\hat{s}, \hat{t}), \delta'_S \cdot \delta'_T) \in R$.

The other direction of weak modal refinement (condition (1) of Definition 4.6.1, from concrete to abstract) is very similar to the proof above. □

Example 4.6.4

The condition that all internal must-transitions that can be added before and after a transition in S which simulates a corresponding must-transition in T is indeed necessary. Let us consider a simple example. Let E be a MIOD with x as required state variable consisting of the transition

$$e_0 \xrightarrow{[x=1]\alpha}_E e_1.$$

Let T be a MIOD with x as provided state variable, initial state predicate $x = 0$ *and with the transition*

$$t_0 \xrightarrow{[x=0]\beta}_T t_1.$$

Clearly, the composition $E \otimes^d T$ will never allow any α to happen. If we refine T to a MIOD S with transitions

$$s_0 \xrightarrow{[x=0]\tau[x'=1]}_S s_1 \xrightarrow{[x=1]\tau[x'=0]}_S s_2 \xrightarrow{[x=0]\beta}_S s_3$$

then the composition $E \otimes^d S$ would require α to happen which renders $E \otimes^d S$ not to be a refinement of $E \otimes^d T$.

4.6.2 Weak Environment Correctness

Finally, we propose weak environment correctness for MIODs. Similar to weak environment correctness for MIOs in Section 3.4.2, Chapter 3, we can allow internal or non-shared output must-transitions before the relevant input transition.

Definition 4.6.5 (Weak environment correctness)
Let $S, E \in \mathbb{MIOD}$ be composable. E is a weakly correct environment for *S if for all reachable states*

$$((s,e), \delta_S \cdot \delta_E) \text{ in } S \otimes^d E,$$

for all $\alpha \in \Sigma^{out}_S \cap \Sigma^{in}_E$, whenever

$$s \overset{[\varphi_S]\alpha![\pi_S]}{\dashrightarrow}_S$$

such that $(\delta_S \cdot \delta_E \cdot \nu; \rho) \models \varphi_S$ for some $\nu \in \mathscr{D}(V^{req}_{S \otimes^d E})$ and some $\rho \in \mathscr{D}(par(\alpha))$, then there exist $(\delta_E, \nu \cdot \delta_S)$-stable must-transitions with actions in Σ^{int}_E leading from e to a state e' in E such that

$$e \xrightarrow{[\varphi_E]\alpha?[\pi_E]}_E$$

and $(\delta_E \cdot \delta_S \cdot \nu; \rho) \models \varphi_E$.

Theorem 4.6.6 (Preservation of weak environment correctness)
If $S \rightarrow^d_w E$ and $S' \leq^d_w S$ and $E' \leq^d_w E$, then $S' \rightarrow^d_w E'$.

Proof. This proof is similar to the case of preservation of $\rightarrow^d_s$ by $\leq^d_s$, see Theorem 4.3.12. The only difference is that weak modal refinement can introduce some stable must-transitions before the relevant input transition which do not fail weak environment correctness to be satisfied, see Definition 4.6.5. □

4.6.3 Definition of $Th^{\mathbb{MIOD}}_{weak}$

We arrive at another specification theory for MIODs with weak notions of refinement and environment correctness.

Corollary 4.6.7
$Th^{\mathbb{MIOD}}_{weak} \triangleq (\mathbb{MIOD}, \mathbb{MIOD}^i, \leq^d_w, \otimes^d, \rightarrow^d_w)$ is a specification theory.

Proof. Compositionality of weak modal refinement is shown in Theorem 4.6.3, preservation of weak environment correctness is shown in Theorem 4.6.6. Finality of implementations can be proven as follows: for any $I \in \mathbb{MIOD}^i$ and $S \in \mathbb{MIOD}$ such that $S \leq_w^d I$ demonstrated by a refinement relation R, the refinement relation $R' \triangleq \{(i,s) \mid (s,i) \in R\}$ proves $I \leq_w^d S$. □

Again, the introduced specification theory $Th_{weak}^{\mathbb{MIOD}}$ can be integrated into our existing hierarchy, cf. Figure 4.9, as follows. Firstly, the identity function $id : \mathbb{MIOD} \to \mathbb{MIOD}$ is an embedding of $Th_{strong}^{\mathbb{MIOD}}$ in $Th_{weak}^{\mathbb{MIOD}}$; note that id is not a reflective embedding as weak modal refinement does not imply strong modal refinement, and similarly, weak environment correctness does not imply strong environment correctness. Secondly, the same function $f : \mathbb{MIO} \to \mathbb{MIOD}$ from Lemma 4.3.16, which maps every MIO to a MIOD with empty state signature and trivial pre- and postconditions, is a reflective embedding of $Th_{weak}^{\mathbb{MIO}}$ in $Th_{weak}^{\mathbb{MIOD}}$. The new modal specification theory $Th_{weak}^{\mathbb{MIOD}}$ and both embeddings are shown in Figure 4.16.

$$\begin{tikzcd}
Th_{weak}^{\mathbb{MIOD}} & Th_{weak}^{\mathbb{MIO}} \arrow[l, hook', -{Triangle[open]}] \\
Th_{strong}^{\mathbb{MIOD}} \arrow[u, hook] & Th_{strong}^{\mathbb{MIO}} \arrow[l, hook', -{Triangle[open]}] \arrow[u, hook] \\
Th_{strong}^{d\mathbb{MIOD}} \arrow[u, hook, -{Triangle[open]}] & Th_{strong}^{d\mathbb{MIO}} \arrow[l, hook', -{Triangle[open]}] \arrow[u, hook, -{Triangle[open]}]
\end{tikzcd}$$

Figure 4.16: Modal specification theories for MIOs and MIODs

4.7 Discussion and Related Work

Related works to the specification of component behaviours on the basis of transition systems has been already summarized in Section 3.5; most of them incorporate a notion of data in their model, but lack the flexibility of modalities. Here, we would like to summarize other related works with a focus on how the specification of interaction and of the data flow are integrated.

Several process algebras have been studied to integrate data. As a prominent example, we mention μCRL [95] that uses ACP and integrates data defined by equational abstract data types. μCRL allows process variables and actions with parameters. LOTOS [110] is a combination of an process algebraic part, based on CCS and CSP, and a data algebraic part, based on the algebraic data type language ACT ONE [54]. More recently, CSP-CASL [160] has been proposed, a combination of CSP and the algebraic specification language CASL. The integration of process algebra and data in the context of object-oriented systems is followed by, e.g., CSP-OZ [85, 166], where classes are specified by Object-Z, and the object-oriented system is specified by CSP. CSP-OZ is equipped with failures and divergence semantics from CSP, and the integration of Object-Z and CSP happens at the semantic level. Contrary to refinement, compatibility has not yet been explicitly studied in CSP-OZ.

Other approaches like Circus [161, 177] or rCos [132] offer means to specify interaction and data aspects, they do not support modalities expressing allowed and required behaviour. Other related approaches are based on symbolic transition systems (STS) [80, 12] but STS are mainly focusing on model checking and not on interface theories supporting the (top down) development of concurrent systems by refinement. In particular, STS do not support loose data specifications in the sense of postconditions being arbitrary relations rather than functions. STS are in fact similar to our implementations. Most closely related to the concept of MIODs is the study of Mouelhi et al. [146] who consider an extension of the theory of interface automata [62] to data states. However, their approach does not take into account modal refinements, and they rely on a global set of variables rather than provided and required variables. Sociable interfaces [58] are another extension of interface automata which take into account data states in a similar way, however, they do not consider modalities for transitions. Beside the lack of modalities in all of the above works, observational refinement taking into account loose data specifications has not been studied explicitly so far. Existing works on modal transition systems and their use as specification formalism for component interfaces [124, 159, 158] do not consider explicit data states.

Predicate abstraction is a well-studied abstraction technique for verification of large or even infinite state transition systems. Originally introduced by Graf and Saïdi in [93], it has been used in the context of generalized model checking [89] to derive modal transition systems as abstractions (more precisely, over-approximations) of usual labelled transition systems. In [163], Shoham and Grumberg use generalized Kripke modal transitions systems for monotonic abstraction-refinement for verifying CTL formulae. This work is in fact very close to our approach since they employ hyper-must-transitions (similar to must-transitions in disjunctive modal transition systems [127]), which also occur to some extent in our formalism due to the looseness of postconditions. Finally, we note that the bisimilarity problem for infinite systems has been intensively studied, and several results show decidability for various subclasses, see [119] for an overview.

Publication history. Prior versions of extending input/output automata with an alternating refinement relation (similar to refinement in interface automata) which takes data into account has been published in [19]. MIODs have been first presented in a more restricted formulation in [18] with a longer and revised version in [22]. This chapter is mostly based on the recent work [25] in which the notion of strong modal refinement and the predicate abstraction technique are introduced.

4.8 Summary

MIODs extend of modal input/output automata by taking into account data specifications. Provided and required state variables for components allow for modelling data-dependent communications increasing the modelling power considerably. We have studied strong and weak modal refinement and strong and weak environment correctness in the context of MIODs, and identified them as conservative extensions of the corresponding notions for MIOs, leading to reflective embeddings of the specification theories for MIOs in the ones for MIODs. We have also discussed how strong modal refinement for finite MIODs can be verified, and proposed a predicate abstraction technique for the verification of refinement involving infinite variable domains. Finally, we have studied denotational semantics of implementations of MIODs, by translating them into concrete input/output automata with data states, and we have shown that the denotational semantics is compositional and preserves environment correctness.

Chapter 5

$\mathscr{K}$-Weighted Modal Input/Output Automata

Nowadays, software systems are often subject of strict non-functional requirements, as they operate in restricted environments, like for instance in embedded systems. Therefore, a suitable formalism for interface specifications of components shall express not only functional but also non-functional properties. The latter are quantifiable properties addressing aspects like resource consumption (time, power, etc.), costs or probabilities. In order to adress both functional and non-functional properties of reactive systems, we need new integrated specification formalisms allowing to naturally express quantitative information which can be effectively analyzed. In the literature, several automata-based formalisms have been proposed, for instance, timed automata [5], weighted automata [74], hybrid automata [104] or probabilistic automata [155].

We introduce the novel formalism of $\mathscr{K}$-weighted modal input/output automata that extends modal input/output automata by transition weights from an algebraic structure $\mathscr{K} = (K, \preceq, \oplus)$. $\mathscr{K}$ features a set of weights K, a weight refinement relation $\preceq \subseteq K \times K$ and a synchronization operator $\oplus : K \times K \to K$. The algebraic structure $\mathscr{K}$ is general enough to represent quantitative aspects of the model with $\preceq$ describing how transition weights can be refined and with $\oplus$ describing the effect on transition weights during synchronization in composition. The novel specification formalism maintains the desirable properties required by any specification theory supporting compositional reasoning. Besides compositionality and an equivalence result of modal and thorough refinement, we provide a logical characterization of $\mathscr{K}$-weighted modal input/output automata with a Hennessy-Milner-Logic [100] as done for modal transition systems by Larsen in [121].

Outline. We start in Section 5.1 by defining $\mathscr{K}$-weighted modal input/output automata by adding weights from a weight structure $\mathscr{K}$ to transition labels.

We define modal synchronous composition for $\mathscr{K}$-weighted MIOs in Section 5.2 and introduce a specification theory for strong modal refinement in Section 5.3. Hennessy-Milner-Logic for $\mathscr{K}$-weighted MIOs is proposed in Section 5.4 and it is shown that it characterizes strong modal refinement. In Section 5.5 we introduce weak modal refinement for $\mathscr{K}$-weighted MIOs giving rise to another specification theory. In Section 5.6, the envisaged hierarchy of modal specification theory is completed by integrating $\mathscr{K}$-WMIODs. Finally, we summarize some related works in Section 5.7 and conclude this chapter in Section 5.8.

5.1 Definition

We shall now introduce the notion of $\mathscr{K}$-weighted modal input/output automata and some basic properties of the formalism. Before that we need to define the notion of weight structures used during the system design and specification refinement.

Definition 5.1.1 (Weight structure)
A weight structure $\mathscr{K} = (K, \preceq, \oplus)$ *consists of a nonempty (possibly infinite) set K of weights, a partial order $\preceq$ on K expressing weight refinement, and total function $\oplus : K \times K \to K$ describing the resulting weight of synchronizing two weighted transitions.*

A weight $k \in K$ is called an final weight *if $k' \preceq k$ implies $k \preceq k'$ and hence $k = k'$ for all $k' \in K$. In other words, final weights are all elements in K that cannot be refined further by other weights. The set of all final weights of $\mathscr{K}$ is denoted by $\mathscr{K}^i$.*

We require any weight structure $\mathscr{K} = (K, \preceq, \oplus)$ to satisfy the following properties:

1. *The synchronization operator $\oplus$ is commutative and associative.*
2. *The synchronization operator $\oplus$ is compositional: for all $k, k', \ell, \ell' \in K$, if $k' \preceq k$ and $\ell' \preceq \ell$, then $k' \oplus \ell' \preceq k \oplus \ell$.*
3. *The synchronization operator $\oplus$ is closed under final weights: for all $k, k' \in \mathscr{K}^i$, $k \oplus k' \in \mathscr{K}^i$.*
4. *For any weight $\ell \in K$ there exists a final weight $k \in \mathscr{K}^i$ such that $k \preceq \ell$.*

To each weight $k \in K$ we associate the set $[\![k]\!]$ of all final weights below k by

$$[\![k]\!] = \left\{ k' \in \mathscr{K}^i \;\middle|\; k' \preceq k \right\}.$$

Note that by condition (4.) in Definition 5.1.1 any weight $k \in K$ can be refined to a final weight, i.e. $[\![k]\!] \neq \emptyset$ for every $k \in K$.

We can now define $\mathscr{K}$-weighted modal input/output automata that combine MIOs with weight structures.

Definition 5.1.2 ($\mathcal{K}$-Weighted modal input/output automata)
A $\mathcal{K}$-weighted modal input/output automaton ($\mathcal{K}$-WMIO) *is a tuple*

$$(St, s_0, \Sigma, \dashrightarrow, \longrightarrow)$$

where St is a set of states with initial state $s_0 \in St$, Σ is an action signature, $\dashrightarrow \subseteq St \times (\bigcup \Sigma) \times K \times St$ is a may-transition relation, and $\longrightarrow \subseteq \dashrightarrow$ is a must-transition relation. A $\mathcal{K}$-WMIO $(St, s_0, \Sigma, \dashrightarrow, \longrightarrow)$ is an implementation *if $\dashrightarrow = \longrightarrow$ and all weights on the transitions are final weights.*

A transition $(s, \alpha, k, s') \in \dashrightarrow$ in a $\mathcal{K}$-WMIO S is written

$$s \overset{\alpha,k}{\dashrightarrow}_S s'.$$

Similar to MIOs in Chapter 3 we use $\mathcal{K}$-WMIOs as representatives of their isomorphism classes w.r.t. bijections on states. The set of those isomorphism classes is denoted by $\mathcal{K}$-$\mathbb{WMIO}$, the set of isomorphism classes of all implementations is denoted by $\mathcal{K}$-$\mathbb{WMIO}^i$.

As in the previous chapters, $\mathcal{K}$-WMIOs are represented as graphs with the convention that whenever two states are connected by both a must- and a may-transition under the same action and the same weight, then we draw only the must-transition.

Example 5.1.3
The most trivial instance of the weight structure is $\mathcal{K}_{triv} = (\{\bullet\}, \preceq_{triv}, \oplus_{triv})$ where $\preceq_{triv} = \{(\bullet, \bullet)\}$ and $\oplus(\bullet, \bullet) = \bullet$. In this minimal weight structure, the single weight $\bullet$ is also a final weight.

Example 5.1.4
Another more interesting weight structure is given by integer intervals with the natural inclusion ordering as partial order:

$$\mathcal{K}_{intv} = (K_{intv}, \preceq_{intv}, \oplus_{intv})$$

where

$$K_{intv} = \big\{[x, y] \,\big|\, x \in \mathbb{Z}, y \in \mathbb{Z}, x \leq y\big\}$$

and $[x', y'] \preceq_{intv} [x, y]$ if $x \leq x'$ and $y' \leq y$. From the definition of weight refinement $\preceq_{intv}$ it follows that final weights are singleton intervals:

$$\mathcal{K}^i_{intv} = \{\,[x, x] \mid x \in \mathbb{Z}\,\}.$$

The reader may consult Figure 5.1 for the illustration of the ordering $\preceq_{intv}$ of $\mathcal{K}_{intv}$ and Figure 5.2 for three examples of $\mathcal{K}_{intv}$-weighted modal input/output automata. Strong modal refinement $\leq_s^{\mathcal{K}_{intv}}$ is introduced in the next section. Note that the automaton I is an implementation while S and T are not. Lastly, there is

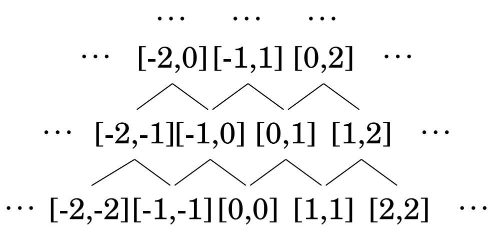


Figure 5.1: The partial order $\preceq_{intv}$ on integer intervals

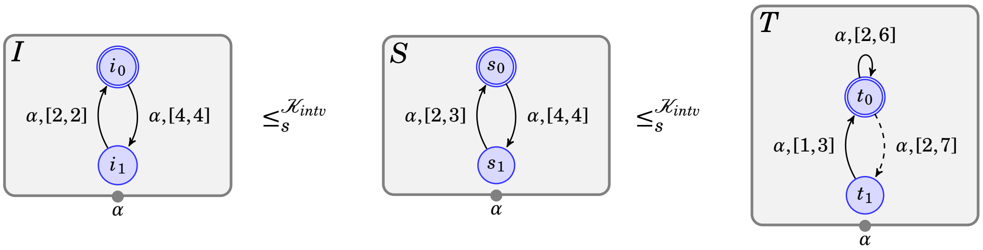


Figure 5.2: Strong modal refinement of $\mathscr{K}_{intv}$-WMIOs

the weight composition operator $\oplus_{intv}$ for integer intervals. The definition of $\oplus_{intv}$ depends on how we want to interpret the weights. If the weights on transitions model, e.g., costs or energy consumption then the composition operator may be defined as the sum of intervals:

$$[i_1, j_1] \oplus_{intv} [i_2, j_2] \triangleq [i_1 + i_2, j_1 + j_2]$$

Other options for the interpretation of weights may be that intervals model, for instance, (discrete) time intervals in which a transition can be executed. This interpretation leads to taking the interval intersection as the weight composition operator. We will stick to interpreting weights as energy consumption, thus $\oplus_{intv}$ is assumed to be defined as above from now on. It is straightforward to check that $\oplus_{intv}$ is commutative, associative, compositional and closed under final weights.

We conclude this exemplifying instantiation by considering a more realistic example. Figure 5.3 presents a simple electronic wiper control component for a car, with a normal mode and an optional fast mode. Integer intervals express the allowed energy consumption of each action (using abstract energy units).

Remark 5.1.5

Weight structures can be naturally combined by a product construction allowing to form (by a general construction) new weight structures combining several aspects from existing ones.

If $\mathscr{K}_1 = (K_1, \preceq_1, \oplus_1)$ and $\mathscr{K}_2 = (K_2, \preceq_2, \oplus_2)$ are two weight structures, then the product *$\mathscr{K}_1 \otimes \mathscr{K}_2$ is the weight structure $(K_1 \times K_2, \preceq, \oplus)$ with $\preceq$ and $\oplus$ defined as follows, for all $k_1, k_1' \in K_1$ and all $k_2, k_2' \in K_2$:*

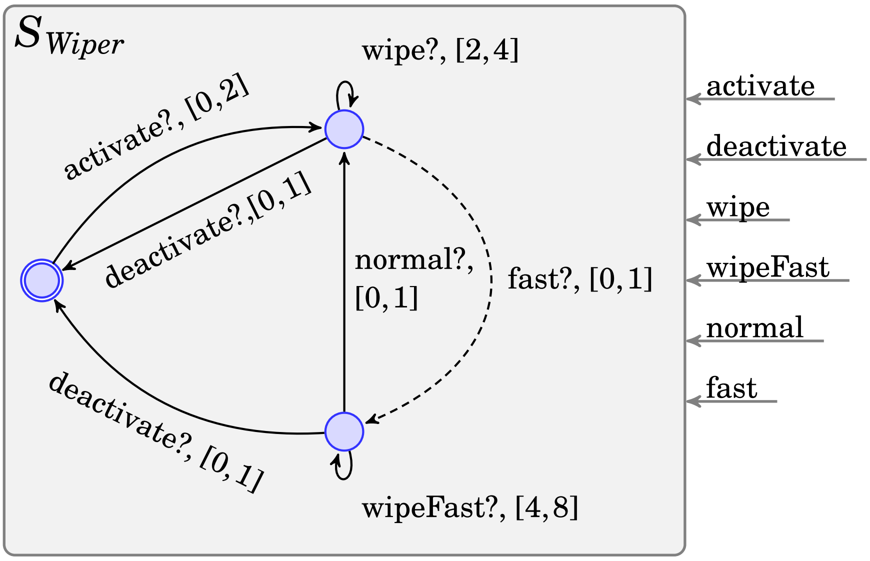


Figure 5.3: $\mathscr{K}_{intv}$-WMIO S_{Wiper} modeling a simple wiper control component of a car

- $(k'_1,k'_2) \preceq (k_1,k_2)$ *if and only if* $k'_1 \preceq_1 k_1$ *and* $k'_2 \preceq_2 k_2$,
- $(k_1,k_2) \oplus (k'_1,k'_2) \triangleq (k_1 \oplus_1 k'_1, k_2 \oplus_2 k'_2)$.

It is easy to see that the product construction is well-defined, i.e. that $\preceq$ is a partial order, $\oplus$ is commutative, associative, compositional and closed under final weights, and any weight can be refined by a final weight. Using the product construction of weight structures, we can e.g. combine existing weight structures to new (multi-)weight structures.

5.2 Modal Synchronous Composition

Two $\mathscr{K}$-WMIOs can be synchronously composed by synchronizing transitions with the same shared action. Regarding the weight of the resulting transition in the composition we use the weight synchronization operator $\oplus$. Two transitions for the action a, labelled with weight $k \in K$ and $\ell \in K$, result in a synchronized transition labelled by action a and weight $k \oplus \ell$.

Definition 5.2.1 (Modal synchronous composition)
Let $S_1, S_2 \in \mathscr{K}$-$\mathbb{WMIO}$ be composable. The modal synchronous composition *of S_1 and S_2 is defined as the $\mathscr{K}$-WMIO*

$$S_1 \otimes^{\mathscr{K}} S_2 \triangleq (St_1 \times St_2, (s_{0,1}, s_{0,2}), \Sigma_1 \otimes \Sigma_2, \dashrightarrow, \longrightarrow)$$

where the transition relations $\dashrightarrow$ and $\longrightarrow$ are defined by the following rules:

$$\frac{s_1 \overset{\alpha,k}{\dashrightarrow}_1 s_1' \quad \alpha \notin \Sigma_2^{ext}}{(s_1,s_2) \overset{\alpha,k}{\dashrightarrow} (s_1',s_2)} \qquad \frac{s_1 \xrightarrow{\alpha,k}_1 s_1' \quad \alpha \notin \Sigma_2^{ext}}{(s_1,s_2) \xrightarrow{\alpha,k} (s_1',s_2)}$$

$$\frac{s_2 \overset{\alpha,k}{\dashrightarrow}_2 s_2' \quad \alpha \notin \Sigma_1^{ext}}{(s_1,s_2) \overset{\alpha,k}{\dashrightarrow} (s_1,s_2')} \qquad \frac{s_2 \xrightarrow{\alpha,k}_2 s_2' \quad \alpha \notin \Sigma_1^{ext}}{(s_1,s_2) \xrightarrow{\alpha,k} (s_1,s_2')}$$

$$\frac{s_1 \overset{\alpha,k_1}{\dashrightarrow}_1 s_1' \quad s_2 \overset{\alpha,k_2}{\dashrightarrow}_2 s_2' \quad \alpha \in \Sigma_1^{ext} \cap \Sigma_2^{ext}}{(s_1,s_2) \overset{\alpha,k_1 \oplus k_2}{\dashrightarrow} (s_1',s_2')}$$

$$\frac{s_1 \xrightarrow{\alpha,k_1}_1 s_1' \quad s_2 \xrightarrow{\alpha,k_2}_2 s_2' \quad \alpha \in \Sigma_1^{ext} \cap \Sigma_2^{ext}}{(s_1,s_2) \xrightarrow{\alpha,k_1 \oplus k_2} (s_1',s_2')}$$

Clearly, modal synchronous composition is commutative and pseudo-associative.

5.3 The Specification Theories $Th_{strong}^{\mathscr{K}\text{-}\mathbb{WMIO}}$ and $Th_{strong}^{d\mathscr{K}\text{-}\mathbb{WMIO}}$

5.3.1 Strong Modal Refinement

We shall now define the notion of strong modal refinement that combines the already known different simulation directions for may- and must-transitions with the weight refinement given by the partial ordering on the weight structure. More precisely, a may-transition labelled with action α and weight k in the concrete $\mathscr{K}$-WMIO must be simulated by the abstract $\mathscr{K}$-WMIO with a may-transition labelled with action α and weight ℓ such that $k \preceq \ell$; a must-transition labelled with action α and weight ℓ in the abstract $\mathscr{K}$-WMIO must be simulated by the concrete $\mathscr{K}$-WMIO with a must-transition labelled with action α and weight k such that $k \preceq \ell$.

Definition 5.3.1 (Strong modal refinement)
Let $S,T \in \mathscr{K}$-$\mathbb{WMIO}$ with the same action signature Σ. S is a strong modal refinement of T, written $S \leq_s^{\mathscr{K}} T$, if there exists a refinement relation $R \subseteq S \times T$ with $(s_0,t_0) \in R$ such that for all $(s,t) \in R$, $\alpha \in \bigcup\Sigma$, $k \in K$:

1. *for all $s' \in St_S$, $k \in K$, if $s \overset{\alpha,k}{\dashrightarrow}_S s'$, then there exist $t' \in St_T$ and $\ell \in K$ such that $t \overset{\alpha,\ell}{\dashrightarrow}_T t'$, $k \preceq \ell$ and $(s',t') \in R$,*

2. *for all $t' \in St_T$, $\ell \in K$, if $t \xrightarrow{\alpha,\ell}_T t'$, then there exist $s' \in St_S$ and $k \in K$ such that $s \xrightarrow{\alpha,k}_S s'$, $k \preceq \ell$ and $(s',t') \in R$.*

The *implementation semantics* of a $\mathcal{K}$-WMIO S is defined by the class $[\![S]\!]^{\mathcal{K}}_s$ of all implementations refining S, i.e. $[\![S]\!]^{\mathcal{K}}_s = \{ I \in \mathcal{K}\text{-}\mathbb{WMIO}^i \mid I \le^{\mathcal{K}}_s S \}$.

Example 5.3.2
Strong modal refinement of $\mathcal{K}$-WMIOs is illustrated in Figure 5.2 for the weight structure $\mathcal{K}_{intv}$. The refinement relation $\{(s_0,t_0),(s_1,t_0)\}$ is witnessing the strong modal refinement between S and T. Note that the refined specification S is not an implementation yet as it contains the weight $[2,3]$ which is not a final weight. We can thus refine it further, ending up with an implementation I. Here, the witnessing relation is $\{(i_0,s_0),(i_1,s_1)\}$.

It can be easily proven that modal refinement $\le^{\mathcal{K}}_s$ is a preorder. The proof relies on the fact that weight refinement $\preceq$ is a partial order.

Lemma 5.3.3
Every implementation $I \in \mathcal{K}\text{-}\mathbb{WMIO}^i$ is a final element with respect to strong modal refinement.

Proof. Let $I \in \mathcal{K}\text{-}\mathbb{WMIO}^i$ and assume another $\mathcal{K}$-WMIO $S \in \mathcal{K}\text{-}\mathbb{WMIO}$ such that $S \le^{\mathcal{K}}_s I$, with refinement relation R. It is straightforward to prove that the refinement relation $R^{-1} = \{(i,s) \mid (s,i) \in R\}$. □

The notion of strong modal refinement can be understood as refinement defined at the syntactical level as it directly relates the states of two specifications. A semantically motivated notion of refinement is *thorough refinement*, as already explained in Chapter 2: S is a thorough refinement of T if every implementation of S is also an implementation of T. A consequence of Theorem 2.1.2 in Chapter 2 is that strong modal refinement implies thorough refinement. In general, thorough refinement does not imply strong modal refinement. A counterexample, using the weight structure K_{intv} is given in Figure 5.4. Clearly, the transition

$$s_0 \overset{\alpha,[0,1]}{\dashrightarrow}_S s_1$$

cannot be matched by any of the two transitions from t_0 as $[0,1] \not\preceq_{intv} [0,0]$ and $[0,1] \not\preceq_{intv} [1,1]$. Hence $S \not\le^{\mathcal{K}_{intv}}_s T$. On the other hand, any implementation of S is either empty or it is a tree of height one with the outgoing transitions labelled by α and either $[0,0]$ or $[1,1]$. All such implementations are also refinements of T.

It is known that for classical modal transition systems thorough refinement implies the modal one, under the assumption of determinism [32]; see also Theorem 3.3.7. We can generalize this result to the set of $\mathcal{K}$-weighted modal input/output automata. Before we define when a $\mathcal{K}$-WMIO is deterministic we

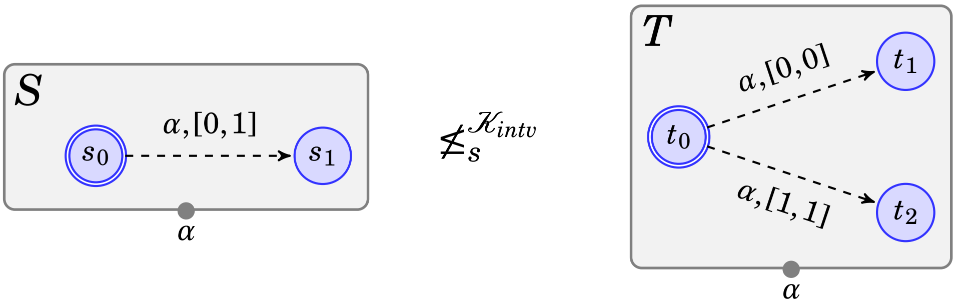


Figure 5.4: Completeness of strong modal refinement does not hold in general: $[\![S]\!]_s^{\mathcal{K}} \subseteq [\![T]\!]_s^{\mathcal{K}}$, but $S \not\leq_s^{\mathcal{K}_{intv}} T$

first define when two weights k_1, k_2 are unifiable, that is, if there is another label k which overlaps with k_1 and k_2 with respect to their sets of final weights.

Definition 5.3.4 (Unifiable weights)
Two weights $k_1, k_2 \in K$ are called unifiable *if there exists $k \in K$ such that $[\![k]\!] \cap [\![k_1]\!] \neq \emptyset$ and $[\![k]\!] \cap [\![k_2]\!] \neq \emptyset$.*

Then, determinism expresses that for any two outgoing may-transitions from the same state and labelled by the same action α, but with different weights k_1 and k_2, the weights k_1 and k_2 are not unifiable.

Definition 5.3.5 (Determinism)
Let $\mathcal{K} = (K, \preceq, \oplus)$ be a weight structure. A $\mathcal{K}$-WMIO S is called deterministic *if for any state $s \in St_S$, any action $\alpha \in \bigcup \Sigma_S$ and any two transitions*

$$s \overset{\alpha,k_1}{\dashrightarrow}_S s'_1 \quad \textit{and} \quad s \overset{\alpha,k_2}{\dashrightarrow}_S s'_2,$$

if k_1 and k_2 are unifiable, then $k_1 = k_2$ and $s'_1 = s'_2$.

The set of the isomorphism classes of deterministic $\mathcal{K}$-WMIOs is denoted by $d\mathcal{K}$-$\mathbb{WMIO}$, the set of isomorphism classes of all deterministic implementations is denoted by $d\mathcal{K}$-$\mathbb{WMIO}^i$.

Returning to Figure 5.4 we can realize that the system T is not deterministic as there is a branching of the transitions with weights $[0,0]$ and $[1,1]$, while there exists a weight $[0,1]$ such that $[\![[0,1]]\!] \cap [\![[0,0]]\!] \neq \emptyset$ and $[\![[0,1]]\!] \cap [\![[1,1]]\!] \neq \emptyset$.

A very natural assumption that has to be imposed on the weight structures later on in order to show completeness of strong modal refinement, is completeness of weight refinement: inclusion of sets of final weights implies weight refinement.

Definition 5.3.6 (Completeness of weight refinement)
Let $\mathcal{K} = (K, \preceq, \oplus)$ be a weight structure. Weight refinement $\preceq$ is complete *if for all $k, \ell \in K$, $[\![k]\!] \subseteq [\![\ell]\!]$ implies $k \preceq \ell$.*

Note that weight refinement is always sound by definition, i.e. $k \preceq \ell$ implies $[\![k]\!] \subseteq [\![\ell]\!]$ by transitivity of weight refinement.

Under the assumption of (1) completeness of weight refinement, and (2) determinism of the abstract $\mathcal{K}$-WMIO, thorough refinement implies strong modal refinement.

Theorem 5.3.7 (Relative completeness of strong modal refinement) *Let $\mathcal{K} = (K, \preceq, \oplus)$ be a weight structure for which weight refinement $\preceq$ is complete. Let $S, T \in \mathcal{K}$-$\mathbb{WMIO}$ such that T is deterministic. Then $[\![S]\!]^{\mathcal{K}}_{s} \subseteq [\![T]\!]^{\mathcal{K}}_{s}$ implies $S \leq^{\mathcal{K}}_{s} T$.*

Proof. In this proof, for a given $\mathcal{K}$-WMIO S, we write (s,S) for S where the initial state s_0 is replaced by $s \in S$.

Assume that $[\![S]\!]^{\mathcal{K}}_{s} \subseteq [\![T]\!]^{\mathcal{K}}_{s}$. We define a relation $R \subseteq St_S \times St_T$ as the smallest relation satisfying:

1. $(s_0, t_0) \in R$,
2. if $(s,t) \in R$, $s \overset{\alpha,k}{\dashrightarrow} s'$, $t \overset{\alpha,\ell}{\dashrightarrow} t'$, and $[\![k]\!] \cap [\![\ell]\!] \neq \emptyset$ then $(s', t') \in R$.

First, we show a technical result (that we use later on) saying that any $(s,t) \in R$ satisfies $[\![(s,S)]\!]^{\mathcal{K}}_{s} \subseteq [\![(t,T)]\!]^{\mathcal{K}}_{s}$. For $(s_0,t_0) \in R$, we have $[\![(s_0,S)]\!]^{\mathcal{K}}_{s} = [\![S]\!]^{\mathcal{K}}_{s} \subseteq [\![T]\!]^{\mathcal{K}}_{s} = [\![(t_0,T)]\!]^{\mathcal{K}}_{s}$ from the assumption $[\![S]\!]^{\mathcal{K}}_{s} \subseteq [\![T]\!]^{\mathcal{K}}_{s}$. Now, let $(s,t) \in R$ such that $[\![(s,S)]\!] \subseteq [\![(t,T)]\!]$ and assume that there are

$$s \overset{\alpha,k}{\dashrightarrow}_S s' \text{ and } t \overset{\alpha,\ell}{\dashrightarrow}_T t',$$

and $[\![k]\!] \cap [\![\ell]\!] \neq \emptyset$. Let $I' \in [\![(s',S)]\!]^{\mathcal{K}}_{s}$ and let $m \in [\![k]\!] \cap [\![\ell]\!]$ which exists by the construction. Then there exists an implementation $(i_0, I) \in [\![(s,S)]\!]^{\mathcal{K}}_{s}$ such that

$$i_0 \xrightarrow{\alpha,m} i'$$

and $(i', I) \leq^{\mathcal{K}}_{s} I'$. From $[\![(s,S)]\!]^{\mathcal{K}}_{s} \subseteq [\![(t,T)]\!]^{\mathcal{K}}_{s}$ it follows that $I \in [\![(t,T)]\!]^{\mathcal{K}}_{s}$. Then there exists a transition

$$t \overset{\alpha,\ell'}{\dashrightarrow} t''$$

such that $(i', I) \in [\![(t'',T)]\!]^{\mathcal{K}}_{s}$ and $m \in [\![\ell']\!]$. Now, we have

$$t \overset{\alpha,\ell}{\dashrightarrow} t' \text{ and } t \overset{\alpha,\ell'}{\dashrightarrow} t''$$

such that $m \in [\![\ell]\!] \cap [\![\ell']\!]$, hence ℓ and ℓ' are unifiable. As T is deterministic it follows that $\ell = \ell'$ and $t' = t''$, so $(i', I) \in [\![(t',T)]\!]^{\mathcal{K}}_{s}$. Finally, from $(i', I) \leq^{\mathcal{K}}_{s} I'$ and Lemma 5.3.3 it follows that $I' \leq^{\mathcal{K}}_{s} (i', I)$, and hence $I' \in [\![(t',T)]\!]^{\mathcal{K}}_{s}$.

Now we show that R is a relation witnessing $S \leq^{\mathcal{K}}_{s} T$. Clearly $(s_0, t_0) \in R$. Let $(s,t) \in R$.

1. Assume $s \overset{\alpha,k}{\dashrightarrow} s'$. Then, for each final weight $m \in \llbracket k \rrbracket$, there exists an implementation $I_m \in \llbracket (s,S) \rrbracket_s^{\mathscr{K}}$ such that
$$i_0 \xrightarrow{\alpha,m} i'.$$
We also know that $I_m \in \llbracket (t,T) \rrbracket_s^{\mathscr{K}}$ because $\llbracket (s,S) \rrbracket_s^{\mathscr{K}} \subseteq \llbracket (t,T) \rrbracket_s^{\mathscr{K}}$. Hence there exists a transition
$$t \overset{\alpha,\ell_m}{\dashrightarrow} t'_m$$
such that $m \in \llbracket \ell_m \rrbracket$. We have to show that, for all $m \in \llbracket k \rrbracket$, the weights ℓ_m are the same. Suppose that there are $m_1, m_2 \in \llbracket k \rrbracket$ and transitions
$$t \overset{\alpha,\ell_{m_1}}{\dashrightarrow} t'_{m_1} \quad \text{and} \quad t \overset{\alpha,\ell_{m_2}}{\dashrightarrow} t'_{m_2}$$
such that $m_1 \in \llbracket \ell_{m_1} \rrbracket$ and $m_2 \in \llbracket \ell_{m_2} \rrbracket$. Then, since $m_1 \in \llbracket \ell_{m_1} \rrbracket \cap \llbracket k \rrbracket$ and $m_2 \in \llbracket \ell_{m_2} \rrbracket \cap \llbracket k \rrbracket$ and T is deterministic, it follows that $\ell_{m_1} = \ell_{m_2}$ and $t'_{m_1} = t'_{m_2}$. It follows that there is a unique transition
$$t \overset{\alpha,\ell}{\dashrightarrow} t'$$
such that $m \in \llbracket \ell \rrbracket$ for all final weights $m \in \llbracket k \rrbracket$, this means $\llbracket k \rrbracket \subseteq \llbracket \ell \rrbracket$ which implies $k \preceq \ell$ by completeness of weight refinement. Moreover, by the definition of R, we get $(s',t') \in R$.

2. Assume $t \xrightarrow{\alpha,\ell} t'$. Then, for each implementation $I \in \llbracket (t,T) \rrbracket_s^{\mathscr{K}}$ we have that there exists a transition
$$i_0 \xrightarrow{\alpha,m} i'$$
for some weight $m \in \llbracket \ell \rrbracket$. We know that $\llbracket (s,S) \rrbracket_s^{\mathscr{K}} \subseteq \llbracket (t,T) \rrbracket_s^{\mathscr{K}}$, so every implementation $(j_0, J) \in \llbracket (s,S) \rrbracket_s^{\mathscr{K}}$ has a transition
$$j_0 \xrightarrow{\alpha,m} j'.$$
It follows that S must have a transition
$$s \xrightarrow{\alpha,k} s'$$
such that $m \in \llbracket k \rrbracket$. Suppose that $\llbracket k \rrbracket \not\subseteq \llbracket \ell \rrbracket$, then there would exist an implementation $(\bar{j_0}, \bar{J}) \in \llbracket (s,S) \rrbracket_s^{\mathscr{K}}$ having
$$\bar{j_0} \xrightarrow{\alpha,n} \bar{j}'$$

with a final weight $n \in [\![k]\!]$ not belonging to $[\![\ell]\!]$, which means that there must exist another transition in T, say

$$t \overset{\alpha,\ell'}{\dashrightarrow} t''$$

such that $n \in [\![\ell']\!]$. But then we have $m \in [\![\ell]\!] \cap [\![k]\!]$ and $n \in [\![\ell']\!] \cap [\![k]\!]$ which contradicts determinism of T. Thus, we have $[\![k]\!] \subseteq [\![\ell]\!]$, which implies $k \preceq \ell$ by completeness of weight refinement. Moreover, by definition of R, we get $(s',t') \in R$. □

The desired property of the interplay between composition and refinement, required for any specification theory, is compositional refinement. In other words, strong modal refinement is a precongruence with respect to modal synchronous composition. This is formalized in the following theorem.

Theorem 5.3.8 (Compositional refinement)
For any $S, S', T, T' \in \mathcal{K}\text{-}\mathbb{WMIO}$, if $S \otimes^{\mathcal{K}} T$ is defined and $S' \leq_s^{\mathcal{K}} S$ and $T' \leq_s^{\mathcal{K}} T$, then $S' \otimes^{\mathcal{K}} T'$ is defined and $S' \otimes^{\mathcal{K}} T' \leq_s^{\mathcal{K}} S \otimes^{\mathcal{K}} T$.

Proof. It suffices to prove the case $T = T'$. Assume that R_S is a relation showing $S' \leq_s^{\mathcal{K}} S$. We define a relation $R \subseteq (St_{S'} \times St_T) \times (St_S \times St_T)$ by $((s',t),(s,t)) \in R$ if and only if $(s',s) \in R_S$ and $t \in St_T$. We show that R is a refinement relation witnessing $S' \otimes^{\mathcal{K}} T \leq_s^{\mathcal{K}} S \otimes^{\mathcal{K}} T$.

Obviously $((s'_0,t_0),(s_0,t_0)) \in R$ where s_0, s'_0, t_0 are the initial states of S, S', T, respectively. Let $((s',t),(s,t)) \in R$. We only consider those transitions where a synchronization happened during composition, the other cases are less complicated and therefore omitted.

1. Assume $(s',t) \overset{\alpha,k' \oplus \ell}{\dashrightarrow}_{S' \otimes^{\mathcal{K}} T} (\hat{s}',\hat{t})$. By the rule of composition, we have

   $$s' \overset{\alpha,k'}{\dashrightarrow}_{S'} \hat{s}' \text{ and } t \overset{\alpha,\ell}{\dashrightarrow}_{T} \hat{t}.$$

   Then, from $(s',s) \in R_S$ it follows that there exists

   $$s \overset{\alpha,k}{\dashrightarrow}_{S} \hat{s}$$

   such that $k' \preceq k$ and $(\hat{s}',\hat{s}) \in R_S$. It follows that

   - there exists a transition

     $$(s,t) \overset{\alpha,k \oplus \ell}{\dashrightarrow}_{S \otimes^{\mathcal{K}} T} (\hat{s},t),$$

   - $k' \oplus \ell \preceq k \oplus \ell$ by compositionality of the weight operator $\oplus$, and

- $((\hat{s}',t),(\hat{s},t)) \in R$.

2. Assume $(s,t) \overset{\alpha,k\oplus\ell}{\dashrightarrow}_{S\otimes^{\mathscr{K}}T} (\hat{s},\hat{t})$. By the rule of composition, we have
$$s \xrightarrow{\alpha,k}_S \hat{s} \text{ and } t \xrightarrow{\alpha,\ell}_T \hat{t}.$$
Then, from $(s',s) \in R_S$ it follows that there exists
$$s' \xrightarrow{\alpha,k'}_{S'} \hat{s}'$$
such that $k' \preceq k$ and $(\hat{s}',\hat{s}) \in R_S$. It follows that
   - there exists a transition
   $$(s',t) \xrightarrow{\alpha,k'\oplus\ell}_{S'\otimes^{\mathscr{K}}T} (\hat{s}',t),$$
   - $k' \oplus \ell \preceq k \oplus \ell$ by compositionality of the weight operator $\oplus$, and
   - $((\hat{s}',t),(\hat{s},t)) \in R$. □

5.3.2 Definition of $Th^{\mathscr{K}\text{-WMIO}}_{strong}$ and $Th^{d\mathscr{K}\text{-WMIO}}_{strong}$

The last ingredient for a specification theory for $\mathscr{K}$-WMIOs is environment correctness which strongly depends on the interpretation of weights and which property environment correctness should express. It should be clear that the definition of environment correctness must be carefully designed such that it is preserved by refinement. In our example we instantiate the weight structure to integer intervals with componentwise addition of interval bounds as the synchronization operator – an intepretation that is suitable if weights are considered as resource consumption. In this setting an obvious definition of environment correctness is to just ignore transition weights and resort to environment correctness of MIOs. Formally, for $\mathscr{K}$-WMIOs S and E we can define $S \rightarrow^{\mathscr{K}}_{s} E$ by $S\downarrow \rightarrow_s E\downarrow$ where $S\downarrow$ and $E\downarrow$ are MIOs that are obtained by removing any transition weights from S and E, respectively. However, we note that other more involved definitions of environment correctness are possible but for this thesis we stick to the above definition.[1]

[1] Environment correctness notions must be carefully chosen such that they are preserved by refinement. For instance, if integer intervals are interpreted as time units after which the action can be performed, then an obvious definition of $S \rightarrow^{\mathscr{K}}_{s} E$ is that whenever S issues an output with transition weight k, then E must be ready to accept it with transition weight ℓ such that $k \leq^{\mathscr{K}}_{s} \ell$. This would formalize the requirement that the environment E should be less restrictive than S regarding the time units after which the action is performed. This environment correctness notion would not be preserved by our refinement in which we are allowed to shrink intervals. Hence, for this interpretation, other notions of refinement might be needed; this is out of the scope of this thesis.

Corollary 5.3.9
Let $\mathcal{K} = (K, \preceq, \oplus)$ be a weight structure. Then

$$Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}} \triangleq (\mathcal{K}\text{-}\mathbb{WMIO}, \mathcal{K}\text{-}\mathbb{WMIO}^i, \leq_s^{\mathcal{K}}, \otimes^{\mathcal{K}}, \rightarrow_s^{\mathcal{K}}) \text{ and}$$
$$Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIO}} \triangleq (d\mathcal{K}\text{-}\mathbb{WMIO}, d\mathcal{K}\text{-}\mathbb{WMIO}^i, \leq_s^{\mathcal{K}}, \otimes^{\mathcal{K}}, \rightarrow_s^{\mathcal{K}})$$

are specification theories.

Proof. Compositionality of refinement is shown in Theorem 5.3.8. Preservation of environment correctness follows from the definition of $\rightarrow_s^{\mathcal{K}}$ and strong modal refinement $\leq_s^{\mathcal{K}}$: if weights on transitions are not considered, $\rightarrow_s^{\mathcal{K}}$ and $\leq_s^{\mathcal{K}}$ concide with $\rightarrow_s$ and $\leq_s$, and the latter fulfil the requirement of preservation. Any implementation in $\mathcal{K}\text{-}\mathbb{WMIO}^i$ is a final element with respect to strong modal refinement, see Lemma 5.3.3. All results similarly hold for deterministic $\mathcal{K}$-WMIOs; in particular, note that modal synchronous composition of deterministic $\mathcal{K}$-WMIOs yields a deterministic $\mathcal{K}$-WMIO. □

In order to relate the introduced specification theories, we can define a reflective embedding of $Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIO}}$ in $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ by the identity function. The proof is straightforward and omitted.

Corollary 5.3.10
The identity function $id : d\mathcal{K}\text{-}\mathbb{WMIO} \rightarrow \mathcal{K}\text{-}\mathbb{WMIO}$ is a reflective embedding of $Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIO}}$ in $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}}$.

Finally, we integrate $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ and $Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIO}}$ into our existing hierarchy of modal specification theories. Clearly, $Th_{strong}^{\mathbb{MIO}}$ can be related to $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ by defining a function $g_k : \mathbb{MIO} \rightarrow \mathcal{K}\text{-}\mathbb{WMIO}$ as follows: For any MIO S, $g_k(S)$ is the $\mathcal{K}$-WMIO which results from labelling all transitions with a fixed final weight $k \in \mathcal{K}^i$ which is a neutral element for $\oplus$, i.e. $k \oplus \ell = \ell$ for all $\ell \in K$. Such a final weight, which is in addition a neutral element for $\oplus$, is given, for instance, by $[0,0]$ in $\mathcal{K}_{intv}$.

Corollary 5.3.11
The function g_k as defined above, with a final weight $k \in K$ which is a neutral element for $\oplus$, is a reflective embedding

- *of $Th_{strong}^{\mathbb{MIO}}$ in $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ and*
- *of $Th_{strong}^{d\mathbb{MIO}}$ in $Th_{strong}^{d\mathcal{K}\text{-}\mathbb{WMIO}}$.*

Proof. To prove the first reflective embedding, we have to check whether

$$g_k(S \otimes T) = g_k(S) \otimes^{\mathcal{K}} g_k(T)$$

for all $S,T \in \mathbb{MIO}$. Let us prove the assertion for may-transitions, for must-transitions the argument is analogous.

$$\begin{aligned}
(s,t) \overset{\alpha,k}{\dashrightarrow}_{g_k(S\otimes T)} (s',t') \quad & \text{if and only if} \quad (s,t) \overset{\alpha}{\dashrightarrow}_{S\otimes T} (s',t') \\
& \text{if and only if} \quad s \overset{\alpha}{\dashrightarrow}_{S} s' \text{ and } t \overset{\alpha}{\dashrightarrow}_{T} t' \\
& \text{if and only if} \quad s \overset{\alpha,k}{\dashrightarrow}_{g_k(S)} s' \text{ and } t \overset{\alpha,k}{\dashrightarrow}_{g_k(T)} t' \\
& \text{if and only if} \quad (s,t) \overset{\alpha,k\oplus k}{\dashrightarrow}_{g_k(S)\otimes^{\mathscr{K}} g_k(T)} (s',t')
\end{aligned}$$

Since k is a neutral element for $\oplus$ we get that $k \oplus k = k$. The other conditions can be shown as well. The second reflective embedding similarly holds. □

The new hierarchy with the above results integrated is shown in Figure 5.5.

$$\begin{tikzcd}
Th^{\mathbb{MIOD}}_{weak} & Th^{\mathbb{MIO}}_{weak} \arrow[l, hook, -{Triangle[open]}] & \\
Th^{\mathbb{MIOD}}_{strong} \arrow[u, hook] & Th^{\mathbb{MIO}}_{strong} \arrow[u, hook] \arrow[l, hook, -{Triangle[open]}] \arrow[r, hook, -{Triangle[open]}] & Th^{\mathscr{K}\text{-}\mathbb{WMIO}}_{strong} \\
Th^{d\mathbb{MIOD}}_{strong} \arrow[u, hook, -{Triangle[open]}] & Th^{d\mathbb{MIO}}_{strong} \arrow[u, hook, -{Triangle[open]}] \arrow[l, hook, -{Triangle[open]}] \arrow[r, hook, -{Triangle[open]}] & Th^{d\mathscr{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[u, hook, -{Triangle[open]}]
\end{tikzcd}$$

Figure 5.5: The updated hierarchy of modal specification theories

5.4 Logical Characterization of Strong Modal Refinement

It was shown in [121] that Hennessy-Milner logic [100] can be used as a logical characterization for modal refinement of modal transition systems (the reader may also consult [42]). In this section we shall extend this result to $\mathscr{K}$-WMIOs and study other related topics. For the rest of this section, we fix a weight structure $\mathscr{K} = (K, \preceq, \oplus)$.

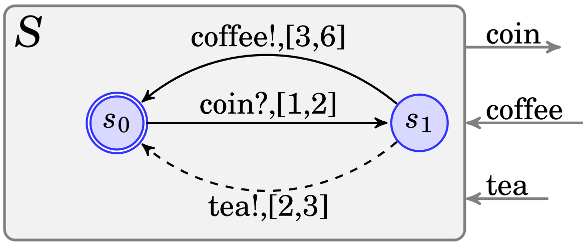


Figure 5.6: Simple vending machine with specification of energy consumption

Let us first introduce $\mathscr{K}$-HML, an extension of Hennessy-Milner logic (HML) that is interpreted over $\mathscr{K}$-WMIOs, taking into account their weights. The syntax of the logic is given by the abstract syntax

$$\varphi ::= \mathit{true} \mid \mathit{false} \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \langle \alpha, k \rangle \psi \mid [\alpha, k]\psi$$

where $k \in K$ is a label. We define the $\vee$-free fragment of $\mathscr{K}$-HML as a set of formulae in $\mathscr{K}$-HML not containing the disjunction operator.

The formula $\langle \alpha, k \rangle \psi$ intuitively means that there exists a must-transition in the current state labelled with α and a weight ℓ such that $[\![\ell]\!] \subseteq [\![k]\!]$ and the next state satisfies ψ. The formula $[\alpha, k]\psi$ expresses that for all may-transitions labelled by α and a weight ℓ such that $[\![k]\!] \cap [\![\ell]\!] \neq \emptyset$ (which means that there are common final weights of k and ℓ), the formula ψ must hold in the next state. Formally, the satisfaction relation between a state $s \in St_S$ of a $\mathscr{K}$-WMIO S and a formula φ is defined inductively as follows.

$$\begin{array}{lll}
s \models \mathit{true} & & \\
s \not\models \mathit{false} & & \\
s \models \varphi_1 \wedge \varphi_2 & \text{iff} & S \models \varphi_1 \text{ and } S \models \varphi_2 \\
s \models \varphi_1 \vee \varphi_2 & \text{iff} & S \models \varphi_1 \text{ or } S \models \varphi_2 \\
s \models \langle \alpha, k \rangle \varphi & \text{iff} & \exists (s \xrightarrow{\alpha,\ell} s') : [\![\ell]\!] \subseteq [\![k]\!] \text{ and } s' \models \varphi \\
s \models [\alpha, k]\varphi & \text{iff} & \forall (s \overset{\alpha,\ell}{\dashrightarrow} s') : \text{if } [\![\ell]\!] \cap [\![k]\!] \neq \emptyset \text{ then } s' \models \varphi
\end{array}$$

We write $S \models \varphi$ iff $s_0 \models \varphi$ where s_0 is the initial state of S.

Example 5.4.1
Consider the $\mathscr{K}_{intv}$-WMIO S of a vending machine given in Figure 5.6. After inserting a coin which consumes between 1 *and* 10 *energy units,*

- *a coffee must be possible to be dispensed with energy consumption between* 4 *and* 5*:*

$$S \models [\text{coin?}, [1, 10]]\langle \text{coffee!}, [4, 5] \rangle \mathit{true}$$

- *However, the possibility to dispense a tea consuming between* 1 *or* 2 *energy units is not guaranteed:*

  $$S \not\models [\text{coin?}, [1,0]]\langle \text{tea!}, [1,2]\rangle \mathit{true}$$

  Observe that the must-transition labelled with tea!, [2,3] *can be refined by a must-transition labelled with* tea!, [3,3].

We are now ready to prove the soundness and completeness theorems for our logic. The following theorem ensures soundness of $\mathcal{K}$-HML, i.e., if a formula holds for a specification, then it holds for any of its refinements.

Theorem 5.4.2 (Soundness)
Let $T \in \mathcal{K}$-$\mathbb{WMIO}$, and φ be a formula in $\mathcal{K}$-HML. Then

$$T \models \varphi \quad \Longrightarrow \quad \forall S \leq_s^{\mathcal{K}} T : S \models \varphi \, .$$

In the proofs in this chapter we frequently use the following notation. For a given $\mathcal{K}$-WMIO S and a state $s \in S$, we write (s,S) to mean S with the initial state replaced by s.

Proof. We prove the claim by induction on the structure of φ. Assume that $T \models \varphi$ and $S \leq_s^{\mathcal{K}} T$. Let s_0 and t_0 be the initial states of S and T, respectively. The induction basis, where $\varphi = \mathit{true}$ and $\varphi = \mathit{false}$, is trivial.

$\varphi = \varphi_1 \wedge \varphi_2$. By the definition of $\models$ and then from the induction hypothesis.

$\varphi = \varphi_1 \vee \varphi_2$. As in the case above.

$\varphi = \langle \alpha, k\rangle \psi$. From $T \models \langle \alpha, k\rangle \psi$ it follows that there exists

$$t_0 \xrightarrow{\alpha,\ell} t$$

such that $[\![\ell]\!] \subseteq [\![k]\!]$ and $t \models \psi$. Since $S \leq_s^{\mathcal{K}} T$ there exists

$$s_0 \xrightarrow{\alpha,\ell'} s$$

such that $\ell' \preceq \ell$ and $(s,S) \leq_s^{\mathcal{K}} (t,T)$. By the induction hypothesis we get $s \models \psi$. From transitivity of $\preceq$ we have $[\![\ell']\!] \subseteq [\![k]\!]$ and therefore $S \models \langle \alpha, k\rangle \psi$.

$\varphi = [\alpha, k]\psi$. Let $s_0 \overset{\alpha,\ell}{\dashrightarrow} s$ such that $[\![k]\!] \cap [\![\ell]\!] \neq \emptyset$. Since $S \leq_s^{\mathcal{K}} T$ we know that there exists

$$t_0 \overset{\alpha,\ell'}{\dashrightarrow} t$$

in T with $\ell \preceq \ell'$ and $(s,S) \leq_s^{\mathcal{K}} (t,T)$. Clearly, $[\![k]\!] \cap [\![\ell']\!] \neq \emptyset$ and because $T \models [\alpha,k]\psi$ we know that $t \models \psi$. By the induction hypothesis we get $s \models \psi$ and hence $S \models [\alpha,k]\psi$. □

We shall now focus on the issue of completeness. We consider two possible definitions:

1. *Completeness with respect to implementations*: if all implementations of a specification S satisfy a formula of the logic, then so does S.
2. *Completeness with respect to modal refinement*: if all formulae satisfied by some specification S are satisfied also by another specification T, then $S \leq_s^{\mathcal{K}} T$.

The latter, completeness with respect to modal refinement, is also known as logical characterization in the literature [121].

We first study the completeness with respect to implementations and observe that $\mathcal{K}$-HML-logic is not complete in this case. This observation can already be made with the original HML for modal transition systems, and the reason for incompleteness is the disjunction.

Theorem 5.4.3
The logic $\mathcal{K}$-HML is incomplete with respect to implementations.

Proof. Let T be an $\mathcal{K}_{triv}$-WMIO consisting of a single transition $t_0 \overset{\alpha,\bullet}{\dashrightarrow} t$, with $\mathcal{K}_{triv}$ being the weight structure from Example 5.1.3. Consider the formula $\varphi = \langle \alpha, \bullet \rangle true \vee [\alpha, \bullet] false$. Since there is no must-transition from t_0 and at the same time there is a may transition, we get $t_0 \not\models \varphi$. On the other hand, any implementation of T either contains no transition at all (and then it satisfies $[\alpha, \bullet] false$) or it contains at least one outgoing transition (and then it satisfies $\langle \alpha, \bullet \rangle true$). Hence any implementation of T satisfies φ and we get the incompleteness result with respect to implementations. □

Inspecting the proof of the above theorem, one can notice that it is the disjunction that breaks the completeness property. In fact, we can show completeness if we consider the $\vee$-free fragment of $\mathcal{K}$-HML.

Theorem 5.4.4 (Relativized completeness w.r.t. $[\![\cdot]\!]_s^{\mathcal{K}}$)
Let $T \in \mathcal{K}$-$\mathbb{WMIO}$ and let φ be a $\vee$-free formula in $\mathcal{K}$-HML. Then

$$(\forall I \in [\![T]\!]_s^{\mathcal{K}} : I \models \varphi) \quad \Longrightarrow \quad T \models \varphi \,.$$

Proof. We prove the contraposition. We show that for any $\vee$-free formula φ

$$\text{if } T \not\models \varphi \text{ then there exists } I \in [\![T]\!]_s^{\mathcal{K}} \text{ such that } I \not\models \varphi \,.$$

The proof is by induction on the structure of the formula φ and under the assumption that $T \not\models \varphi$ we construct an implementation (i_0, I) if T such that $i_0 \not\models \varphi$. During the construction we will write that we add a transition $i_0 \xrightarrow{n} (i_0', I')$ for an implementation (i_0', I'), meaning that together with this transition we implicitly add also a disjoint copy of I' rooted at i_0' to the implementation I.

The induction basis, where $\varphi = true$ and $\varphi = false$, is trivial.

$\varphi = \varphi_1 \wedge \varphi_2$. By the definition of $\models$ either $T \not\models \varphi_1$ or $T \not\models \varphi_2$ Assume w.l.o.g. that $T \not\models \varphi_1$. By applying the induction hypothesis there is $I \in [\![T]\!]_s^{\mathscr{K}}$ such that $I \not\models \varphi_1$ and we conclude that $I \not\models \varphi_1 \wedge \varphi_2$.

$\varphi = \langle \alpha, k\rangle \psi$. Assume that $T \not\models \langle \alpha, k\rangle\psi$, which is the case if for all $t_0 \xrightarrow{\alpha,\ell} t$ we have either (1) $[\![\ell]\!] \not\subseteq [\![k]\!]$ or (2) $(t,T) \not\models \psi$. We construct an implementation $(i_0, I) \in [\![T]\!]_s^{\mathscr{K}}$ as follows. For every

$$t_0 \xrightarrow{\alpha,\ell} t$$

such that (1) is satisfied, we add the transition

$$i_0 \xrightarrow{\alpha,n} (i'_0, I')$$

into I where $n \in [\![\ell]\!] \smallsetminus [\![k]\!]$ and $(i'_0, I') \leq_s^{\mathscr{K}} (t,T)$. For every

$$t_0 \xrightarrow{\alpha,\ell} t$$

such that (2) is satisfied, we have by induction hypothesis an implementation $(i'_0, I') \in [\![(t,T)]\!]_s^{\mathscr{K}}$ such that $i'_0 \not\models \psi$. We add

$$i_0 \xrightarrow{\alpha,m} (i'_0, I')$$

to I for some $m \in [\![\ell]\!]$. It is easy to see that $I \leq_s^{\mathscr{K}} T$, and moreover $I \not\models \langle \alpha, k\rangle\psi$ by the construction.

$\varphi = [\alpha, k]\psi$. Assume that $T \not\models [\alpha,k]\psi$. Then there exists $t_0 \overset{\alpha,\ell}{\dashrightarrow} t$ such that $[\![\ell]\!] \cap [\![k]\!] \neq \emptyset$ and $(t,T) \not\models \psi$. By induction hypothesis there exists $(i'_0, I') \in [\![(t,T)]\!]_s^{\mathscr{K}}$ such that $i'_0 \not\models \psi$. Let $(i_0, I) \in [\![T]\!]_s^{\mathscr{K}}$ be some implementation of T where we add the transition

$$i_0 \xrightarrow{\alpha,n} (i'_0, I')$$

with $n \in [\![\ell]\!] \cap [\![k]\!]$. Clearly, we still have $I \leq_s^{\mathscr{K}} T$ and moreover the transition

$$i_0 \xrightarrow{\alpha,n} i'_0$$

ensures that $i_0 \not\models [\alpha,k]\psi$. □

A similar completeness result in the setting of partial Kripke structures can be found also in [6].

We now study completeness with respect to modal refinement, that is the completeness definition considered in [121]. In this article, it was shown that for classical modal transition systems HML is complete with respect to refinement. We first observe that the result does not extend to $\mathscr{K}$-HML and $\mathscr{K}$-WMIOs.

Theorem 5.4.5
$\mathscr{K}$-HML is incomplete with respect to strong modal refinement.

Proof. Consider the systems S and T from Figure 5.4. By case analysis it is easy to verify that $s_0 \models \varphi$ if and only if $t_0 \models \varphi$ for any $\mathscr{K}$-HML-formula φ. However, as argued before, $S \not\leq_s^{\mathscr{K}} T$. □

On the other hand, if we consider only deterministic systems, $\mathscr{K}$-HML is complete even with disjunction, as proved below. We let $\mathscr{F}(S) = \{\varphi \mid S \models \varphi\}$ denote the set of all formulae in $\mathscr{K}$-HML satisfied by S.

Theorem 5.4.6 (Relativized completeness of strong modal refinement)
Let $S, T \in \mathscr{K}$-$\mathbb{WMIO}$ be deterministic $\mathscr{K}$-WMIOs (see Definition 5.3.5) and assume that the weight refinement relation $\preceq$ is complete. Then

$$\mathscr{F}(T) \subseteq \mathscr{F}(S) \quad \Longrightarrow \quad S \leq_s^{\mathscr{K}} T\ .$$

Proof. Assume that $\mathscr{F}(T) \subseteq \mathscr{F}(S)$. We define a relation $R \subseteq S \times T$ by

$$R = \{(s,t) \mid \mathscr{F}((t,T)) \subseteq \mathscr{F}((s,S))\}.$$

We show that R is a relation witnessing $S \leq_s^{\mathscr{K}} T$. Clearly $(s_0, t_0) \in R$ for the respective initial states. Let $(s,t) \in R$.

- First, assume that $s \overset{\alpha,k}{\dashrightarrow} s'$. Clearly, $t \overset{\alpha,\ell}{\dashrightarrow} t'$ for some ℓ such that $[\![k]\!] \cap [\![\ell]\!] \neq \emptyset$, otherwise the formula $[\alpha,k]\mathit{false}$ is satisfied in (t,T) but not in (s,S), contradicting our assumption that $\mathscr{F}((t,T)) \subseteq \mathscr{F}((s,S))$. By the determinism of T there can only be one such ℓ with $[\![k]\!] \cap [\![\ell]\!] \neq \emptyset$.

  For the sake of contradiction assume that $k \not\preceq \ell$. By completeness of weight refinement we get $[\![k]\!] \not\subseteq [\![\ell]\!]$. Thus, there exists some $m \in [\![k]\!] \setminus [\![\ell]\!]$. The formula $[\alpha,m]\mathit{false}$ holds in (t,T) due to the choice of m and the absence of any other may transition having any common final weights with $[\![k]\!]$, hence in particular also with $[\![m]\!] = \{m\}$. However, $[\alpha,m]\mathit{false}$ does not hold in (s,S), contradicting the assumption that $\mathscr{F}((t,T)) \subseteq \mathscr{F}((s,S))$. Thus, we can assume the existence of

  $$t \overset{\alpha,\ell}{\dashrightarrow} t'$$

  with $k \preceq \ell$.

  Now we need to argue that $(s',t') \in R$. Assume that this is not the case. Then we have $\mathscr{F}((t',T)) \not\subseteq \mathscr{F}((s',S))$, and therefore there is a formula φ' such that $(t',T) \models \varphi'$ and $(s',S) \not\models \varphi'$. Consider the formula $\varphi = [\alpha,k]\varphi'$. Again $(t,T) \models \varphi$, but $(s,S) \not\models \varphi$. This contradicts the assumption that

  $$\mathscr{F}((t,T)) \subseteq \mathscr{F}((s,S)).$$

  Thus $(s',t') \in R$.

- Second, assume that $t \xrightarrow{\alpha,\ell} t'$. As in the previous item, there must be a transition
$$s \xrightarrow{\alpha,k} s'$$
such that $[\![k]\!] \subseteq [\![\ell]\!]$, otherwise the formula $\langle \alpha, \ell \rangle true$ is satisfied in (t,T) but not in (s,S), contradicting the assumption that $\mathscr{F}((t,T)) \subseteq \mathscr{F}((s,S))$. By the determinism of S we know that
$$s \xrightarrow{\alpha,k} s'$$
is a unique transition such that $[\![k]\!] \subseteq [\![\ell]\!]$ and due to the completeness of weight refinement we know that $k \preceq \ell$.

  It remains to argue that $(s',t') \in R$. Assume that this is not the case. The arguments are similar to the previous case by considering the formula $\langle \alpha, \ell \rangle \varphi'$ where $t' \models \varphi'$ and $s' \not\models \varphi'$. Thus $(s',t') \in R$ and this completes the proof. □

5.5 The Specification Theory $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$

5.5.1 Weak Modal Refinement

We finally propose a specification theory for $\mathcal{K}$-WMIOs with weak modal refinement and weak environment correctness. Weak modal refinement for $\mathcal{K}$-WMIOs is defined along the lines of weak modal refinement for MIOs, see Section 3.4.1 in Chapter 3, however, weights can be refined and can – in contrast to strong modal refinement for $\mathcal{K}$-WMIOs – be distributed via $\oplus$ to several transitions. Intuitively, if weights model for instance energy consumption, then a must-transition
$$t \xrightarrow{\alpha,\ell}_T t'$$
can be refined to a sequence
$$s_1 \xrightarrow{\tau,k_1}_S \dots s_m \xrightarrow{\tau,k_m}_S s_{m+1} \xrightarrow{\alpha,k_{m+1}}_S s_{m+2} \xrightarrow{\tau,k_{m+2}}_S \dots s_n \xrightarrow{\tau,k_n}_S s_{n+1},$$
such that the accumulated resource consumption of this path is below the specified resource consumpion ℓ on the abstract level, i.e.
$$k_1 \oplus \dots \oplus k_n \preceq \ell.$$

Definition 5.5.1 (Weak modal refinement)
Let $S,T \in \mathcal{K}$-$\mathbb{WMIO}$ with the same sets of input and output actions. S is a weak modal refinement of T, written $S \leq^{\mathcal{K}}_{w} T$, if there exists a refinement relation $R \subseteq St_S \times St_T$ with $(s_0,t_0) \in R$ such that for all $(s,t) \in R$ and all $\alpha \in (\bigcup \Sigma_S) \cup (\bigcup \Sigma_T)$:

1. *for all* $s' \in St_S$*, for all* $k \in K$ *such that* $s \overset{\alpha,k}{\dashrightarrow}_S s'$*,*
   - *if* $\alpha \in \Sigma_S^{int}$ *then*
     - *either* k *is neutral for* $\oplus$ *and* $(s',t) \in R$*,*
     - *or there exists* $t' \in St_T$ *such that*

$$t \overset{(\tau,\ell_1)...(\tau,\ell_n)}{\dashrightarrow}_T t',$$

       $k \preceq \ell_1 \oplus \ldots \oplus \ell_n$ *and* $(s',t') \in R$*;*
   - *if* $\alpha \in \Sigma_S^{ext}$ *then there exists* $t' \in St_T$ *such that*

$$t \overset{(\tau,\ell_1)..(\tau,\ell_m)(\alpha,\ell_{m+1}),(\tau,\ell_{m+2}),...,(\tau,\ell_n)}{\dashrightarrow}_T t',$$

     $k \preceq \ell_1 \oplus \ldots \oplus \ell_n$ *and* $(s',t') \in R$*;*

2. *for all* $t' \in St_T$*, for all* $\ell \in K$ *such that* $t \xrightarrow{\alpha,\ell}_T t'$*,*
   - *if* $\alpha \in \Sigma_T^{int}$ *then*
     - *either* k *is neutral for* $\oplus$ *and* $(s,t') \in R$*,*
     - *or there exists* $s' \in St_S$ *such that*

$$s \xrightarrow{(\tau,k_1)...(\tau,k_n)}_S s',$$

       $k_1 \oplus \ldots \oplus k_n \preceq \ell$ *and* $(s',t') \in R$*;*
   - *if* $\alpha \in \Sigma_T^{ext}$ *then there exists* $s' \in St_S$ *such that*

$$s \xrightarrow{(\tau,k_1)..(\tau,k_m)(\alpha,k_{m+1}),(\tau,k_{m+2}),...,(\tau,k_n)}_S s',$$

     $k_1 \oplus \ldots \oplus k_n \preceq \ell$ *and* $(s',t') \in R$.

It is straightforward to show that weak modal refinement $\leq_w^{\mathcal{K}}$ is reflexive and transitive. Finality of implementations is shown in the following lemma.

Lemma 5.5.2
For any $I \in \mathcal{K}\text{-}\mathbb{WMIO}^i$ *and any* $S \in \mathcal{K}\text{-}\mathbb{WMIO}$*, if* $S \leq_w^{\mathcal{K}} I$ *then* $I \leq_w^{\mathcal{K}} S$*.*

Proof. Let R be a refinement relation proving $S \leq_w^{\mathcal{K}} I$. Then it is straighforward to prove that $R^{-1} = \{(i,s) \mid (s,i) \in R\}$ is a refinement relation demonstrating $I \leq_w^{\mathcal{K}} S$. Clearly, $(i_0,s_0) \in R^{-1}$. Let $(i,s) \in R^{-1}$. We only consider may-transitions, must-transitions can be checked analogously. Assume that

$$s \xrightarrow{\alpha,\ell}_S s'$$

for some $\alpha \in \Sigma_S^{ext}$ and some $\ell \in K$. Then this is also a may-transition, and $(s,i) \in R$ implies that there are transitions

$$i \overset{(\tau,k_1)..(\tau,k_m)(\alpha,k_{m+1}),(\tau,k_{m+2}),...,(\tau,k_n)}{\dashrightarrow}_I i'$$

such that $\ell \preceq k_1 \oplus \ldots \oplus k_n$ and $(s',i') \in R$. Since $\oplus$ is closed under final weights, $k_1 \oplus \ldots \oplus k_n$ is a final weight and hence $\ell = k_1 \oplus \ldots \oplus k_n$. All may-transitions in I are also must-transitions, and $(i',s') \in R^{-1}$ by definition of R^{-1}. □

Theorem 5.5.3 (Compositional refinement)
Let $S,S',T,T' \in \mathcal{K}$-$\mathbb{WMIO}$. If $S \otimes^{\mathcal{K}} T$ is defined and $S' \leq_w^{\mathcal{K}} S$ and $T' \leq_w^{\mathcal{K}} T$, then $S' \otimes^{\mathcal{K}} T'$ is defined and $S' \otimes^{\mathcal{K}} T' \leq_w^{\mathcal{K}} S \otimes^{\mathcal{K}} T$.

Proof. It suffices to prove the case $T = T'$. Assume that R_S is a relation showing $S' \leq_w^{\mathcal{K}} S$. We define a relation $R \subseteq (St_{S'} \times St_T) \times (St_S \times St_T)$ by $((s',t),(s,t)) \in R$ if and only if $(s',s) \in R_S$. We show that R is a refinement relation witnessing $S' \otimes^{\mathcal{K}} T \leq_s^{\mathcal{K}} S \otimes^{\mathcal{K}} T$.

Obviously, $((s_0',t_0),(s_0,t_0)) \in R$ where s_0,s_0',t_0 are the initial states of S, S', T, respectively. Let $((s',t),(s,t)) \in R$. We only check the may-transitions, i.e. the direction from concrete to abstract; the proof for must-transitions is analogous. Moreover, we only consider those may-transitions where a synchronization happened during composition; all other cases are less complicated and therefore omitted.

Assume $(s',t) \overset{\alpha,k'\oplus\ell}{\dashrightarrow}_{S'\otimes^{\mathcal{K}}T} (\hat{s}',\hat{t})$. By the rule of composition, we have

$$s' \overset{\alpha,k'}{\dashrightarrow}_{S'} \hat{s}' \text{ and } t \overset{\alpha,\ell}{\dashrightarrow}_T \hat{t}.$$

Then, from $(s',s) \in R_S$ it follows that there exists

$$s \overset{(\tau,k_1)...(\tau,k_m)(\alpha,k_{m+1})(\tau,k_{m+2})...(\tau,k_n)}{\dashrightarrow}_S \hat{s}$$

such that $k' \preceq k_1 \oplus \ldots \oplus k_n$ and $(\hat{s}',\hat{s}) \in R_S$. It follows that

- there exist transitions

$$(s,t) \overset{(\tau,k_1)...(\tau,k_m)(\alpha,k_{m+1}\oplus\ell)(\tau,k_{m+2})...(\tau,k_n)}{\dashrightarrow}_{S\otimes^{\mathcal{K}}T} (\hat{s},t),$$

- $k' \oplus \ell \preceq k_1 \oplus \ldots \oplus k_n \oplus \ell$ by compositionality of the weight operator $\oplus$, and

- $((\hat{s}',t),(\hat{s},t)) \in R$. □

5.5.2 Definition of $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$

Weak modal refinement gives rise to a new specification theory for $\mathcal{K}$-WMIO, with weak modal refinement and weak environment correctness $\rightarrow^{\mathcal{K}}_{w}$, defined by $S \rightarrow^{\mathcal{K}}_{w} E$ if $S\downarrow \rightarrow_{w} E\downarrow$ for any $S, E \in \mathcal{K}\text{-}\mathbb{WMIO}$.[2]

Corollary 5.5.4
$Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak} \triangleq (\mathcal{K}\text{-}\mathbb{WMIO}, \mathcal{K}\text{-}\mathbb{WMIO}^{i}, \leq^{\mathcal{K}}_{w}, \otimes^{\mathcal{K}}, \rightarrow^{\mathcal{K}}_{w})$ is a specification theory.

Proof. Compositionality of refinement is shown in Theorem 5.5.3. Preservation of environment correctness follows from the preservation of $\rightarrow_{w}$ by $\leq_{w}$ since transition weights are not considered in the definition of $\rightarrow^{\mathcal{K}}_{w}$. Finality of implementations is shown in Lemma 5.5.2. □

We conclude this section by relating the newly obtained specification theory $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$ to the other theories of the last chapters.

Lemma 5.5.5
The identity function $id : \mathcal{K}\text{-}\mathbb{WMIO} \rightarrow \mathcal{K}\text{-}\mathbb{WMIO}$ is an embedding of $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{strong}$ in $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$.

Proof. *id* is injective. Moreover, *id* is a morphism: Conditions (1) and (2), see Definition 2.1.3 in Chapter 2, are trivial. For condition (3), the reader can easily verify that $S \leq^{\mathcal{K}}_{s} T$ implies $S \leq^{\mathcal{K}}_{w} T$. Condition (4) was already shown for the embedding *id* of $Th^{\mathbb{MIO}}_{strong}$ in $Th^{\mathbb{MIO}}_{weak}$. □

Second, we can reuse the function g_k (for a final weight $k \in \mathcal{K}^{i}$ which is a neutral element for $\oplus$) of Section 5.3.2 to obtain a reflective embedding of $Th^{\mathbb{MIO}}_{weak}$ in $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$ as follows: given a MIO S, $g_k(S)$ is the $\mathcal{K}$-WMIO which results from S by adding $k \in \mathcal{K}^{i}$ to every transition.

Lemma 5.5.6
The function $g_k : \mathbb{MIO} \rightarrow \mathcal{K}\text{-}\mathbb{WMIO}$ as defined above is a reflective embedding of $Th^{\mathbb{MIO}}_{weak}$ in $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$.

Proof. g_k is injective. Moreover, g_k is a morphism from $Th^{\mathbb{MIO}}_{weak}$ to $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$, cf. Definition 2.1.3:

1. Let $S \in \mathbb{MIO}^{i}$. Then $g_k(S) \in \mathcal{K}\text{-}\mathbb{WMIO}^{i}$ because every may-transition is also a must-transition and k with which each transition is annotated is a final weight.

[2] Recall from Section 5.3.2 that $S\downarrow$ is the MIO obtained from S by omitting all transition weights.

2. Let $S,T \in \mathbb{MIO}$. If $S \otimes T$ is defined, then $g_k(S) \otimes^{\mathcal{K}} g_k(T)$ is defined because composability only depends on the action signatures. Moreover, $g_k(S \otimes T) = g_k(S) \otimes^{\mathcal{K}} g_k(T)$ since the synchronization of transitions with weight k result in a transition with weight $k \oplus k$ which is k because k is neutral.

3. Let $S,T \in \mathbb{MIO}$. Then $S \leq_w T$ implies $g_k(S) \leq_w^{\mathcal{K}} g_k(T)$: if R is a refinement relation for $S \leq_w T$, then the same relation proves $g_k(S) \leq_w^{\mathcal{K}} g_k(T)$. Again, it is crucial that k is neutral.

4. Weak environment correctness is trivially preserved by g_k as $\rightarrow_w^{\mathcal{K}}$ does not depend on the weights.

Second, we show that g_k satisfies the conditions of a reflective embedding, cf. Definition 2.1.5.

1. Let $S \in \mathbb{MIO}$. If $g_k(S) \in \mathcal{K}\text{-}\mathbb{WMIO}^i$, then every may-transition is also a must-transition, hence $S \in \mathbb{MIO}^i$.

2. Let $S,T \in \mathbb{MIO}$. If $g_k(S) \otimes^{\mathcal{K}} g_k(T)$ is defined, then also $S \otimes T$ is defined and $g_k(S \otimes T) = g_k(S) \otimes^{\mathcal{K}} g_k(T)$ because k is a neutral element for $\oplus$.

3. Let $S,T \in \mathbb{MIO}$. If $g_k(S) \leq_w^{\mathcal{K}} g_k(T)$, then $S \leq_w T$ which is witnessed by the same refinement relation that proves $g_k(S) \leq_w^{\mathcal{K}} g_k(T)$.

4. For $S,T \in \mathbb{MIO}$, $S \rightarrow_w T$ trivially holds as soon as $g_k(S) \rightarrow_w^{\mathcal{K}} g_k(T)$ is satisfied because weights are not considered in $\rightarrow_w^{\mathcal{K}}$. □

The result that *id* is an embedding of $Th_{strong}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ in $Th_{weak}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ and g_k is a reflective embedding of $Th_{weak}^{\mathbb{MIO}}$ in $Th_{weak}^{\mathcal{K}\text{-}\mathbb{WMIO}}$ is integrated in Figure 5.7.

5.6 Completing the Hierarchy of Modal Specification Theories

In this section, we complete our hierarchy of modal specification theories by integrating the data aspect addressed by MIODs and the quantitative aspect of $\mathcal{K}$-WMIODs into *$\mathcal{K}$-weighted MIOs with data constraints* and by defining specification theories for $\mathcal{K}$-WMIODs.

Definition 5.6.1 ($\mathcal{K}$-WMIOD)
*Let $\mathcal{K} = (K, \preceq, \oplus)$ be a weight structure. A $\mathcal{K}$-*weighted modal input/output automaton with data constraints *($\mathcal{K}$-WMIOD)*

$$S = (St, s_0, \varphi_0, \Sigma, V, \dashrightarrow, \longrightarrow)$$

$$\begin{tikzcd}
Th^{\mathbb{MIOD}}_{weak} & Th^{\mathbb{MIO}}_{weak} \arrow[l, hook', -{Triangle[open]}] \arrow[r, hook, -{Triangle[open]}] & Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak} \\
Th^{\mathbb{MIOD}}_{strong} \arrow[u, hook'] & Th^{\mathbb{MIO}}_{strong} \arrow[u, hook'] \arrow[l, hook', -{Triangle[open]}] \arrow[r, hook, -{Triangle[open]}] & Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[u, hook'] \\
Th^{d\mathbb{MIOD}}_{strong} \arrow[u, hook', -{Triangle[open]}] & Th^{d\mathbb{MIO}}_{strong} \arrow[u, hook', -{Triangle[open]}] \arrow[l, hook', -{Triangle[open]}] \arrow[r, hook, -{Triangle[open]}] & Th^{d\mathcal{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[u, hook', -{Triangle[open]}]
\end{tikzcd}$$

Figure 5.7: The updated hierarchy of modal specification theories

consists of a set of control states St, an initial control state $s_0 \in St_S$*, an initial data state predicate* $\varphi_0 \in \mathcal{S}(V^{prov}, \emptyset)$*, an extended action signature* $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$*, a state signature* $V = (V^{prov}, V^{req})$*, and a may- and must-transition relation*

$$\dashrightarrow, \longrightarrow \subseteq St \times \mathcal{L}(\Sigma, V) \times K \times St.$$

All notions of well-formedness, implementations, modal synchronous composition, strong and weak modal refinement, and strong and weak environment correctness can be integrated easily. As an example, we would like to make precise the integration of strong modal refinement $\le_s^d$ of MIODs and strong modal refinement $\le_s^{\mathcal{K}}$ $\mathcal{K}$-WMIOs. To illustrate the integration, we basically take Definition 4.3.1 of strong modal refinement for MIODs and add the conditions for the weights, highlighted by small framed boxes.

Definition 5.6.2 (Strong modal refinement for $\mathcal{K}$-WMIODs)
Let S and T be two well-formed $\mathcal{K}$-WMIODs with the same extended action and state signature. Then S is a strong modal refinement *of T if* $\models_\forall \varphi_{0,S} \Rightarrow \varphi_{0,T}$ *and there exists a refinement relation*

$$R \subseteq St_S \times St_T \times \mathcal{D}(V^{prov})$$

such that $(s_0, t_0, \delta_0) \in R$ *for all* $\delta_0 \in \mathcal{D}(\varphi_{0,S})$*, and for all* $(s, t, \delta) \in R$*:*

1. *For all* $t' \in St_T$, $[\varphi_T]\alpha[\pi_T] \in \mathcal{L}(\Sigma, V)$, $\boxed{\ell \in K}$, $v \in \mathcal{D}(V^{req})$ *and* $\rho \in \mathcal{D}(par(\alpha))$*, if*

$$t \xrightarrow{[\varphi_T]\alpha[\pi_T]\ \boxed{\ell}}_T t' \text{ and } (\delta \cdot v; \rho) \models \varphi_T$$

then there exist $s' \in St_S$, $[\varphi_S]\alpha[\pi_S] \in \mathscr{L}(\Sigma, V)$, $\boxed{k \in K}$ *such that*

$$s \xrightarrow{[\varphi_S]\alpha[\pi_S]\ \boxed{k}}_S s' \text{ and } (\delta\cdot v;\rho) \models \varphi_S \text{ and } \boxed{k \preceq \ell}$$

and for all $\delta' \in \mathscr{D}(V^{prov})$,

$$\text{if } (\delta\cdot v, \delta';\rho) \models \pi_S \text{ then } (\delta\cdot v, \delta';\rho) \models \pi_T \text{ and } (s', t', \delta') \in R.$$

2. *For all* $s' \in St_S$, $[\varphi_S]\alpha[\pi_S] \in \mathscr{L}(\Sigma, V)$, $\boxed{k \in K}$, $v \in \mathscr{D}(V^{req})$, $\rho \in \mathscr{D}(par(\alpha))$ *and* $\delta' \in \mathscr{D}(V^{prov})$, *if*

$$s \overset{[\varphi_S]\alpha[\pi_S]\ \boxed{k}}{\dashrightarrow}_S s' \text{ and } (\delta\cdot v, \delta';\rho) \models \varphi_S \wedge \pi_S$$

then there exist $t' \in St_T$, $[\varphi_T]\alpha[\pi_T] \in \mathscr{L}(\Sigma, V)$, $\boxed{\ell \in K}$ *such that*

$$t \overset{[\varphi_T]\alpha[\pi_T]\ \boxed{\ell}}{\dashrightarrow}_T t' \text{ and } (\delta\cdot v, \delta';\rho) \models \varphi_T \wedge \pi_T \text{ and } \boxed{k \preceq \ell}$$

and $(s', t', \delta') \in R$.

In a similar way, all other notions can be integrated in a straightforward way. All required properties for a specification theory, like compositionality of refinement and preservation of environment correctness under refinement can be easily verified along the lines of the proofs for MIODs and $\mathcal{K}$-WMIOs. To not overburden the reader with repeating many similar proofs again and again, we omit the details and directly state which modal specification theories can be obtained:

- $Th^{d\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$ for deterministic $\mathcal{K}$-WMIODs based on strong modal refinement and strong environment correctness

- $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$ for $\mathcal{K}$-WMIODs based on strong modal refinement and strong environment correctness

- $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{weak}$ for $\mathcal{K}$-WMIODs based on weak modal refinement and weak environment correctness

These new specification theories can be integrated and related to the previous specification theories as shown in Figure 5.8. The embedding of $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$ in $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{weak}$ is shown by the identity function $id : \mathcal{K}\text{-}\mathbb{WMIOD} \to \mathcal{K}\text{-}\mathbb{WMIOD}$. The other new embeddings shown in Figure 5.8 are based on (the integration of) the function f and g_k that we have used previously (see Sections 4.3.3 and 5.5.2) to relate the different specification theories for MIOs, MIODs and $\mathcal{K}$-WMIOs.

$$\begin{tikzcd}
 & & Th^{\mathscr{K}\text{-}\mathbb{WMIOD}}_{weak} & \\
Th^{\mathbb{MIOD}}_{weak} \arrow[urr, hook, -{Triangle[open]}] & Th^{\mathbb{MIO}}_{weak} \arrow[l, hook, -{Triangle[open]}] \arrow[rr, hook, -{Triangle[open]}] \arrow[ur, hook, -{Triangle[open]}] & & Th^{\mathscr{K}\text{-}\mathbb{WMIO}}_{weak} \arrow[ul, hook, -{Triangle[open]}] \\
 & & Th^{\mathscr{K}\text{-}\mathbb{WMIOD}}_{strong} \arrow[uu, hook] & \\
Th^{\mathbb{MIOD}}_{strong} \arrow[uu, hook] \arrow[urr, hook, -{Triangle[open]}] & Th^{\mathbb{MIO}}_{strong} \arrow[uu, hook] \arrow[l, hook, -{Triangle[open]}] \arrow[rr, hook, -{Triangle[open]}] \arrow[ur, hook, -{Triangle[open]}] & & Th^{\mathscr{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[uu, hook] \arrow[ul, hook, -{Triangle[open]}] \\
 & & Th^{d\mathscr{K}\text{-}\mathbb{WMIOD}}_{strong} \arrow[uu, hook, -{Triangle[open]}] & \\
Th^{d\mathbb{MIOD}}_{strong} \arrow[uu, hook, -{Triangle[open]}] \arrow[urr, hook, -{Triangle[open]}] & Th^{d\mathbb{MIO}}_{strong} \arrow[uu, hook, -{Triangle[open]}] \arrow[l, hook, -{Triangle[open]}] \arrow[rr, hook, -{Triangle[open]}] \arrow[ur, hook, -{Triangle[open]}] & & Th^{d\mathscr{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[uu, hook, -{Triangle[open]}] \arrow[ul, hook, -{Triangle[open]}]
\end{tikzcd}$$

Figure 5.8: The final hierarchy of modal specification theories

5.7 Discussion and Related Work

The idea of modelling quantitative aspects in transition systems by adding weights to transitions is well studied, see e.g. [14] for an overview. In the context of interface theories for component-based design, interface specifications including quantitative aspects other than time has been investigated only in a few works. Probably the most relevant related work is about resource interfaces [52] that extend well-known state-based interfaces by a resource algebra. In contrast to our work here, a resource label is assigned to states rather than to transitions. Resource labels in states allow for modelling, e.g., the amount of power that is consumed while the system resides in a particular state. A resource algebra, amongst others, also defines a composition operator on resource labels, hence in this sense they are equally abstract as in our approach. It is shown that resource interfaces can be instantiated to model interesting quantitative problems. Refinement and logical characterization are not studied.

Extending modal transition systems by weights is inspired by Juhl et al. [114]. They define (interval) weighted modal transition systems, which coincide with our instantiation of $\mathscr{K}_{intv}$-WMIOs except that they do not distinguish between input, output and internal actions. They study (strong) modal refinement, that coincides with our refinement notion $\leq_s^{\mathscr{K}_{intv}}$, and prove that it is complete w.r.t. the derived notion of thorough refinement – a result that follows also from our Theorem 5.3.7 for the weight structure $\mathscr{K}_{intv}$. In the second part of the paper an algorithm is presented for computing largest common refinements of weighted modal transitions systems (also known as “conjunction” of specifications, see Section 7.1 in Chapter 7). Finally, weighted CTL is suggested and strong modal refinement logically characterized.

Publication history. This chapter is based on [23] which is slightly more general than this thesis chapter. In [23] we introduced label-structured modal transition systems that extend pure modal transition systems and abstract away from actions by labelling transitions just with a weight from a fixed weight structure (in [23] weight structures are called label-sets). It is shown that MIOs can be obtained by instantiating the weight structure to actions. Moreover, in [23] we have studied conjunction and quotient of label-structured modal transition systems, and proposed a determinization algorithm yielding the least deterministic over-approximation with respect to strong modal refinement.

5.8 Summary

This chapter presents $\mathscr{K}$-weighted modal input/output automata ($\mathscr{K}$-WMIOs) that extend modal input/output automata by transitions weight from a weight structure $\mathscr{K}$. The weight structure $\mathscr{K} = (K, \preceq, \oplus)$ features weight refinement $\preceq$ and a weight synchronization operator $\oplus$ that is general enough to model, for instance, energy consumption. Modal synchronous composition and modal refinement for MIOs can be conservatively extended to $\mathscr{K}$-WMIOs giving rise to a specification theory. We have proven strong modal refinement to be equivalent to inclusion of implementation semantics, (as expected) by assuming determinism. We have suggested an extension of Hennessy-Milner logic to $\mathscr{K}$-WMIOs and have shown the interplay between the logic and the refinement theory in a similar way as known from the classical theory of labelled transition systems and bisimulation. We have proposed a specification theory for $\mathscr{K}$-WMIO based on weak modal refinement that allows to distribute weights over several transitions in a refined specification. Most of the proof techniques have been straightforward generalizations of the techniques developed for MIOs in Chapter 3. Finally, we have completed our hierarchy of modal specification theories by defining modal specification theories for $\mathscr{K}$-WMIODs, that integrate data and quantitative aspects, and integrating them into the hierarchy by suitable embeddings.

Chapter 6

Quantitative Modal Refinement

Modern reactive systems, in particular embedded systems, often operate in physically constrained and unpredictable environments. This fact causes additional challenges in the design and verification of such systems when compared to classical system design. Therefore, the design of embedded systems calls for new heterogeneous models that appropriately address functional as well as non-functional properties [106]. Quantitative constraints must be naturally expressible in the models and still allow for an effective analysis of quantitative properties of interest.

Traditional verification frameworks are based on a discrete interpretation of property satisfaction, returning either *true* or *false* as the answer to a verification problem. With regards to functional properties, like the temporal ordering of emitted and received signals, a Boolean answer is sufficient. However, non-functional properties will naturally be approximations of real-world implementations, and small inaccuracies of an implementation with respect to its specification might turn the verification answer from *true* to *false*. This observation leads us to the need for a framework in which answers to quantitative verification problems are of quantitative nature as well. The aim is to be able to achieve robustness of the satisfaction relation and to differentiate between implementations that *almost* – as opposed to *not at all* – satisfy its (quantitative) specification.

In the literature, several frameworks for quantitative analysis for reactive systems have been proposed, mainly by de Alfaro et al. [66, 64, 59, 60]. The key idea in these works is – and which we follow here, too – that relating an abstract behaviour and a refined one yields a value between 0, for "perfect refinement", and ∞ meaning "no refinement at all". This value is called the refinement distance from the concrete to the abstract behaviour.

The contribution of this chapter is the first formal foundation for quantitative specification theories based on $\mathscr{K}$-WMIOs which allow for quantitative analysis during the refinement and implementation process, thus alleviating the problems of the qualitative setting. Although the quantitative study can be carried out on the abstract level of generic weight structures we stick here to the concrete weight structure $\mathscr{K}_{intv}$ of integer intervals introduced in Example 5.1.4, Chapter 5.

Similarly, there is a wide range of concrete refinement distances that one can choose – this choice typically depends on the application domain of the quantitative constraints. In the literature, one finds point-wise distance [59] measuring the largest individual difference between specifications, accumulated distance [173] measuring the sum of individual differences along executions of the specifications, and maximum-lead distance [105] measuring the largest distance between accumulated differences along executions. For an overview of various simulation distances that exist in the literature we refer the reader to [172, 51].

In this thesis, we stick to the accumulated refinement distance [173]. This distance is defined as the fix point of recursive equations, and to ensure convergence with limits different to infinity in more cases we use a discounting factor which renders differences at later points in executions converge to zero. We finally remark that the following considerations in this chapter could have been carried out on an abstract level of distances leaving open the concrete distance to use [16].

The main result of this chapter is the demonstration how strong modal refinement for $\mathscr{K}_{intv}$-WMIOs can be generalized to modal refinement distances. Most importantly, we show that compositionality of strong modal refinement can be lifted to a similar result for modal refinement distances.

Motivation. Let us start with a motivating example illustrating the idea of modal refinement distances. First, consider the MIO shown in Figure 6.1 which models the requirements of a simple email system in which emails are first received and then delivered. Before delivering the email, the system may check or process the email, e.g. for en- or decryption, filtering of spam emails, or generating automatic answers using an auto-reply feature.[1] Any implementation of this email system must be able to receive and deliver emails, and it may also be able to check arriving emails before delivering it.

Let us add some quantitative information to the specification, resulting in a $\mathscr{K}_{intv}$-WMIO, see Figure 6.2. Every transition label is extended by integer intervals modeling upper and lower bounds of resource units (which could model either energy units, memory or some other costs) required for performing the corresponding actions. For instance, the reception of a new email (*receive?*) must take between one and three resource units, the checking of the email (*check*) is allowed to take up to five resource units.

In this quantitative setting, there is a problem with using strong modal re-

[1] The example is inspired by [97].

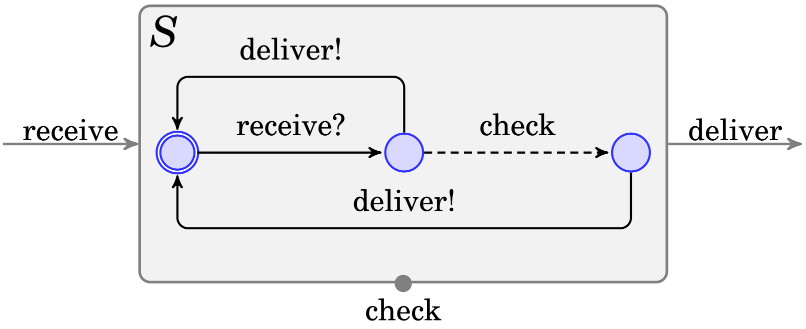


Figure 6.1: A simple email system specified by $S \in \mathbb{MIO}$

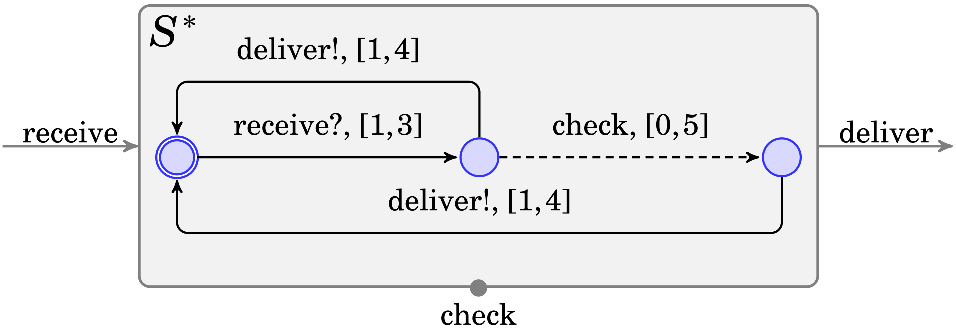


Figure 6.2: A simple email system specified by $S^* \in \mathcal{K}$-$\mathbb{WMIO}$

finement for $\mathcal{K}$-WMIOs as introduced in Section 5.3.1: If one can only decide *whether or not* an implementation refines a specification, then the quantitative aspects are not adquately taken into account in the refinement process. As an example, consider the email system implementations in Figure 6.3. Implementation I_1 does not refine the specification S^*, as there is an error in the discrete structure of actions: after receiving an email, the system can perform one or more checks, however, S^* only permits a single check. Implementations I_2 and I_3 neither refine S^*: in I_2 the action *receive?* consumes one resource unit too much and in I_3 the action *deliver!* after performing a check also consumes one resource unit too much. Implementation I_4 on the other hand is a strong modal refinement of S^*, i.e. $I_4 \leq_s^{\mathcal{K}_{intv}} S^*$.

From an intuitive point of view, however, implementations I_2 and I_3 conform much better to the specification S^* than implementation I_1: there are no discrepancies in the discrete structure, only the weights differ by one. Additionally, observe that the quantitative error in implementation I_3 occurs later than the one in I_2. Hence one may want to say that implementation I_4 is a perfect refinement of the specification S^*, I_3 is almost a (perfect) refinement of S^*, I_2 is a bit more problematic than I_3 because the difference occurs earlier, whereas implementation I_1 is completely unacceptable as a refinement of S^*. A refine-

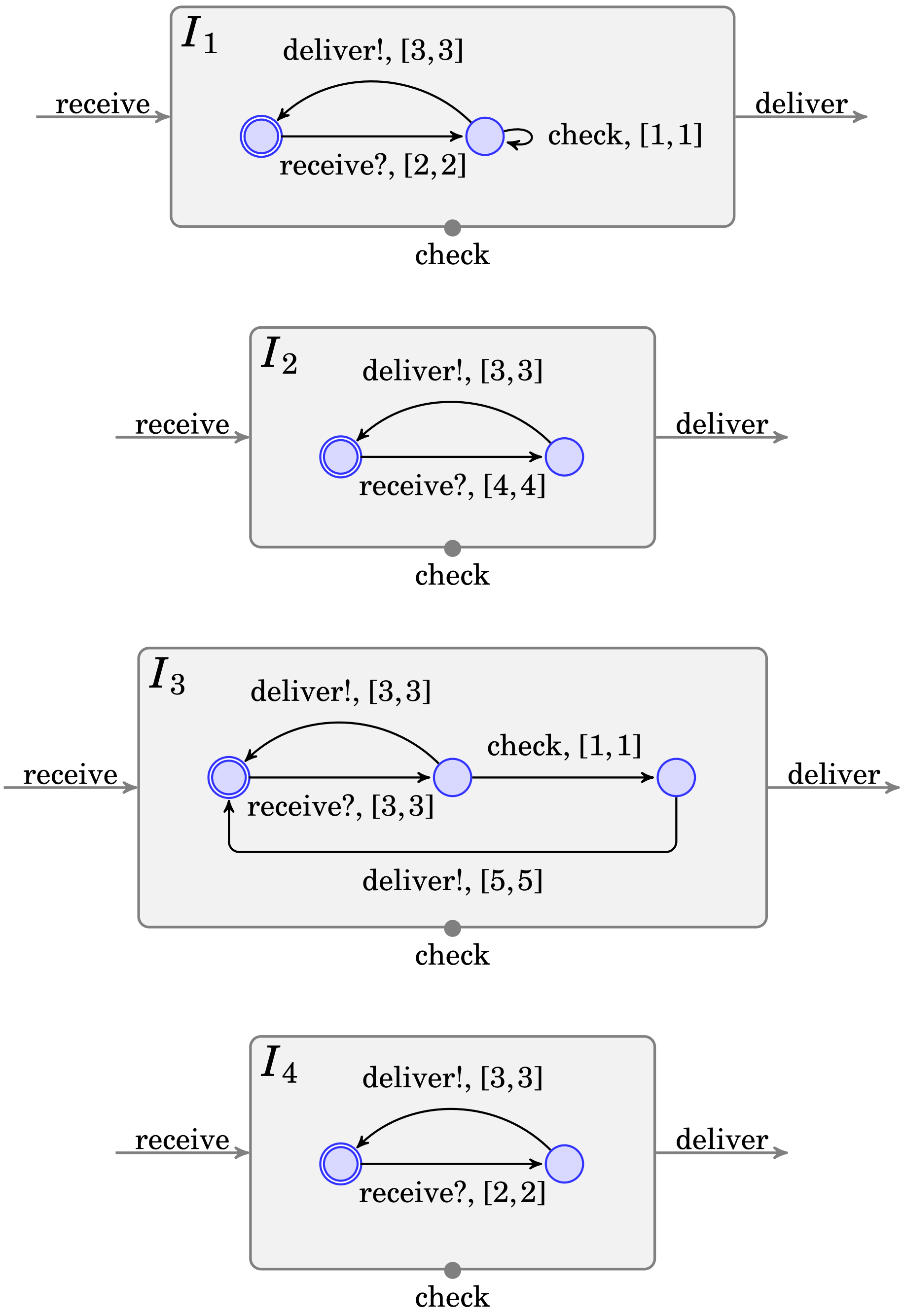


Figure 6.3: Four implementations of the simple email system

ment notion being based on a relation does not allow to make such distinctions between different negative answers. Therefore, we propose a modal refinement distance d_m that allows to measure how “well” an implementation I satisfies its specification S, with

- $d_m(I,S) = 0$ if and only if $I \leq_s^{\mathcal{K}_{intv}} S$, and
- $d_m(I,S) > 0, < \infty$ if and only if $I \downarrow \leq_s S \downarrow$, but $I \not\leq_s^{\mathcal{K}_{intv}} S$,[2]
- $d_m(I,S) = \infty$ if and only if there is an error in the discrete structure, i.e. $I \downarrow \not\leq_s S \downarrow$ and $I \not\leq_s^{\mathcal{K}_{intv}} S$.

Our goal is to show that in the example the following relationships between the individual refinement distances hold:

$$\infty = d_m(I_1,S) > d_m(I_2,S) > d_m(I_3,S) > d_m(I_4,S) = 0$$

To sum up, a refinement notion based on a relation is not flexible enough for quantitative formalisms, as minor and major discrepancies in the refinements cannot be distinguished. As also observed by Alfaro et al. in [60] we need quantitative notions to measure how well a refinement is satisfied. The introduction of such a quantitative notion of refinement, and its consequences for the specification theory, in particular how strong modal refinement and the property of compositionality are affected, are the subject of this chapter.

Depending on the precise application of our quantitative formalism, there are a two choices which one has to make: The first choice concerns the underlying weight structure $\mathcal{K}$. The second choice is the precise definition of quantitative refinement as the way quantitative discrepancies between specifications are measured which depends on whether differences accumulate over time or the interest more lies in the maximal individual differences. In this chapter we stick to the weight structure $\mathcal{K}_{intv}$ of integer intervals, and regarding the second choice we decide for accumulated refinement distances [173]; a more general treatment leaving both weight structure and treatment of quantitative discrepancies abstract can be found in [16].

Outline. In Section 6.1 we introduce modal refinement distances; in particular, we discuss the relationship of a syntactical definition of modal refinement distance and a thorough one, and prove their equivalence if the abstract specification is strongly deterministic. Section 6.2 shows how to lift compositional refinement to a similar result for modal refinement distances. Finally, we mention related works in 6.3 and summarize this chapter in Section 6.4.

[2] Recall that the MIOs $I \downarrow$ and $S \downarrow$ are obtained by removing any transition weights.

6.1 Modal Refinement Distances

Throughout this chapter we use $\mathscr{K}_{intv}$-WMIOs. Recall that

$$\mathscr{K}_{intv} = (K_{intv}, \preceq_{intv}, \oplus_{intv})$$

is the weight structure for integer intervals, with inclusion as weight refinement $\preceq_{intv}$ and integer addition as synchronization operator $\oplus_{intv}$. We also assume in this chapter *finite* $\mathscr{K}_{intv}$-WMIOs, i.e. the set of states and the may-transition relation (and hence the must-transition relation as well) are finite sets.

When given an implementation I and a specification S, both enhanced with quantitative information from $\mathscr{K}_{intv}$, we would like to be able to reason not only whether I correctly implements S, but also to what extent. Therefore we introduce a notion of *modal refinement distance* between $\mathscr{K}_{intv}$-WMIOs.

First we define the distance on weights by

$$d_{\mathscr{K}_{intv}}([x_1, y_1], [x_2, y_2]) = \max(x_2 - x_1, y_1 - y_2, 0).$$

The distance $d_{\mathscr{K}_{intv}}$ measures the largest difference of lower and upper bounds that render the inclusion of the intervals not to hold, e.g. $d_{\mathscr{K}_{intv}}([2,3],[1,5]) = 2$. Note that $d_{\mathscr{K}_{intv}}$ is asymmetric, and that $d_{\mathscr{K}_{intv}}(k,\ell) = 0$ if and only if $k \preceq_{intv} \ell$.

Similar to [59, 60], the modal refinement distance is defined by the unique solution to recursive equations. To ensure convergence of the modal refinement distance from S to T whenever $S\downarrow \leq_s T\downarrow$, we use a *discounting factor* that renders differences at later points in executions converge to zero. More precisely, if $\lambda \in \mathbb{R}$ is a discounting factor with $0 < \lambda < 1$, then the difference in quantitative constraints which occur n steps in the future are discounted by the factor λ^n. This is akin to discounted games [180] where one reasons on behaviours in a discounted manner, giving more importance to differences that happen in the near future, while accumulating the amount by which the specifications deviate at each step. In the rest of the chapter, let $\lambda \in \mathbb{R}$ with $0 < \lambda < 1$ be a fixed discounting factor.

The recursive equations in Definition 6.1.1 below are motivated by viewing strong modal refinement for $\mathscr{K}_{intv}$-WMIOs as a two-player-game [32], very similar to the standard bisimulation games used as a characterization of bisimilarity [168]. In each round of such a *modal refinement game* for $S \leq_s^{\mathscr{K}_{intv}} T$ with $S, T \in \mathscr{K}_{intv}$-$\mathbb{WMIO}$ the players proceed in a state (s,t) with $s \in St_S$, $t \in St_T$ as follows:

1. The attacker chooses either (1.a) a transition $s \overset{\alpha,k}{\dashrightarrow}_S s'$ or (1.b) a transition $t \xrightarrow{\alpha,\ell}_T t'$.

2. The defender responds by choosing a transition under the same action α, either

- in case (1.a) a transition $t \overset{\alpha,\ell}{\dashrightarrow}_T t'$ or
- in case (1.b) a transition $s \xrightarrow{\alpha,k}_S s'$,

in both cases it must hold $k \preceq_{intv} \ell$.

3. The next state becomes (s',t') and the game continues with a new round.

A play then corresponds to a sequence of state pairs formed according to the above rule. A play is finite if one of the players gets stuck by not being able to select a transition. The player who gets stuck looses the play and the other player is the winner. If the play is infinite then the defender is the winner. One can show, cf. [32] for the analogous result for modal transition systems, that a strong modal refinement $S \leq_s^{\mathcal{K}_{intv}} T$ for $\mathcal{K}_{intv}$-WMIOs S and T is equivalent to the existence of a winning strategy for the defender in the game starting in the initial state (s_0,t_0).

Let us now extend modal refinement games for $\mathcal{K}_{intv}$-WMIOs to take into account the accumulated differences of integer intervals according to $d_{\mathcal{K}_{intv}}$. The rules for each round are as above, however, we are not interested in winning strategies anymore but minimizing/maximizing the *payoff* of the game. The payoff v of the game is defined by the accumulated discounted differences of weights of the selected transitions during the play induced by strategies of players such that the attacker is maximizing v and the defender is minimizing v. If the attacker gets stuck in a play then the payoff v is the current value, if the defender gets stuck in a play then the payoff v is ∞. Note that an easy consequence of the definition of the payoff is that if there is a strategy of the attacker to make the defender unable to respond then the payoff is ∞. The payoff is named *modal refinement distance* and formally defined as follows.

Definition 6.1.1 (Modal refinement distance)
Let $S = (St_S, s_0, \Sigma, \dashrightarrow_S, \longrightarrow_S)$ and $T = (St_T, t_0, \Sigma, \dashrightarrow_T, \longrightarrow_T)$ be two finite $\mathcal{K}_{intv}$-WMIOs. Let $\ll$ be the partial order on the function space $\{f \mid f : St_S \times St_T \to \mathbb{R}_{\geq 0} \cup \{\infty\}\}$ defined by $f \ll f'$ iff $f(s',t') \leq f'(s',t')$ for all $s' \in St_S$ and all $t' \in St_T$.

The modal refinement distance $d_m : St_S \times St_T \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ *from the state $s \in St_S$ to the state $t \in St_T$ is the unique least fixed point, according the partial order $\ll$, of the recursive equation*

$$d_m(s,t) = \max \begin{cases} \max\limits_{s \overset{\alpha,k}{\dashrightarrow}_S s'} \; \min\limits_{t \overset{\alpha,\ell}{\dashrightarrow}_T t'} \; d_{\mathcal{K}_{intv}}(k,\ell) + \lambda d_m(s',t'), \\ \max\limits_{t \xrightarrow{\alpha,\ell}_T t'} \; \min\limits_{s \xrightarrow{\alpha,k}_S s'} \; d_{\mathcal{K}_{intv}}(k,\ell) + \lambda d_m(s',t'). \end{cases}$$

The distance from S to T is defined by $d_m(S,T) \triangleq d_m(s_0,t_0)$. We write $S \leq_s^{\varepsilon} T$ if $d_m(S_1,S_2) \leq \varepsilon$.

Note that $\max_{\emptyset} \triangleq 0$ and $\min_{\emptyset} \triangleq \infty$. To justify the definition of $d_m(s,t)$, we need to show that the recursive equations in the definition above indeed have a unique least fixed point.

Lemma 6.1.2
The recursive equation in Definition 6.1.1 has a unique least fixed point according to the partial ordering $\ll$.

Proof. The proof is an adaptation of Lemma 3 in [122]. We first assume that S and T are non-blocking such that in every state there is a must-transition enabled for each action. Then we know for sure that the distance is in $\mathbb{R}_{\geq 0}$. Now, the idea is to use the Banach fixed point theorem [3]:

> If (X,d) is a complete metric space, and $F : X \to X$ defines a contraction, i.e. there exists a constant $c \in (0,1)$ such that
>
> $$d(F(x),F(y)) \leq c \cdot d(x,y) \text{ for all } x,y \in X,$$
>
> then F has a unique fixed point.

Let us define the complete metric space (X,d) and the function F such that we can apply the Banach fixed point theorem. Since we only consider finite $\mathcal{K}_{intv}$-WMIOs we can assume that S and T have p and q states, respectively. The set $\mathbb{R}_{\geq 0}^{p \times q}$ of $(p \times q)$-matrices with entries in $\mathbb{R}_{\geq 0}$ is a complete metric space with the metric

$$\rho(M,N) \triangleq \max_{1 \leq i \leq p, 1 \leq j \leq q} |M_{i,j} - N_{i,j}|$$

where $M_{i,j}$ and $N_{i,j}$ denote the (i,j)-th entry of the matrix M and N, respectively. Define $F : \mathbb{R}_{\geq 0}^{p \times q} \to \mathbb{R}_{\geq 0}^{p \times q}$ by

$$F(M)_{i,j} = \max \begin{cases} \max\limits_{s \overset{\alpha,k}{\dashrightarrow}_S s'} \ \min\limits_{t \overset{\alpha,\ell}{\dashrightarrow}_T t'} \ d_{\mathcal{K}_{intv}}(k,\ell) + \lambda \cdot M_{k,\ell}, \\ \max\limits_{t \xrightarrow{\alpha,\ell}_T t'} \ \min\limits_{s \xrightarrow{\alpha,k}_S s'} \ d_{\mathcal{K}_{intv}}(k,\ell) + \lambda \cdot M_{k,\ell}. \end{cases}$$

Following [122] we partition $\mathbb{R}_{\geq 0}^{p \times q}$ into finitely many closed polyhedral regions such that for two matrices M and N in a same region the recursive equation get resolved to the same transitions, i.e. there are mappings

$$e,f : \{1,\ldots,p\} \times \{1,\ldots,q\} \to K_{intv}, \quad g : \{1,\ldots,p\} \times \{1,\ldots,q\} \to \{1,\ldots,p\} \times \{1,\ldots,q\}$$

such that

$$F(M)_{i,j} = d_{\mathcal{K}_{intv}}(e(i,j), f(i,j)) + \lambda \cdot M_{g(i,j)}$$

for all M in the same region. If M and N are in a common region then

$$\begin{aligned}\rho(F(M),F(N)) &= \max_{1\le i\le p, 1\le j\le q} |F(M)_{i,j} - F(N)_{i,j}| \\ &= \lambda\left(\max_{1\le i\le p, 1\le j\le q} |M_{g(i,j)} - N_{g(i,j)}|\right) \\ &\le \lambda\left(\max_{1\le i\le p, 1\le j\le q} |M_{i,j} - N_{i,j}|\right) \\ &= \lambda\rho(M,N).\end{aligned}$$

If M and N are in different regions then we consider the straight line segment between M and N with finitely many intersection points $M = Z_0, \ldots, Z_r = N$ with other regions. Then we have

$$\begin{aligned}\rho(F(M),F(N)) &\le \rho(F(Z_0),F(Z_1)) + \ldots + \rho(F(Z_{r-1}),F(Z_r)) \\ &\le \lambda(\rho(Z_0,Z_1) + \ldots + \rho(Z_{r-1},Z_r)) \\ &\le \lambda\rho(M,N).\end{aligned}$$

The last equality holds because ρ is a metric and all $Z_0,\ldots,Z_r$ are on a straight line. Thus, we have shown that for non-blocking S and T we have a unique fixed point d_m which yields a distance in $\mathbb{R}_{\ge 0}$ for all state pairs in $St_S \times St_T$. Now for the general case for arbitrary (but still finite) $\mathcal{K}_{intv}$-WMIOs S and T, F is a function from $(\mathbb{R}_{\ge 0} \cup \{\infty\})^{p\times q}$ to $(\mathbb{R}_{\ge 0} \cup \{\infty\})^{p\times q}$ with additional fixed point $M \in (\mathbb{R}_{\ge 0} \cup \{\infty\})^{p\times q}$ defined by $M_{i,j} = \infty$ for all $1 \le i \le p$, $1 \le j \le q$. Hence the recursive equation has at most *two* fixed points. We can write the recursive equation from the definition as

$$\begin{bmatrix} d_m(s_1,t_1) & \cdots & d_m(s_1,t_q) \\ d_m(s_2,t_1) & \cdots & d_m(s_2,t_q) \\ \vdots & \ddots & \vdots \\ d_m(s_p,t_1) & \cdots & d_m(s_p,t_q) \end{bmatrix} = F \begin{bmatrix} d_m(s_1,t_1) & \cdots & d_m(s_1,t_q) \\ d_m(s_2,t_1) & \cdots & d_m(s_2,t_q) \\ \vdots & \ddots & \vdots \\ d_m(s_p,t_1) & \cdots & d_m(s_p,t_q) \end{bmatrix}$$

and hence there exists a unique least fixed point. □

Except for the symmetrizing max operation, this is precisely the *accumulating branching distance* which is introduced in [173]; see also [172] for a thorough introduction to branching distances as we use them here.

Example 6.1.3
Consider the two implementations S and T in Figure 6.4 with a single internal action α and with discounting factor $\lambda = 0.9$. The below equations are already simplified by removing all expressions that evaluate to ∞.

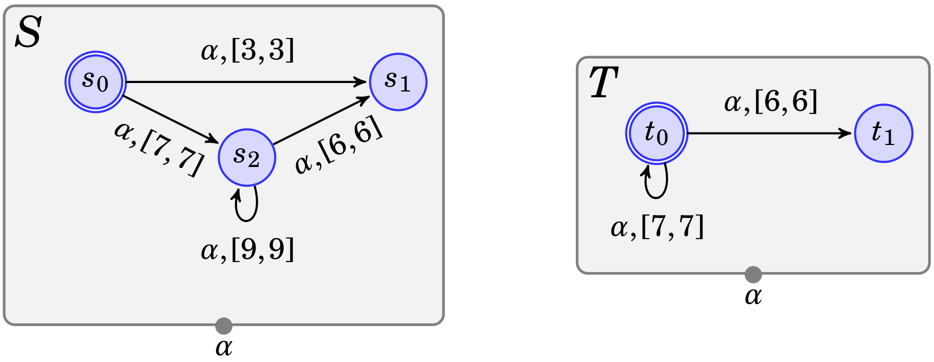


Figure 6.4: $\mathcal{K}_{intv}$-WMIOs S and T, $d_m(S,T) = 18$

$$\begin{aligned}
&d_m(s_0,t_0) = \max\{3+\lambda d_m(s_1,t_1),\ \lambda d_m(s_2,t_0)\}\\
&d_m(s_0,t_1) = \infty\\
&d_m(s_1,t_0) = \infty\\
&d_m(s_1,t_1) = 0\\
&d_m(s_2,t_0) = \max\{2+\lambda d_m(s_2,t_0),\ \lambda d_m(s_1,t_1)\}\\
&d_m(s_2,t_1) = \infty
\end{aligned}$$

What remains to be done is to compute the least fixed point of the equation

$$d_m(s_2,t_0) = \max\{2+\lambda d_m(s_2,t_0), 0\}$$

which is $d_m(k_1,i_2) = 20$. *Hence* $d_m(s_0,t_0) = \max\{3, \lambda \cdot 20\} = 18$.

Note that the interpretation of the distance between two specifications depends entirely on the application one has in mind; but in any case, the distance from S to T is 0 if and only if $S \leq_s^{\mathcal{K}_{intv}} T$.

Example 6.1.4
Let us analyze the modal refinement distances for the $\mathcal{K}_{intv}$*-WMIOs that were given in the introduction in Figures 6.2 and 6.3. We can easily compute the modal refinement distances, and since they are defined as discounted accumulating distances of the respective labels, we get finite distances in three of four cases.*

$$\begin{aligned}
&d_m(I_1,S) = \infty\\
&d_m(I_2,S) \approx 5,26\\
&d_m(I_3,S) \approx 2,98\\
&d_m(I_4,S) = 0
\end{aligned}$$

Indeed, we have verified our descending chain of distances that we have claimed in the beginning of this chapter:

$$\infty = d_m(I_1,S) > d_m(I_2,S) > d_m(I_3,S) > d(I_4,S) = 0$$

Next we define a thorough distance of specifications, by a similar construction as the *Hausdorff distance* between closed subsets of a metric space, see e.g. [147]. Crucially however, our thorough distance is missing the symmetrizing max operation of Hausdorff distance, hence it is *asymmetric*.

Definition 6.1.5 (Thorough refinement distance)
The thorough refinement distance *between $\mathcal{K}_{intv}$-WMIOs S and T is defined as*

$$d_t(S,T) \triangleq \sup_{I \in [\![S]\!]_s^{\mathcal{K}_{intv}}} \inf_{J \in [\![T]\!]_s^{\mathcal{K}_{intv}}} d_m(I,J).$$

Indeed this permits us to measure how well refinement is satisfied; intuitively, if two specifications have thorough distance $\varepsilon \geq 0$, then any implementation of the first specification can be matched by an implementation of the second up to ε. Also observe the special case where S is an implementation: then $d_t(S,T) = \inf_{J \in [\![T]\!]_s^{\mathcal{K}_{intv}}} d_m(S,J)$, which measures how close S is to satisfy the specification T.

In [14] we have shown that similarly to strong modal refinement, computation of thorough refinement distance is EXPTIME-hard [31], whereas the modal refinement distance is computable in NP ∩ CO-NP.

One might ask whether the thorough refinement distance is different to the modal refinement distance and how they relate in general. We show in the rest of this section that the modal refinement distance is in general an over-approximation of the thorough refinement distance, and if the abstract $\mathcal{K}_{intv}$-WMIO is deterministic, the modal refinement distance and thorough refinement distance coincide.

There is a powerful proof technique via families of relations introduced for branching distances between implementations in [173] that we here extend to modal refinement distance.

Definition 6.1.6
We define a modal refinement family *for two finite $\mathcal{K}_{intv}$-WMIOs S and T as an $\mathbb{R}_{\geq 0}$-indexed family of relations $R = \{R_\varepsilon \subseteq St_S \times St_T \mid \varepsilon \geq 0\}$ such that for any $\varepsilon \geq 0$ and any $(s,t) \in R_\varepsilon$, for all $\alpha \in \bigcup \Sigma$,*

- *for all $k \in K$, $s' \in St_S$, if $s \overset{\alpha,k}{\dashrightarrow}_S s'$, then there exists $t \overset{\alpha,\ell}{\dashrightarrow}_T t'$ for some $\ell \in K$, $t' \in T$, such that $d_{\mathcal{K}_{intv}}(k,\ell) \leq \varepsilon$ and $(s',t') \in R_{\varepsilon'}$ for some $\varepsilon' \leq \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell)\big)$,*
- *for all $\ell \in K$, $t' \in St_T$, if $t \xrightarrow{\alpha,\ell}_T t'$, then there exists $s \xrightarrow{\alpha,k}_S s'$ for some $k \in K$, $s' \in S$, such that $d_{\mathcal{K}_{intv}}(k,\ell) \leq \varepsilon$ and $(s',t') \in R_{\varepsilon'}$ for some $\varepsilon' \leq \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell)\big)$,*

and such that R is downward closed *in the sense that for any set $E \subseteq \mathbb{R}_{\geq 0}$, if $(s,t) \in R_\varepsilon$ for all $\varepsilon \in E$, then also $(s,t) \in R_{\inf E}$.*

Following the proof strategy developed in [173] for implementations, we can show the following characterization of modal refinement distance by modal refinement families:

Proposition 6.1.7

Let S and T be two $\mathcal{K}_{intv}$-WMIOs with the same action signature Σ. Then the following are equivalent:

1. $S \leq_m^\varepsilon T$.
2. *There is a modal refinement family R for S and T with $(s_0,t_0) \in R_\varepsilon \in R$.*

Proof. First, assume that $S \leq_m^\varepsilon T$, i.e. $d_m(s_0,t_0) \leq \varepsilon$, and define a relation family $R = \{R_\delta \mid \delta \geq 0\}$ by $R_\delta = \{(s,t) \in St_S \times St_T \mid d_m(s,t) \leq \delta\}$ for all $\delta \geq 0$, then $(s_0,t_0) \in R_\varepsilon$ holds by assumption. We show that R is a modal refinement family. Downward closedness of R follows from the fact that there are only a finite number of choices of transitions for each pair (s,t).

Let $(s,t) \in R_\delta$ for some $\delta \geq 0$, then by definition we know that $d_m(s,t) \leq \delta$. Assume

$$s \overset{\alpha,k}{\dashrightarrow}_S s'.$$

From $d_m(s,t) \leq \delta$ we can infer that

$$\min_{t \overset{\alpha,\ell}{\dashrightarrow}_T t'} d_{\mathcal{K}_{intv}}(k,\ell) + \lambda d_m(s',t') \leq \delta.$$

Hence there exists a may-transition $t \overset{\alpha,\ell}{\dashrightarrow}_T t'$ such that $d_{\mathcal{K}_{intv}}(k,\ell) \leq \delta$ and

$$d_m(s',t') \leq \lambda^{-1}(\delta - d_{\mathcal{K}_{intv}}(k,\ell)).$$

The latter implies that $(s',t') \in R_{\delta'}$ for some $\delta' \leq \lambda^{-1}(\delta - d_{\mathcal{K}_{intv}}(k,\ell))$ which was to be shown. The argument for the other assertion for must-transitions is symmetric. This proves that there is a modal refinement family R such that $(s_0,t_0) \in R_\varepsilon \in R$.

For the reverse direction, assume that $(s_0,t_0) \in R_\varepsilon \in R$ for some modal refinement family $R = \{R_\varepsilon \mid \varepsilon \geq 0\}$. We prove that $(s,t) \in R_\delta$, for some $\delta \geq 0$, implies $d_m(s,t) \leq \delta$. The claim $S \leq_m^\varepsilon T$ then follows from the assumption $(s_0,t_0) \in R_\varepsilon$.

To this end, observe that the space of functions $\Delta = [St_S \times St_T \to \mathbb{R}_{\geq 0} \cup \{\infty\}]$ forms a complete lattice, when the partial order $\ll$ is defined such that for $f, f' \in \Delta$, $f \ll f'$ iff $f(s,t) \leq f'(s,t)$ for all $s \in St_S$, $t \in St_T$. Moreover, since max, min, addition and multiplication by λ are monotone, the function D defined for all $f \in \Delta$ by

$$D(f) = \max \begin{cases} \max\limits_{s \overset{\alpha,k}{\dashrightarrow}_S s'} \min\limits_{t \overset{\alpha,\ell}{\dashrightarrow}_T t'} d_{\mathcal{K}_{intv}}(k,\ell) + \lambda f(s',t'), \\ \max\limits_{t \xrightarrow{\alpha,\ell}_T t'} \min\limits_{s \xrightarrow{\alpha,k}_S s'} d_{\mathcal{K}_{intv}}(k,\ell) + \lambda f(s',t') \end{cases}$$

is an order-preserving function on Δ, hence by Tarski's fixed point theorem [170], D has a least fixed point. Now let us define $h(s,t) = \inf\{\delta \mid (s,t) \in R_\delta \in R\}$, and since R is downward closed, we have that $(s,t) \in R_{h(s,t)}$. By showing that h is a pre-fixed point of D, i.e. that $D(h) \ll h$, we get that $(s,t) \in R_\delta$ implies that $d_m(s,t) \le \delta$, since $h(s,t) \le \delta$ and $d_m(s,t) \le h(s,t)$.

Since $(s,t) \in R_{h(s,t)}$, every

$$s \overset{\alpha,k}{\dashrightarrow}_S s'' \text{ can be matched by some } t \overset{\alpha,\ell}{\dashrightarrow}_T t''$$

such that $d_{\mathcal{K}_{intv}}(k,\ell) + \lambda\delta' \le h(s,t)$ for some $\delta' \ge 0$ and $(s'',t'') \in R_{\delta'}$, implying $h(s'',t'') \le \delta'$, but then also $d_{\mathcal{K}_{intv}}(k,\ell) + \lambda h(s'',t'') \le h(s,t)$. Similarly, every

$$t \xrightarrow{\alpha,\ell}_T t'' \text{ can be matched by some } s \xrightarrow{\alpha,k}_S s''$$

such that $d_{\mathcal{K}_{intv}}(k,\ell) + \lambda h(s'',t'') \le h(s,t)$. Hence we have $D(h) \ll h$ which was to be shown. □

The next theorems show that the modal refinement distance indeed overapproximates the thorough refinement distance, and that it is exact for deterministic $\mathcal{K}_{intv}$-WMIOs. Note that nothing general can be said about the precision of the overapproximation in the non-deterministic case; as an example we can again observe the two specifications in Figure 5.4, Chapter 5, for which $d_t(S,T) = 0$ but $d_m(S,T) = 1$.

Theorem 6.1.8
Let S and T be two finite $\mathcal{K}_{intv}$-WMIOs with the same signature Σ. Then we have $d_t(S,T) \le d_m(S,T)$.

Proof. If $d_m(S,T) = \infty$, we have nothing to prove. Otherwise, let $R = \{R_\varepsilon \subseteq St_S \times St_T \mid \varepsilon \ge 0\}$ be a modal refinement family which witnesses $d_m(S,T)$, i.e. such that $(s_0,t_0) \in R_{d_m(S,T)}$, and let $I \in [\![S]\!]_s^{\mathcal{K}_{intv}}$. We have to expose $J \in [\![T]\!]_s^{\mathcal{K}_{intv}}$ for which $d_m(I,J) \le d_m(S,T)$.

Let $Q \subseteq St_I \times St_S$ be a refinement relation witnessing $I \le_s^{\mathcal{K}_{intv}} S$, define $R'_\varepsilon = Q \circ R_\varepsilon \subseteq St_I \times St_T$ for all $\varepsilon \ge 0$, and let $R' = \{R'_\varepsilon \mid \varepsilon \ge 0\}$. The states of

$$J = (St_J, j_0, \Sigma, \dashrightarrow_J, \longrightarrow_J)$$

are St_T with $j_0 = t_0$, and the must-transition relation we define as follows (the may-transition relation coincides with the must-transition relation):

For any

$$i \xrightarrow{\alpha,k'}_I i'$$

and any $t \in St_T$ for which $(i,t) \in R'_\varepsilon \in R'$ for some ε, we have

$$t \overset{\alpha,\ell}{\dashrightarrow}_T t'$$

with $d_{\mathcal{K}_{intv}}(k',\ell) \le \varepsilon$ and $(i',t') \in R'_{\varepsilon'} \in R'$ for some $\varepsilon' \le \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k',\ell)\big)$. Assume $k' = [m,m]$ and $\ell = [x,y]$. Let

$$n = \begin{cases} x & \text{if } m < x, \\ m & \text{if } x \le m \le y, \\ y & \text{if } y < m \end{cases} \tag{1}$$

and $k'' = [n,n]$, and put $t \xrightarrow{\alpha,k''}_J t'$ in J. Note that

$$d_{\mathcal{K}_{intv}}(k',k'') = d_{\mathcal{K}_{intv}}(k',\ell). \tag{2}$$

Similarly, for any

$$t \xrightarrow{\alpha,\ell}_T t'$$

and any $i \in St_I$ with $(i,t) \in R'_\varepsilon \in R'$ for some ε, we have

$$i \xrightarrow{\alpha,k'}_I i'$$

with $d_{\mathcal{K}_{intv}}(k',\ell) \le \varepsilon$ and $(i',t') \in R'_{\varepsilon'} \in R'$ for some $\varepsilon' \le \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k',\ell)\big)$. Write $k' = [m,m]$ and $\ell = [x,y]$, define n as in (2) above and $k'' = [n,n]$, and put

$$t \xrightarrow{\alpha,k''}_J t'$$

in J. It is straightforward to prove that the identity relation $\text{id} = \{(t,t) \mid t \in St_T\}$ witnesses $J \le_s^{\mathcal{K}_{intv}} T$; this essentially follows from the construction of J according to the above rules.

We also want to show that the family R' is a witness for $d_m(I,J) \le d_m(S,T)$. We have $(i_0,t_0) \in R'_{d_m(S,T)} = R_1 \circ R_{d_m(S,T)}$, so let $(i,t) \in R'_\varepsilon \in R'$ for some $\varepsilon \ge 0$. For any

$$i \xrightarrow{\alpha,k'}_I i'$$

we have

$$t \overset{\alpha,\ell}{\dashrightarrow}_T t' \text{ and } t \overset{\alpha,k''}{\dashrightarrow}_J t'$$

by the first part of our construction above, with $d_{\mathcal{K}_{intv}}(k',k'') = d_{\mathcal{K}_{intv}}(k',\ell) \le \varepsilon$ because of (2), and also $(i',t') \in R'_{\varepsilon'} \in R'$ for some $\varepsilon' \le \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k',\ell)\big)$. For any

$$t \xrightarrow{\alpha,k''}_J t'$$

we must have used one of the constructions above to introduce this transition, and both give us

$$i \xrightarrow{\alpha,k'}_I i'$$

with $d_{\mathcal{K}_{intv}}(k',k'') \le \varepsilon$ and $(i',t') \in R'_{\varepsilon'} \in R'$ for some $\varepsilon' \le \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k',\ell)\big)$. □

The fact that modal refinement only equals thorough refinement for deterministic specifications is well-known from the previous chapters. Observe that a deterministic $\mathcal{K}_{intv}$-WMIO allows at most one transition under each discrete action in each state since every two intervals $[x_1, y_1]$ and $[x_2, y_2]$ are unifiable by $[\min(x_1, x_2), \max(y_1, y_2)]$; cf. Definition 5.3.5 in Chapter 5.

Theorem 6.1.9
Let S and T be two finite $\mathcal{K}_{intv}$-WMIOs. If T is deterministic then $d_t(S,T) = d_m(S,T)$.

Proof. If $d_t(S,T) = \infty$, then we are done by Theorem 6.1.8. Otherwise, let $R = \{R_\varepsilon \mid \varepsilon \geq 0\}$ be the smallest downward closed relation family for which

- $(s_0, t_0) \in R_{d_t(S,T)}$ and
- whenever we have $(s,t) \in R_\varepsilon \in R$, $s \overset{\alpha,k}{\dashrightarrow}_S s'$, and $t \overset{\alpha,\ell}{\dashrightarrow}_T t'$, then $(s',t') \in R_{\lambda^{-1}(\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell))}$.

We show below that this definition makes sense (also that $\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell) \geq 0$ in all cases), and that R is a modal refinement family. We will use the convenient notation (s,S) for the $\mathcal{K}_{intv}$-WMIO S with initial state s_0 replaced by s, similarly for (t,T).

We first show inductively that for any pair of states $(s,t) \in R_\varepsilon \in R$ we have $d_t\big((s,S),(t,T)\big) \leq \varepsilon$. This is obviously the case for $s = s_0$ and $t = t_0$, so assume now that $(s,t) \in R_\varepsilon \in R$ is such that $d_t\big((s,S),(t,T)\big) \leq \varepsilon$ and let

$$s \overset{\alpha,k}{\dashrightarrow}_S s' \text{ and } t \overset{\alpha,\ell}{\dashrightarrow}_T t'.$$

Let $I' \in [\![(s',S)]\!]_s^{\mathcal{K}_{intv}}$ and $[x,x] \in [\![k]\!]$.

There is an implementation $I \in [\![(s,S)]\!]_s^{\mathcal{K}_{intv}}$ for which

$$i_0 \xrightarrow{\alpha,[x,x]}_I i_1$$

and such that $(i_1, I) \leq_s^{\mathcal{K}_{intv}} I'$. Now

$$d_t\big((i_1,I),(t,T)\big) \leq d_t\big((i_1,I),(s,S)\big) + d_t\big((s,S),(t,T)\big) \leq \varepsilon,$$

hence we must have

$$t \overset{\alpha,\ell'}{\dashrightarrow}_T t''$$

with $d_{\mathcal{K}_{intv}}([x,x],\ell') \leq \varepsilon$. But then by determinism of T, $\ell = \ell'$ and $t' = t''$. The above considerations hold for any $[x,x] \in [\![k]\!]$, hence $d_{\mathcal{K}_{intv}}(k,\ell) \leq \varepsilon$. Thus $\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell) \geq 0$, and the definition of R above is justified. Now let $[y,y] \in [\![\ell]\!]$

such that $d_{\mathcal{K}_{intv}}([x,x],[y,y]) = d_{\mathcal{K}_{intv}}([x,x],\ell)$, then there is an implementation $J \in \llbracket (t,T) \rrbracket_s^{\mathcal{K}_{intv}}$ for which

$$j_0 \xrightarrow{\alpha,[y,y]}_J j_1$$

and

$$\begin{aligned} d_m(I,J) &\leq \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}([x,x],[y,y])\big) \\ &= \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell)\big), \end{aligned}$$

which, as $I \in \llbracket (s,S) \rrbracket_s^{\mathcal{K}_{intv}}$ was chosen arbitrarily, entails $d_t\big((s,S),(t,T)\big) \leq \lambda^{-1}\big(\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell)\big)$.

We are ready to show that R is a modal refinement family. Let $(s,t) \in R_\varepsilon \in R$ for some ε, and assume

$$s \overset{\alpha,k}{\dashrightarrow}_S s'.$$

Let $[x,x] \in \llbracket k \rrbracket$, then there is an implementation $I \in \llbracket (s,S) \rrbracket_s^{\mathcal{K}_{intv}}$ with a transition

$$i_0 \xrightarrow{\alpha,[x,x]}_I i_1.$$

Now $d_t\big(I,(t,T)\big) \leq \varepsilon$ by the first part of the proof, hence we have a transition

$$t \overset{\alpha,\ell}{\dashrightarrow}_T t'$$

with $d_{\mathcal{K}_{intv}}([x,x],\ell) \leq \varepsilon$. Also for any other $[x',x'] \in \llbracket k \rrbracket$ we have a transition

$$t \overset{\alpha,\ell'}{\dashrightarrow}_T t''$$

with $d_{\mathcal{K}_{intv}}([x',x'],\ell') \leq \varepsilon$, hence by determinism of T, $\ell = \ell'$ and $t' = t''$. It follows that there is a unique transition

$$t \overset{\alpha,\ell}{\dashrightarrow}_T t'$$

and as $d_{\mathcal{K}_{intv}}([x,x],\ell) \leq \varepsilon$ for all $[x,x] \in \llbracket k \rrbracket$, we have $d_{\mathcal{K}_{intv}}(k,\ell) \leq \varepsilon$, and $(s',t') \in R_{\lambda^{-1}(\varepsilon - d_{\mathcal{K}_{intv}}(k,\ell))}$ by definition. The other direction, for must-transitions, can be shown similarly and the proof is omitted here. □

6.2 Compositionality in the Context of Modal Refinement Distances

In this section we show that compositionality can naturally be extended to the context of modal refinement distances. As the main result we show that the refinement distance between composed systems can be bounded by the distances between their constituent systems.

Recall that the weight synchronization operator $\oplus_{intv}$ of the weight structure $\mathcal{K}_{intv}$ has already been defined in the previous chapter, see Example 5.1.4. The weight synchronization operator $\oplus_{intv}$ for modal synchronous composition of $\mathcal{K}_{intv}$-WMIOs is defined by

$$[x_1, y_1] \oplus_{intv} [x_2, y_2] = [x_1 + x_2, y_1 + y_2].$$

After a technical lemma, the next theorem shows that modal synchronous composition is well-behaved with respect to modal refinement distance in the sense that the distance between the composed systems is bounded by the distances of the individual systems. Note also the special case in the theorem of $S_1 \leq_s^{\mathcal{K}_{intv}} S_2$ and $S_3 \leq_s^{\mathcal{K}_{intv}} S_4$ implying $S_1 \otimes^{\mathcal{K}_{intv}} S_3 \leq_s^{\mathcal{K}_{intv}} S_2 \otimes^{\mathcal{K}_{intv}} S_4$.

Lemma 6.2.1
For $k_1, k_2, k_3, k_4 \in \mathcal{K}_{intv}$, we have

$$d_{\mathcal{K}_{intv}}(k_1 \oplus_{intv} k_3, k_2 \oplus_{intv} k_4) \leq d_{\mathcal{K}_{intv}}(k_1, k_2) + d_{\mathcal{K}_{intv}}(k_3, k_4).$$

Proof. Let $k_i = \big(\alpha, [x_i, y_i]\big)$ for all i. We have

$$\begin{aligned} d_{\mathcal{K}_{intv}}(k_1, k_2) + d_{\mathcal{K}_{intv}}(k_3, k_4) &= \max(x_2 - x_1, y_1 - y_2, 0) + \max(x_4 - x_3, y_3 - y_4, 0) \\ &\geq \max\big((x_2 - x_1) + (x_4 - x_3), (y_1 - y_2) + (y_3 - y_4), 0\big) \\ &= \max\big((x_2 + x_4) - (x_1 + x_3), (y_1 + y_3) - (y_2 + y_4), 0\big) \\ &= d_{\mathcal{K}_{intv}}(k_1 \oplus_{intv} k_3, k_2 \oplus_{intv} k_4). \end{aligned}$$

Theorem 6.2.2
Let $S_i = (St_i, s_{0,i}, \Sigma_i, \dashrightarrow_i, \longrightarrow_i)$, $i \in \{1, \dots, 4\}$, be finite $\mathcal{K}_{intv}$-WMIOs. Then

$$d_m\big(S_1 \otimes^{\mathcal{K}_{intv}} S_3, S_2 \otimes^{\mathcal{K}_{intv}} S_4\big) \leq d_m(S_1, S_2) + d_m(S_3, S_4).$$

Proof. If $d_m(S_1, S_2) = \infty$ or $d_m(S_3, S_4) = \infty$, we have nothing to prove. Otherwise, let $R^1 = \{R^1_\varepsilon \subseteq St_1 \times St_2 \mid \varepsilon \geq 0\}$, $R^2 = \{R^2_\varepsilon \subseteq St_3 \times St_4 \mid \varepsilon \geq 0\}$ be witnesses for $d_m(S_1, S_2)$ and $d_m(S_3, S_4)$, respectively; hence $(s_{0,1}, s_{0,2}) \in R^1_{d_m(S_1, S_2)} \in R^1$ and $(s_{0,3}, s_{0,4}) \in R^2_{d_m(S_3, S_4)} \in R^2$. Define

$$\begin{aligned} R_\varepsilon = \big\{\big((s_1, s_3), (s_2, s_4)\big) \in St_1 \times St_3 \times St_2 \times St_4 \;\big|\; & \\ (s_1, s_2) \in R^1_{\varepsilon_1} \in R^1, (s_3, s_4) \in R^2_{\varepsilon_2} \in R^2, \varepsilon_1 + \varepsilon_2 \leq \varepsilon\big\} & \end{aligned}$$

for all $\varepsilon \geq 0$ and let $R = \{R_\varepsilon \mid \varepsilon \geq 0\}$. We show that R witnesses

$$d_m\big(S_1 \otimes^{\mathcal{K}_{intv}} S_3, S_2 \otimes^{\mathcal{K}_{intv}} S_4\big) \leq d_m(S_1, S_2) + d_m(S_3, S_4).$$

We have $\big((s_{0,1},s_{0,3}),(s_{0,2},s_{0,4})\big)\in R_{d_m(S_1,S_2)+d_m(S_3,S_4)}\in R$. Now let

$$\big((s_1,s_3),(s_2,s_4)\big)\in R_\varepsilon\in R$$

for some ε, then $(s_1,s_2)\in R^1_{\varepsilon_1}\in R^1$ and $(s_3,s_4)\in R^2_{\varepsilon_2}\in R^2$ for some $\varepsilon_1+\varepsilon_2\le\varepsilon$.

Assume

$$(s_1,s_3)\xrightarrow{\alpha,k_1\oplus_{intv}k_3}(s_1',s_3'),$$

then $s_1\xrightarrow{\alpha,k_1}_1 s_1'$ and $s_3\xrightarrow{\alpha,k_3}_3 s_3'$. By $(s_1,s_2)\in R^1_{\varepsilon_1}\in R^1$, we have

$$s_2\xrightarrow{\alpha,k_2}_2 s_2'$$

with $d_{\mathcal{K}_{intv}}(k_1,k_2)\le\varepsilon_1$ and $(s_1',s_2')\in R^1_{\varepsilon_1'}\in R^1$ for some $\varepsilon_1'\le\lambda^{-1}\big(\varepsilon_1-d_{\mathcal{K}_{intv}}(k_1,k_2)\big)$; similarly,

$$s_4\xrightarrow{\alpha,k_4}_4 s_4'$$

with $d_{\mathcal{K}_{intv}}(k_3,k_4)\le\varepsilon_2$ and $(s_3',s_4')\in R^2_{\varepsilon_2'}\in R^2$ for some $\varepsilon_2'\le\lambda^{-1}\big(\varepsilon_2-d_{\mathcal{K}_{intv}}(k_3,k_4)\big)$. Let $\varepsilon'=\varepsilon_1'+\varepsilon_2'$, then

$$\begin{aligned}\varepsilon'&\le\lambda^{-1}\big(\varepsilon_1+\varepsilon_2-(d_{\mathcal{K}_{intv}}(k_1,k_2)+d_{\mathcal{K}_{intv}}(k_3,k_4))\big)\\&\le\lambda^{-1}\big(\varepsilon-d_{\mathcal{K}_{intv}}(k_1\oplus_{intv}k_3,k_2\oplus_{intv}k_4)\big)\end{aligned}$$

by Lemma 6.2.1. We have

$$(s_2,s_4)\xrightarrow{\alpha,k_2\oplus_{intv}k_4}(s_2',s_4'),$$

$d_{\mathcal{K}_{intv}}(k_1\oplus_{intv}k_3,k_2\oplus_{intv}k_4)\le\varepsilon_1+\varepsilon_2\le\varepsilon$ again by Lemma 6.2.1, and

$$\big((s_1',s_3'),(s_2',s_4')\big)\in R_{\varepsilon'}\in R.$$

The reverse direction, starting with a transition

$$(s_2,s_4)\xrightarrow{\alpha,k_2\oplus_{intv}k_4}(t_2,t_4)$$

is similar. □

6.3 Related Work

Our approach to quantitative analysis of modal transition system reuses existing notions and results from literature about simulation distances for weighted automata. Our work is mostly inspired by simulation distances for reactive systems as followed by the series of works from de Alfaro et al. [66, 64, 59, 60], however, adapted to take into account modalities. Another related line of research in the area of quantitative analysis of systems is that of robustness properties of timed systems, originating in [153, 178]. In robust model-checking, one considers enlarged semantics of timed automata where all the clock constraints are widened by perturbation, in order to model the imprecisions of clocks. If no new behaviour is added by this pertubation, then the system under consideration is robust, satisfying the property also under minor timing inaccuracies of the concrete implementation.

For a comprehensive survey on distances for weighted automata, detailed comparisons and logical characterizations of distances, applications to timed automata, etc. we refer the reader to [172].

The results presented in this chapter can be generalized in several directions. Some are covered in our recent work [16], including the generalization to arbitrary distances and weight structures. Environment correctness has not been defined in this chapter. It would be interesting to study distances for environment correctness which depends on the particular interpretation of the weight intervals. The result would be a "quantitative" specification theory which would be a generalization of the specification theory presented in Chapter 2 by refinement distances and environment correctness distances.

Publication history. The present chapter is based on [14, 15] and contains additional material not covered by this thesis chapter, for instance, the logical characterization from Section 5.4 has been completely lifted from the qualitative to the quantitative setting. Furthermore, we have shown in [14, 15] that certain results of specification operators like conjunction and quotient can only be obtained to some extent under certain requirements on the underlying weight structure.

Additional results that are out of the scope of this thesis include the following works: In [15] we study a quantified notion of satisfaction of HML-formulae that can be used to provide a characterization of modal refinement distances. How to generalize modal refinement distances to arbitrary weight structures $\mathcal{K}_{intv}$ and with an arbitrary strategy how to measure distances of specifications is presented in [16].

6.4 Summary

We have shown in this chapter that within the quantitative specification framework of $\mathcal{K}_{intv}$-weighted modal input/output automata, modal refinement distances provide a useful tool for measuring quantitative differences in refinements. Modal refinement distances are proven equivalent to a natural thorough notion of distances if the abstract $\mathcal{K}_{intv}$-WMIO is deterministic. We have shown that modal synchronous composition and modal refinement distances satisfy a quantified compositionality property.

Chapter 7

Moving from Specification Theories to Contracts

Assume-guarantee reasoning [145, 113] is a well-studied approach which proposes proof methods to verify global properties of a system by proving local properties of individual components of the system. For the verification of local properties, system components are equipped with a specification that consists of an assumption on its environment, and a guarantee of the component itself given that the environment meets the assumption. Assume-guarantee reasoning has been very successfully applied in formal specification and verification, and in particular in model checking, see e.g. [150].

The motivation for investigating assume-guarantee reasoning in the context of our research is that, what we have seen so far, only a single specification was used to describe the behaviour of a component, with implicit assumptions on the environment. We would like to make these assumptions explicit by splitting the specification into an assumption on the environment describing what the environment has to fulfil in order to correctly use the component, and a guarantee that describes the behaviour of the component given that the environment fulfils the assumption. This splitting into explicit assumptions and guarantees has the following advantages over single, monolithic specifications of components:

- Assume-guarantee specification style clearly distinguishes between the *external* and *internal view* on the component. The external view (i.e. the assumption) only specifies the correct usage of the component, without any implementation details which are usually not of interest for the environment. The internal view (i.e. the guarantee) is the specification of the behaviour of the component which might or might not include implementation details. From a practical point of view, having a single specification with implicit assumptions can distort the component description and it can be rather difficult or even not feasible to figure out from a single specification what are its guarantees and what are its assumptions on the environment. Moreover, single specifications might lead to unnecessary complex verification problems, e.g. compatibility checking of environment and component behaviour, that might arise if the behaviour contains implementation details that need to be considered.

- Reusing already existing, implemented and tested components is a key concept in component-based development. Assume-guarantee reasoning supports this fundamental concept by reusing specifications of component guarantees and only specifying assumptions for the current use case (or for the environment of interest in which it will be used). The splitting allows to easily exchange assumptions and to adapt specifications according to the changing circumstances in the environment.

In the context of behaviour specifications for components that are based on state transition systems, several approaches were proposed that extend transition systems by assume-guarantee proof rules, for instance, such theories exist for discrete systems [123, 154] and probabilistic systems [179, 67]. A more general approach to deal with assume-guarantee rules was proposed by Benveniste et al. [33] where they use sets of traces as specifications and build contracts on top of it. Those approaches were proposed mostly independently of each other, with a similar aim of achieving sound and compositional proof rules.

Even though the specification theory is abstract enough to view the above theories for assume-guarantee reasoning again as a specification theory, we would like to investigate in this chapter whether there is a uniform way of defining a *contract theory* (i.e. a theory with assume-guarantee specifications and associated proof rules) on top of a given specification theory. Thus, the goal is to investigate to which extent we can define assume-guarantee reasoning *within* a specification theory, and what are the necessary requirements of the specification theory to ensure sound proof rules. This study should deliver a uniform abstract theory of contracts, and we demonstrate how a contract framework can be derived from a specification theory, using our abstract constructions. As a result we are able to instantiate “for free” a contract theory out of any specification theory. Any such derived contract theory is automatically equipped with:

- *Behaviour semantics* and *environment semantics* of contracts reflecting the set of behaviours and environments that satisfy the guarantees and assumptions of the contract, respectively.

- A *refinement relation* that allows to compare contracts in terms of sets of correct behaviours and environments.

- Sufficient conditions called *dominance* for a sound parallel composition of contracts which encapsulates contracts for two communicating components into one contract for the composition of the two.

Furthermore, as an important contribution of this chapter, we identify additional specification operators that guarantee a direct definition of contract composition as the strongest dominating contract. A specification theory is *complete* if it is equipped with *conjunction* to compute a greatest lower bound with respect to refinement, *quotient* to compute the residual of specifications with respect to parallel composition, and a *maximal environment operator* to compute largest correct environments. When the underlying specification theory is complete in the above sense and every contract has a normal form, then contract composition can be directly defined. Finally, an additional assumption on the interplay of composition and environment correctness guarantees that we obtain a specification theory for contracts.

In the course of this chapter, we show how the generic constructions can be applied to $Th^{d\mathbb{MIO}}_{strong}$ for which we show is a complete specification theory. We also show contracts for deterministic MIODs with strong modal refinement and weak environment correctness; in both instantiations we illustrate the abstract concepts with several small examples. We already point out that our framework can only be meaningfully instantiated by deterministic MIOs and MIODs with strong environment correctness; contracts instantiated for weak theories with weak environment correctness do not seem adequate and we illustrate this claim by an example. These limitations of our work are again explained in Section 7.5, together with possible solutions.

At this point, we would like to make our ideas more concrete by illustrating them with a simple example from our domain of deterministic MIOs, see Figure 7.1. Assume an implementation I of a simple message processing component that receives messages, processes them, and then acknowledges the processing to the sender of the messages. It is important to note that I can receive more than one message during the processing with the effect that each received message is processed, and the acknowledgement is sent once for possibly many messages. The MIO G (with the same action alphabet as I) is the behaviour that I should comply to. Clearly, I is not a strong modal refinement of G, since the G does only allow single message processing with one acknowledgement for each processed message.

However, the implementation I is already very “close” to G (in sense of $\leq_s$) and can indeed be very well used in those situations in which the environment waits with sending the next message until it receives the corresponding acknowledgement. This condition on the environment is expressed as assumption A, seen in Figure 7.2. Actually, A expresses even more: after a message has been sent, the environment must accept the acknowledgement, a reasonable assumption on the environment in this context. The specification G has now grown to the contract (A,G), with A the assumption and G the guarantee. Now, I is a correct behaviour of G given that the environment does not make use of the additional message processing feature that I offers. Thus, we have widened the set of re-

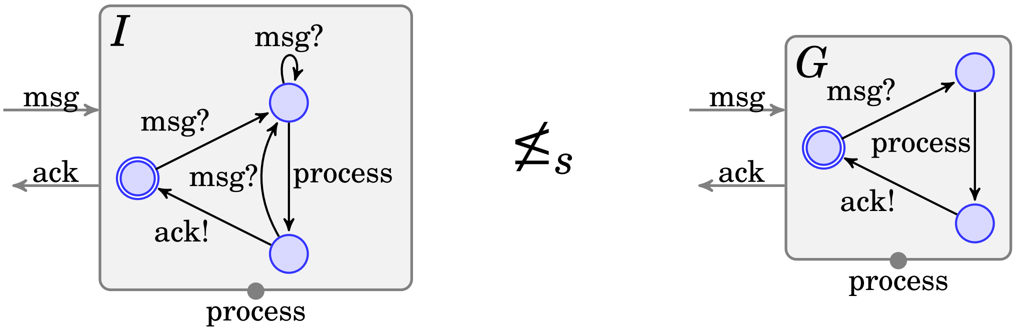


Figure 7.1: Implementation I does not refine the specification G

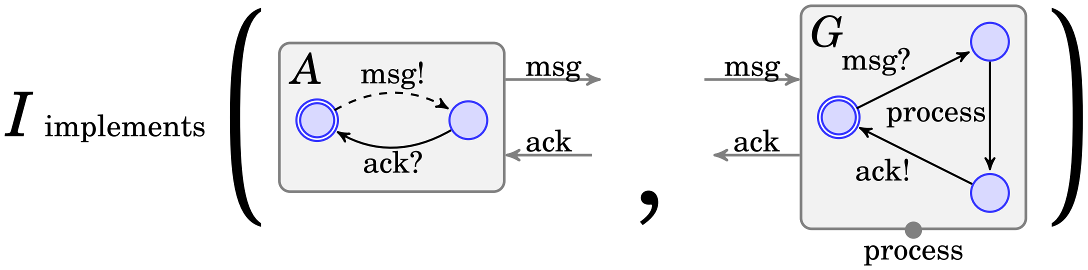


Figure 7.2: Implementation I correctly implements the contract (A,G)

finements of G by explicitly stating assumptions on the environment, and I is a correct refinement of G as long as it is used in a context (i.e. environment) which satisfies the assumption A.

Outline. We start by introducing complete specification theories in Section 7.1. In Section 7.2 we will define, for a given specification theory, the notion of a contract and its semantics, contract refinement and contract composition. In particular, if the underlying specification theory is complete we show how to define a specification theory for contracts. We then instantiate contracts to deterministic MIOs and MIODs and show several examples of such modal contracts that illustrate contract semantics and contract composition. In Section 7.3 the instantiation is shown for a complete specification theory for deterministic MIOs, Section 7.4 illustrates modal contracts based on deterministic MIODs. In Section 7.5 we summarize limitations of the contract framework. Related works are mentioned in Section 7.6 and we conclude this chapter with a summary in Section 7.7.

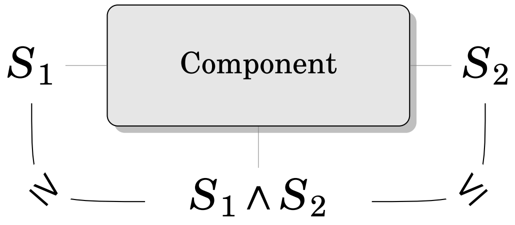


Figure 7.3: Illustration of conjunction

7.1 Complete Specification Theories

Some theories, which we call *complete specification theories*, offer additional operators that are useful in designing interface specifications for component-based systems. Recent works on concrete instances of interface theories [73, 57, 158] have identified conjunction and quotient as two important ingredients of any specification theory. The former, conjunction, computes a largest common refinement of two specifications, whereas the latter, quotient, is adjoint to composition which allows to compute residuals. In addition, we introduce a new operator not seen before, computing a maximal refinement of a specification such that it is a correct environment for a given specification. We see later in Section 7.2 that we can use them to derive most permissive assumptions achieving maximality of contract compositions.

Conjunction. When two separate teams independently develop specifications that are intended to be realized by the *same* component, then it is useful to have *conjunction*, denoted $\wedge$, that computes a most general specification that realizes both specifications (if this is possible, i.e. if there exists a common refinement), cf. Figure 7.3. When the first team proposes a specification S_1 which a component C should comply with, and the second team proposes S_2 for C, then $S_1 \wedge S_2$ is the conjunction of the two specifications which is defined whenever both S_1 *and* S_2 can be refined at the same time. Moreover, $S_1 \wedge S_2$ is a most general specification with this property. Formally, conjunction is a partial commutative function

$$\wedge : \mathfrak{S} \times \mathfrak{S} \to \mathfrak{S}$$

such that

$$S_1 \wedge S_2 \text{ is defined if and only if } \exists X \in \mathfrak{S} : X \leq S_1 \text{ and } X \leq S_2,$$

and, if $S_1 \wedge S_2$ is defined, then

$$S_1 \wedge S_2 \leq S_1 \text{ and } S_1 \wedge S_2 \leq S_2, \text{ and}$$
$$\forall X \in \mathfrak{S} : (X \leq S_1 \text{ and } X \leq S_2) \text{ implies } X \leq S_1 \wedge S_2.$$

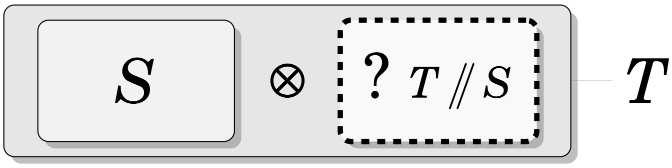


Figure 7.4: Illustration of the quotient operation

Conjunction $S_1 \wedge S_2$, if it exists, is unique up to equivalence $\approx$ of specifications (see Chapter 2, Definition 2.1.1). Note that the definition also immediately implies that $[\![S_1 \wedge S_2]\!] = [\![S_1]\!] \cap [\![S_2]\!]$ whenever $S_1 \wedge S_2$ is defined.

Quotient. Specification theories sometimes come along with a *quotient* operator, written $/\!\!/$, which is dual to parallel composition (or in other words $/\!\!/$ is adjoint to $\otimes$), cf. Figure 7.4. When given a requirement specification T of the overall system and another specification S that is supposed to be part of the system, then the quotient $T /\!\!/ S$ is a most general solution X satisfying

$$S \otimes X \leq T$$

i.e. $T /\!\!/ S$ is a most general specification such that $S \otimes (T /\!\!/ S) \leq T$. Formally, quotient is a partial function

$$/\!\!/ : \mathfrak{S} \times \mathfrak{S} \to \mathfrak{S}$$

that satisfies:

$$T /\!\!/ S \text{ is defined if and only if } \exists X \in \mathfrak{S} : S \otimes X \leq T.$$

And if $T /\!\!/ S$ is defined, then

$$S \otimes (T /\!\!/ S) \leq T, \text{ and } \forall X \in \mathfrak{S} : S \otimes X \leq T \Rightarrow X \leq T /\!\!/ S.$$

The requirements of a quotient operator implies that $T /\!\!/ S$ is unique up to equivalence of specifications.

Maximal Environment Operator. For obtaining maximally correct environments for a given specification we stipulate the existence of an operation that, given two specifications $S, E \in \mathfrak{S}$ such that S and E are composable, computes a largest refinement $E' \in \mathfrak{S}$ of E such that $S \to E'$. Formally, we require the existence of a partial operation

$$\mathsf{max}_{\cdot\to}(\cdot) : \mathfrak{S} \times \mathfrak{S} \to \mathfrak{S}$$

such that

$$\mathsf{max}_{S\to}(E) \text{ is defined if and only if } \exists E' \in \mathfrak{S} : E' \leq E \text{ and } S \to E',$$

and if $\mathsf{max}_{S\rightarrow}(E)$ is defined, then

$$\begin{aligned}&\mathsf{max}_{S\rightarrow}(E) \le E \text{ and } S \rightarrow \mathsf{max}_{S\rightarrow}(E),\\ &\forall E' \in \mathfrak{S} : (E' \le E \text{ and } S \rightarrow E') \text{ implies } E' \le \mathsf{max}_{S\rightarrow}(E)\end{aligned}$$

Again, $\mathsf{max}_{S\rightarrow}(E)$ is unique up to $\approx$.

Definition 7.1.1 (Complete Specification Theory)
A specification theory $(\mathfrak{S}, \mathfrak{S}^i, \le, \otimes, \rightarrow)$ *is* complete *if it offers conjunction* $\wedge$*, quotient* $/\!\!/$*, and a maximal environment operator* $\mathsf{max}_{\cdot\rightarrow}(\cdot)$.

7.2 Contracts and Their Semantics

For the development of our abstract contract framework, we assume to be given a specification theory $(\mathfrak{S}, \mathfrak{S}^i, \le, \otimes, \rightarrow)$ as defined in Chapter 2. Only later, when we give a direct definition of contract composition we stick to complete specification theories, so the first part of this section also applies to specification theories which are not complete.

Assumption 1. $(\mathfrak{S}, \mathfrak{S}^i, \le, \otimes, \rightarrow)$ *is a specification theory.*

We start by defining the notion of a contract which explicitly distinguishes between assumptions and guarantees.

Definition 7.2.1 (Contract)
A contract *is a pair* (A, G) *where* $A, G \in \mathfrak{S}$ *are two specifications such that* $G \rightarrow A$.

In a contract (A, G), the specification A expresses the assumption on the environment of the component, whereas the specification G describes the guarantee of any component to the environment given that the environment respects the assumption A. In any contract (A, G) the assumption A should specify a correct environment for G, i.e. $G \rightarrow A$.

For the definition of when a specification is a correct behaviour of a given contract, we use a notion of *relativized refinement* which is derived from the refinement relation of the underlying specification theory.

Definition 7.2.2 (Relativized Refinement)
Relativized refinement is the ternary relation in $\mathfrak{S} \times \mathfrak{S} \times \mathfrak{S}$ *defined as follows: for all* $S, E, T \in \mathfrak{S}$,

$$\begin{aligned}S \le_E T \quad \textit{if and only if} \quad &\forall E' \in \mathfrak{S} : (E' \le E \textit{ and } T \rightarrow E')\\ &\qquad \Rightarrow (S \rightarrow E' \textit{ and } S \otimes E' \le T \otimes E').\end{aligned}$$

$S \le_E T$ intuitively means that S refines T if both are put in any correct environment E' that refines E. The following lemmata summarize properties of relativized refinement that are easy consequences of the definition.

Lemma 7.2.3
Relativized refinement is a preorder, and for all $S, E, E', T \in \mathfrak{S}$, if $S \leq_E T$ and $E' \leq E$ then $S \leq_{E'} T$.

Proof. Reflexivity and transitivity of relativized refinement is easy to see. For the second claim, assume that $S \leq_E T$ and $E' \leq E$. Let $E'' \leq E'$ and assume that $T \to E''$. By transitivity of refinement also $E'' \leq E$, hence we can use our assumption $S \leq_E T$ to infer $S \to E''$ and $S \otimes E'' \leq T \otimes E''$. Hence $S \leq_{E'} T$ which was to be shown. □

We note that for the second part of Lemma 7.2.3, switching the context by a refined one, the universal quantification in Definition 7.2.2 is essential.

Lemma 7.2.4
Let $S, T, E \in \mathfrak{S}$ and assume that $T \to E$. Then $S \leq_E T$ if and only if

$$\forall E' \in \mathfrak{S} : E' \leq E \Rightarrow S \to E' \text{ and } S \otimes E' \leq T \otimes E'.$$

Proof. Assume that $S \leq_E T$ and let $E' \in \mathfrak{S}$ such that $E' \leq E$. Then $S \to E'$ follows from $S \leq_E T$, the definition of relativized refinement and $T \to E'$. The same argument implies $S \otimes E' \leq T \otimes E'$. For the other direction, let $E' \in \mathfrak{S}$ such that $E' \leq E$ and $T \to E'$. Then $S \to E'$ and $S \otimes E' \leq T \otimes E'$. □

Lemma 7.2.5
Let $S, T \in \mathfrak{S}$. Then $S \leq T$ implies $S \leq_E T$ for any $E \in \mathfrak{S}$.

Proof. Let $E' \in \mathfrak{S}$ such that $E' \leq E$ and $T \to E'$. By preservation of environment correctness, $S \leq T$ implies $S \to E'$. By compositionality of refinement, we can infer $S \otimes E' \leq T \otimes E'$. □

Let us now turn to the definition of the semantics of a contract. The *behaviour semantics* of a contract (A, G) is given by the set of all specifications that satisfy the contract guarantee G *under the assumption* A:

$$[\![C]\!]_{\mathrm{beh}} = \{I \in \mathfrak{S} \mid I \leq_A G\}.$$

This is a significant generalization of pure specification theories where it is usually assumed that refinements must literally satisfy the specification. The *environment semantics* of the contract (A, G) consists of all environments for (or users of) the component satisfying the assumption A of the contract:

$$[\![C]\!]_{\mathrm{env}} = \{E \in \mathfrak{S} \mid E \leq A\}.$$

In summary, the semantics of a contract is given by both behaviour semantics and environment semantics. Two contracts are *semantically equivalent* if they have the same (behaviour and environment) semantics.

Remark 7.2.6
In the following, for a given contract C, we use the terms behaviour *and* environment *for specifications in* $[\![C]\!]_{\text{beh}}$ *and* $[\![C]\!]_{\text{env}}$*, respectively.*

Our first result is a direct consequence of the definition of a contract and contract semantics: Whenever one has a correct environment and a correct behaviour of a contract, then their composition is a refinement of the composition of assumption and guarantee of the contract. In other words, the following theorem states the adequacy of contracts semantics.

Theorem 7.2.7 (Adequacy of Contract Semantics)
Let $C = (A,G)$ *be a contract. For all specifications* $E,I \in \mathfrak{S}$*, if* $E \in [\![C]\!]_{\text{env}}$ *and* $I \in [\![C]\!]_{\text{beh}}$ *then* $I \to E$ *and* $E \otimes I \leq A \otimes G$*.*

Proof. The claim follows from Lemma 7.2.4, by choosing $T = G$, $E = A$, $E' = E$ and $S = I$, and by applying transitivity of refinement. □

This theorem is not used in any later proofs. Still importantly, it states that both behaviour and environment semantics are adequate, i.e. meeting our intuition about what should result whenever we compose correct behaviours and environments: their composition should refine the interaction $A \otimes G$ that is described by any contract (A,G).

The behaviour semantics of a contract in general depends on both the assumption A and the guarantee G. However, when any refinement of G is independent of the assumption A, we say that the contract (A,G) is in normal form.

Definition 7.2.8 (Normal Form)
A contract $C = (A,G)$ *is* in normal form *if for all specifications* $I \in \mathfrak{S}$*,* $I \leq_A G$ *if and only if* $I \leq G$*.*

We say that a contract (A,G) *has a normal form* if it can by transformed into a semantically equivalent contract (A,G^{nf}) in normal form by weakening of G to G^{nf}. In Section 7.3 we show that deterministic MIOs are powerful enough to allow such a semantic-preserving transformation for any contract (A,G). If in a specification theory every contract has a normal form, then we say that the specification theory *has normal forms*.

7.2.1 Refinement of Contracts

Next, we turn to the question how contracts can be refined. We follow here a standard approach inspired by notions of behavioural subtyping [131] and say that a contract C' refines another contract C if C' admits less behaviours than C, but more legal environments than C.

Definition 7.2.9 (Contract Refinement)
Let C and C' be two contracts. The contract C' refines the contract C (is stronger than C), written $C' \sqsubseteq C$, if $[\![C']\!]_{\text{beh}} \subseteq [\![C]\!]_{\text{beh}}$ and $[\![C']\!]_{\text{env}} \supseteq [\![C]\!]_{\text{env}}$.

The refinement relation between contracts is reflexive and transitive. Obviously, two contracts C, C' are semantically equivalent if and only if $C' \sqsubseteq C$ and $C \sqsubseteq C'$. The following theorem characterizes contract refinement by contra-/covariant (relativized) refinement of corresponding assumptions and guarantees.

Theorem 7.2.10 (Characterization of Contract Refinement)
Let (A,G) and (A',G') be two contracts. Then $(A',G') \sqsubseteq (A,G)$ if and only if $A \leq A'$ and $G' \leq_A G$.

Proof. First we show that $(A',G') \sqsubseteq (A,G)$ implies $A \leq A'$ and $G' \leq_A G$.

$$A \in [\![(A,G)]\!]_{\text{env}} \subseteq [\![(A',G')]\!]_{\text{env}}$$

implies $A \leq A'$, and similarly,

$$G' \in [\![(A',G')]\!]_{\text{beh}} \subseteq [\![(A,G)]\!]_{\text{beh}}$$

implies $G' \leq_A G$.

Second, for the other direction, we have to show that $A \leq A'$ and $G' \leq_A G$ imply $(A',G') \sqsubseteq (A,G)$. Thus we have to show

(a) $[\![(A',G')]\!]_{\text{beh}} \subseteq [\![(A,G)]\!]_{\text{beh}}$,

(b) $[\![(A,G)]\!]_{\text{env}} \subseteq [\![(A',G')]\!]_{\text{env}}$.

To show (a), let $I \in [\![(A',G')]\!]_{\text{beh}}$, so $I \leq_{A'} G'$. Then also $I \leq_A G'$ by Lemma 7.2.3 and $A \leq A'$. From $G' \leq_A G$ and transitivity of relativized refinement it follows that $I \leq_A G$ which is $I \in [\![(A,G)]\!]_{\text{beh}}$. To show (b), let $E \in [\![(A,G)]\!]_{\text{env}}$, so $E \leq A$. From $A \leq A'$ and transitivity of refinement it follows that $E \leq A'$, and thus $E \in [\![(A',G')]\!]_{\text{env}}$. □

An immediate consequence is the following corollary:

Corollary 7.2.11
Let (A,G) and (A',G') be two contracts such that (A,G) is in normal form. Then $(A',G') \sqsubseteq (A,G)$ if and only if $A \leq A'$ and $G' \leq G$.

Remark 7.2.12
Thorough refinement of contracts and the question of completeness of refinement, cf. Section 2.1, is not considered here and left for future work.

7.2.2 Dominance and Composition of Contracts

When behaviours I_1 and I_2 of individual components are composed, their composition is only semantically meaningful if the contracts, say C_1, C_2, of the single components fit together. This means that there exists a 'larger' contract C which subsumes C_1 and C_2 such that (1) the composition of any behaviours of C_1 and C_2 is a correct behaviour of C, and (2) each correct environment of C controls the single behaviours in such a way that they mutually satisfy the assumptions of the single contracts. Inspired by [154] we call such a contract C a *dominating* contract for C_1 and C_2.

Definition 7.2.13 (Dominance)
Let C, C_1 and C_2 be contracts. C dominates C_1 and C_2 *if the following two conditions are satisfied:*

1. *Any composition of correct behaviours of C_1 and C_2 results in a correct behaviour of the contract C:*
   - $\forall I_1 \in [\![C_1]\!]_{\mathrm{beh}} : \forall I_2 \in [\![C_2]\!]_{\mathrm{beh}} : I_1 \otimes I_2$ *is defined and* $I_1 \otimes I_2 \in [\![C]\!]_{\mathrm{beh}}$
2. *For any correct environment of C, the composition with a correct behaviour of C_1 (C_2) results in a correct environment of C_2 (C_1, respectively): for all $E \in [\![C]\!]_{\mathrm{env}}$,*
   - $\forall I_1 \in [\![C_1]\!]_{\mathrm{beh}} : E \otimes I_1$ *is defined and* $E \otimes I_1 \in [\![C_2]\!]_{\mathrm{env}}$,
   - $\forall I_2 \in [\![C_2]\!]_{\mathrm{beh}} : E \otimes I_2$ *is defined and* $E \otimes I_2 \in [\![C_1]\!]_{\mathrm{env}}$.

We say that two contracts C_1, C_2 are dominable *if there exists a contract C dominating C_1, C_2.*

An immediate consequence of the definition of dominance is the following lemma.

Lemma 7.2.14
Let C, C_1 and C_2 be contracts. If C dominates C_1 and C_2 then for any $E \in [\![C]\!]_{\mathrm{env}}$, $I_1 \in [\![C_1]\!]_{\mathrm{beh}}$ and $I_2 \in [\![C_2]\!]_{\mathrm{beh}}$, it holds that

1. $I_1 \otimes I_2 \to E$,
2. $I_1 \to E \otimes I_2$,
3. $I_2 \to E \otimes I_1$.

Proof. Let $C = (A, G)$, $C_1 = (A_1, G_1)$ and $C_2 = (A_2, G_2)$. Condition 1 of Definition 7.2.13 requires $I_1 \otimes I_2 \in [\![C]\!]_{\mathrm{beh}}$ which is $I_1 \otimes I_2 \leq_A G$. Since $G \to A$ and $E \leq A$, we also have $I_1 \otimes I_2 \to E$ by applying Lemma 7.2.4 for $T = G$, $E = A$, $S = I_1 \otimes I_2$, $E' = E$. Furthermore, condition 2 of Definition 7.2.13 requires $E \otimes I_1 \in$

$[\![C_2]\!]_{\text{env}}$ which implies $E \otimes I_1 \leq A_2$. Since $G_2 \to A_2$ and $I_2 \leq_{A_2} G_2$ it follows from Lemma 7.2.4 (applied for $T = G$, $E = A_2$, $S = I_2$ and $E' = E \otimes I_1$) that $I_2 \to E \otimes I_1$. The third claim $I_1 \to E \otimes I_2$ is symmetric. □

Thus, when taking any legal environment E according to the dominating contract C and any implementations I_1 composed with I_2 of C_1 and C_2, respectively, then I_1 and I_2 feel well in the environment E, but also individually they feel well in the composed environment of E together with the other implementation.

Dominance is preserved under refinement of individual contracts.

Theorem 7.2.15 (Preservation of Dominance under Refinement)
Let C_1, C_1', C_2, C_2', C be contracts such that $C_1' \sqsubseteq C_1$ and $C_2' \sqsubseteq C_2$. If C dominates C_1 and C_2, then C dominates C_1' and C_2'.

Proof. This theorem follows from the semantic definition of contract refinement and the fact that dominance of two contracts C_1 and C_2 is based on the semantics of the individual contracts in such a way that by replacing C_1 with C_1' and C_2 with C_2' all conditions continue to hold. Formally, we have $[\![C_i']\!]_{\text{beh}} \subseteq [\![C_i']\!]_{\text{beh}}$ and $[\![C_i']\!]_{\text{env}} \supseteq [\![C_i]\!]_{\text{env}}$, for $i = 1,2$. Observing the conditions of dominance, we can see that we can replace any occurrence of $[\![C_i']\!]_{\text{beh}}$ by $[\![C_i']\!]_{\text{beh}}$ without changing the satisfaction of the statements, and similarly for the environment semantics. □

For the following results, we generally assume that the underlying specification theory has normal forms, i.e. for any $C = (A,G)$ there exists a semantically equivalent contract $C^{nf} = (A^{nf}, G^{nf})$ which is in normal form. Due to the definition of environment semantics, without loss of generality, we can always assume in the following that $A^{nf} = A$.

Assumption 2. *The specification theory $(\mathfrak{S}, \mathfrak{S}^i, \leq, \otimes, \to)$ has normal forms.*

The following lemma is a consequence of the definition of a dominating contract.

Lemma 7.2.16
Two contracts C_1 and C_2 are dominable if and only if their normal forms C_1^{nf} and C_2^{nf} are dominable.

Proof. This simply follows from the fact that the definition of dominance of two contracts C_1 and C_2 only relies on their semantics, and normal forms C_1^{nf} and C_2^{nf} are semantically equivalent to C_1 and C_2, respectively. □

The next theorem provides a characterization of dominance for specification theories with normal forms. The idea is that there must exist an environment under which behaviours of the single contracts can be adapted to meet each others assumptions, while satisfying the remaining assumptions on the environment of the composed system.

Theorem 7.2.17
Let $C_1 = (A_1, G_1)$ and $C_2 = (A_2, G_2)$ be two contracts with normal forms $C_1^{nf} = (A_1, G_1^{nf})$ and $C_2^{nf} = (A_2, G_2^{nf})$, respectively. The following are equivalent:

1. *C_1 and C_2 are dominable,*
2. *there exists $E \in \mathfrak{S}$ such that*
   (a) *$E \otimes G_1^{nf} \leq A_2$ and $E \otimes G_2^{nf} \leq A_1$,*
   (b) *$G_1^{nf} \otimes G_2^{nf} \rightarrow E$.*

Proof. By Lemma 7.2.16, we can show the equivalence of the two conditions for contracts in normal form.

First, assume that C_1^{nf} and C_2^{nf} are dominable, say by a contract $C = (A, G)$. Note that $A \in [\![C]\!]_{\text{env}}$, $G_1^{nf} \in [\![C_1^{nf}]\!]_{\text{beh}}$, and $G_2^{nf} \in [\![C_2^{nf}]\!]_{\text{beh}}$. Then, by the definition of dominance, we can conclude that $A \otimes G_1^{nf} \leq A_2$ and $A \otimes G_2^{nf} \leq A_1$. The claim $G_1^{nf} \otimes G_2^{nf} \rightarrow E$ follows from the first condition of dominance: We know that $G_1^{nf} \otimes G_2^{nf} \in [\![C]\!]_{\text{beh}}$ and $G \rightarrow A$, hence by perservation of environment correctness we get $G_1^{nf} \otimes G_2^{nf} \rightarrow A$, hence $G_1^{nf} \otimes G_2^{nf} \rightarrow E$.

Second, for the other direction, we assume that there exists $E \in \mathfrak{S}$ such that $E \otimes G_1^{nf} \leq A_2$, $E \otimes G_2^{nf} \leq A_1$, and $G_1^{nf} \otimes G_2^{nf} \rightarrow E$. It is easy to verify that the contract $(E, G_1^{nf} \otimes G_2^{nf})$ dominates C_1^{nf} and C_2^{nf}. □

This theorem shows that in order to find a contract that dominates two contracts (A_1, G_1^{nf}) and (A_2, G_2^{nf}) it suffices to come up with an E such that the above two points are satisfied; then $(E, G_1^{nf} \otimes G_2^{nf})$ is a dominating contract. In the following we propose two different variants of finding such a “good” environment E:

(Variant 1) In this variant we assume that assumptions are decomposable leading to a simple syntactical construction of a “good” environment E.

(Variant 2) The most general case is covered by the second variant. It relies on completeness of the underlying specification theory, and a “good” environment E can be constructed by using quotient, conjunction and the maximal environment operator. In particular, the so constructed E is shown to be a most permissive environment E rendering $(E, G_1^{nf} \otimes G_2^{nf})$ a strongest dominating contract.

Contract Dominance based on Decomposability of Assumptions (Variant 1)

Let us consider two contracts $C_1 = (A_1, G_1^{nf})$ and $C_2 = (A_2, G_2^{nf})$ in normal form for which we would like to find a dominating contract $(E, G_1^{nf} \otimes G_2^{nf})$. The situation is illustrated in Figure 7.5: If

- the assumptions A_1 and A_2 of C_1 and C_2 are decomposable into separate assumptions A'_1, A''_1 and A'_2, A''_2, respectively, and
- A''_1 and A''_2 are satisfied by $G_2^{nf} \otimes A'_2$ and $G_1^{nf} \otimes A'_1$, respectively, and
- $A'_1 \otimes A'_2$ is a correct environment of $G_1^{nf} \otimes G_2^{nf}$,

then E can be easily obtained by composing the assumptions A'_1 and A'_2 that specify the requirements for the remaining environment. The so obtained dominating contract is shown in Figure 7.6.

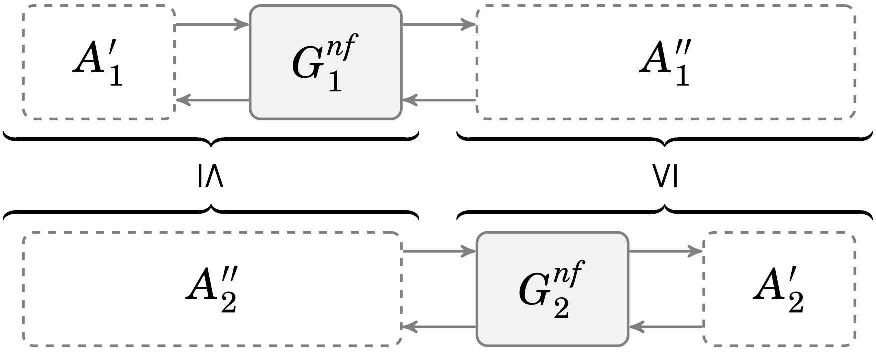


Figure 7.5: Contracts $C_1 = (A'_1 \otimes A''_1, G_1^{nf})$ and $C_2 = (A'_2 \otimes A''_2, G_2^{nf})$ with decomposable assumptions such that $A'_1 \otimes G_1^{nf} \leq A''_2$ and $A'_2 \otimes G_2^{nf} \leq A''_1$

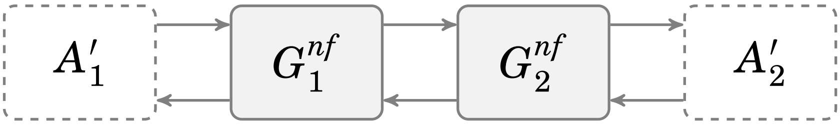


Figure 7.6: Contract $(A'_1 \otimes A'_2, G_1^{nf} \otimes G_2^{nf})$ which dominates the contracts shown in Figure 7.5

Theorem 7.2.18
Let $C_1 = (A_1, G_1)$ and $C_2 = (A_2, G_2)$ be two contracts with normal forms (A_1, G_1^{nf}) and (A_2, G_2^{nf}). If if there exists $A'_1, A''_1, A'_2, A''_2 \in \mathfrak{S}$ such that

1. $A'_1 \otimes A''_1 \leq A_1$ *and* $A'_2 \otimes A''_2 \leq A_2$, *and*
2. $A'_2 \otimes G_2^{nf} \leq A''_1$ *and* $A'_1 \otimes G_1^{nf} \leq A''_2$, *and*
3. $G_1^{nf} \otimes G_2^{nf} \rightarrow A'_1 \otimes A'_2$,

then

$$(A'_1 \otimes A'_2, G_1^{nf} \otimes G_2^{nf})$$

dominates C_1 and C_2.

Proof. First, we note that $(A_1' \otimes A_2', G_1^{nf} \otimes G_2^{nf})$ is a valid contract since by (3.) $G_1^{nf} \otimes G_2^{nf} \to A_1' \otimes A_2'$. We show the two conditions of dominance, see Definition 7.2.13.

1. Let $I_1 \leq G_1^{nf}, I_2 \leq G_2^{nf}$. By compositionality of refinement we get $I_1 \otimes I_2 \leq G_1^{nf} \otimes G_2^{nf}$. Then by Lemma 7.2.5 it follows that
$$I_1 \otimes I_2 \leq_{A_1' \otimes A_2'} G_1^{nf} \otimes G_2^{nf}.$$
2. Let $E \leq A_1' \otimes A_2'$ and $I_1 \leq G_1^{nf}$. Then by compositionality of refinement
$$E \otimes I_1 \leq A_1' \otimes A_2' \otimes G_1^{nf}$$
which by assumption $A_1' \otimes G_1^{nf} \leq A_2''$ and transitivity of refinement and commutativity of $\otimes$ implies
$$E \otimes I_1 \leq A_2' \otimes A_2'' \leq A_2, \text{ hence } E \otimes I_1 \leq A_2$$
which was to be shown. The other condition $E \otimes I_2 \leq A_1$ can be shown similarly. □

Contract Composition with Underlying Complete Specification Theory (Variant 2)

In the previous variant we relied on decomposable assumptions such that all the conditions of Theorem 7.2.18 are satisfied. Clearly, decomposing assumptions accordingly is often not possible, which is the case as soon as there are behavioural dependencies between the two communication points (as illustrated in the figures) of the component.

What we present in this second variant of finding a environment for a dominating contract is much more general approach works as soon as the underlying specification theory is complete (see Section 7.1). In this case we can obtain a dominating contract by using the specification operators of the complete specification theory, without having to assume decomposable assumptions anymore. More precisely, when given two dominable contracts $C_1 = (A_1, G_1)$ and $C_2 = (A_2, G_2)$ with normal forms (A_1, G_1^{nf}) and (A_2, G_2^{nf}), then we can compute E that renders $(E, G_1^{nf} \otimes G_2^{nf})$ a dominating contract. Even better, we can show that the obtained E is a most permissive assumption rendering $(E, G_1^{nf} \otimes G_2^{nf})$ a strongest dominating contract for C_1 and C_2.

In general terms, we first define *contract composition* of two contracts C_1 and C_2 as the strongest dominating contract for C_1 and C_2, or in other words, a contract composition is a dominating contract which satisfies a universal property.

Definition 7.2.19 (Contract Composition)
A contract C is called contract composition *of the contracts C_1 and C_2 if*

1. *C dominates C_1 and C_2,*
2. *for all contracts C', if C' dominates C_1 and C_2, then $C \sqsubseteq C'$.*

Contract compositions, if they exist, are unique up to semantic equivalence of contracts. We will now turn to the question whether the composition of two contracts exists and, if so, whether it can be *constructively* defined in the presence of a complete specification theory. Thus, to answer this question we assume from now on a complete specification theory (recall that such a theory has quotient, conjunction and a maximal environment correctness operator, see Section 7.1) over which contracts are constructed.

Assumption 3. *The specification theory $(\mathfrak{S}, \mathfrak{S}^i, \leq, \otimes, \rightarrow)$ is complete.*

In what follows we show that for two given contracts (A_1, G_1^{nf}) and (A_2, G_2^{nf}) the specification

$$\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \rightarrow}((A_1 \,/\!\!/\, G_2^{nf}) \wedge (A_2 \,/\!\!/\, G_1^{nf}))$$

forms the most permissive assumptions E rendering $(E, G_1^{nf} \otimes G_2^{nf})$ a composition of C_1 and C_2. We first show that the definedness of the above construction is equivalent to dominability of C_1 and C_2.

Lemma 7.2.20
Let $C_1 = (A_1, G_1)$ and $C_2 = (A_2, G_2)$ be two contracts with normal forms (A_1, G_1^{nf}) and (A_2, G_2^{nf}). The following are equivalent:

1. $\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \rightarrow}((A_1 \,/\!\!/\, G_2^{nf}) \wedge (A_2 \,/\!\!/\, G_1^{nf}))$ *is defined,*
2. *C_1 and C_2 are dominable.*

Proof. We only need to show that

$$\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \rightarrow}((A_1 \,/\!\!/\, G_2^{nf}) \wedge (A_2 \,/\!\!/\, G_1^{nf}))$$

is defined if and only if there exists $E \in \mathfrak{S}$ such that $E \otimes G_1^{nf} \leq A_2$, $E \otimes G_2^{nf} \leq A_1$, and $G_1^{nf} \otimes G_2^{nf} \rightarrow E$, the rest follows from Theorem 7.2.17. For this proof, we basically use the assumptions on the specification operators as listed in Section 7.1.

The specification

$$\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \rightarrow}((A_1 \,/\!\!/\, G_2^{nf}) \wedge (A_2 \,/\!\!/\, G_1^{nf}))$$

is defined if and only if there exists X such that $X \leq (A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf})$ and $G_1^{nf} \otimes G_2^{nf} \to X$. The former two hold if and only if $X \leq (A_1 /\!\!/ G_2^{nf})$ and $X \leq (A_2 /\!\!/ G_1^{nf})$, and again, this is the case if and only if both quotients involved are defined. Hence, the definedness of

$$\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf}))$$

is equivalent to the fact that there exists X such that $X \otimes G_2^{nf} \leq A_1$, $X \otimes G_1^{nf} \leq A_2$, and $G_1^{nf} \otimes G_2^{nf} \to X$. □

We can now define contract composition of two dominable contracts.

Definition 7.2.21
Let $C_1 = (A_1, G_1)$ and $C_2 = (A_2, G_2)$ be two contracts with normal forms (A_1, G_1^{nf}) and $C_2 = (A_2, G_2^{nf})$. $C_1 \boxtimes C_2$ is defined if and only if C_1 and C_2 are dominable and then

$$C_1 \boxtimes C_2 \triangleq (\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf})), G_1^{nf} \otimes G_2^{nf}).$$

Note that $C_1 \boxtimes C_2$ is a valid contract since

$$\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf}))$$

is defined by Lemma 7.2.20, moreover,

$$G_1^{nf} \otimes G_2^{nf} \to \mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf}))$$

by definition of $\mathsf{max}_{\cdot \to}(\cdot)$.

The next theorem states that $\boxtimes$ yields a *strongest* dominating contract.

Theorem 7.2.22
If the contracts C_1 and C_2 are dominable, then $C_1 \boxtimes C_2$ is (up to semantic equivalence) the composition of C_1 and C_2.

Proof. By the previous Lemma 7.2.20,

$$C_1 \boxtimes C_2 = (\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf})), G_1^{nf} \otimes G_2^{nf})$$

is defined.

We first show that $C_1 \boxtimes C_2$ dominates C_1 and C_2. The first condition simply follows from compositionality of refinement. For the second condition, we have to consider some $E \in [\![C_1 \boxtimes C_2]\!]_{\text{env}}$, i.e.

$$E \leq \mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf})).$$

Since the right hand side is a correct environment for $G_1^{nf} \otimes G_2^{nf}$, also E is a correct environment for $G_1^{nf} \otimes G_2^{nf}$ which follows by preservation of compatibility. Since $E \otimes (G_1^{nf} \otimes G_2^{nf})$ is defined, also $E \otimes G_1^{nf}$ is defined. Then we can infer the following refinements:

$$\begin{aligned}
&E \otimes G_1^{nf} \\
&\leq \mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf})) \otimes G_1^{nf} \\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{(by compositionality of refinement)} \\
&\leq ((A_2 /\!\!/ G_1^{nf}) \wedge (A_1 /\!\!/ G_2^{nf})) \otimes G_1^{nf} \qquad \text{(by } \mathsf{max}_{E\to}(S) \leq S\text{)} \\
&\leq (A_2 /\!\!/ G_1^{nf}) \otimes G_1^{nf} \qquad \text{(conjunction is greatest lower bound w.r.t. } \leq\text{)} \\
&\leq A_2
\end{aligned}$$

Thus $E \otimes G_1^{nf} \leq A_2$ and $E \otimes G_2^{nf} \leq A_1$.

Now, we have to show that $C_1 \boxtimes C_2$ is the strongest dominating contract of C_1 and C_2. Assume $C' = (A', G')$ another dominating contract of C_1 and C_2. We know $A' \in \llbracket C' \rrbracket_{\mathrm{env}}$, hence by definition of dominance, we can conclude that $A' \otimes G_2^{nf} \leq A_1$, $A' \otimes G_1^{nf} \leq A_2$, and $G_1^{nf} \otimes G_2^{nf} \to A'$. It follows that

$$A' \leq \mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf})) \qquad (\star)$$

thus $A' \in \llbracket C_1 \boxtimes C_2 \rrbracket_{\mathrm{env}}$. If $I \in \llbracket C_1 \boxtimes C_2 \rrbracket_{\mathrm{beh}}$, then

$$I \leq_{\mathsf{max}_{G_1^{nf} \otimes G_2^{nf} \to}((A_1 /\!\!/ G_2^{nf}) \wedge (A_2 /\!\!/ G_1^{nf}))} G_1^{nf} \otimes G_2^{nf}.$$

Then we can infer by Lemma 7.2.3 and ($\star$) that $I \leq_{A'} G_1^{nf} \otimes G_2^{nf}$. By dominance we know that $G_1^{nf} \otimes G_2^{nf} \in \llbracket C' \rrbracket_{\mathrm{beh}}$ which is $G_1^{nf} \otimes G_2^{nf} \leq_{A'} G'$, hence by transitivity $I \leq_{A'} G'$ which is $I \in \llbracket C' \rrbracket_{\mathrm{beh}}$. □

The next theorem shows that contract refinement is preserved under contract composition.

Theorem 7.2.23 (Compositionality)
Let C_1, C_2, C_1', C_2' be contracts such that C_1 and C_2 are dominable. If $C_1' \sqsubseteq C_1$ and $C_2' \sqsubseteq C_2$ then C_1' and C_2' are dominable and $C_1' \boxtimes C_2' \sqsubseteq C_1 \boxtimes C_2$.

Proof. In this proof, we omit the superscript *nf* for the guarantees (and for the assumptions, as before) to improve readability. Assume that $C_1 = (A_1, G_1)$, $C_2 = (A_2, G_2)$, $C_1' = (A_1', G_1')$, and $C_2' = (A_2', G_2')$.

Dominability of C_1' and C_2' is already shown in Theorem 7.2.15. We now prove $C_1' \boxtimes C_2' \sqsubseteq C_1 \boxtimes C_2$. Let $E \in \llbracket C_1 \boxtimes C_2 \rrbracket_{\mathrm{env}}$, hence

$$E \leq \mathsf{max}_{G_1 \otimes G_2 \to}((A_1 /\!\!/ G_2) \wedge (A_2 /\!\!/ G_1)).$$

From the contract refinement of the individual contracts we can conclude that $A_1 \leq A_1'$, $A_2 \leq A_2'$, $G_1' \leq G_1$ and $G_2' \leq G_2$. Applying the properties of conjunction, quotient, and compositionality of refinement (see Chapter 2) we can infer that $(A_1 /\!\!/ G_2) \wedge (A_2 /\!\!/ G_1) \leq (A_1' /\!\!/ G_2') \wedge (A_2' /\!\!/ G_1')$. It follows that

$$\mathsf{max}_{G_1 \otimes G_2 \to}((A_1 /\!\!/ G_2) \wedge (A_2 /\!\!/ G_1)) \leq (A_1' /\!\!/ G_2') \wedge (A_2' /\!\!/ G_1')$$

and since the left hand side is a correct environment for $G_1' \otimes G_2'$ (by $G_1' \otimes G_2' \leq G_1 \otimes G_2$ and preservation of environment correctness), it holds that

$$\mathsf{max}_{G_1 \otimes G_2 \to}((A_1 /\!\!/ G_2) \wedge (A_2 /\!\!/ G_1)) \leq \mathsf{max}_{G_1' \otimes G_2' \to}((A_1' /\!\!/ G_2') \wedge (A_2' /\!\!/ G_1')), \qquad (\star)$$

hence $E \in [\![C_1' \boxtimes C_2']\!]_{\mathsf{env}}$.

Let $I \in [\![C_1' \boxtimes C_2']\!]_{\mathsf{beh}}$, thus

$$I \leq_{\mathsf{max}_{G_1' \otimes G_2' \to}((A_1' /\!\!/ G_2') \wedge (A_2' /\!\!/ G_1'))} G_1' \otimes G_2'.$$

Hence by Lemma 7.2.3 and $(\star)$ we get that

$$I \leq_{\mathsf{max}_{G_1 \otimes G_2 \to}((A_1 /\!\!/ G_2) \wedge (A_2 /\!\!/ G_1))} G_1' \otimes G_2'.$$

From the first condition of dominance we know that

$$G_1' \otimes G_2' \leq_{\mathsf{max}_{G_1 \otimes G_2 \to}((A_1 /\!\!/ G_2) \wedge (A_2 /\!\!/ G_1))} G_1 \otimes G_2$$

and by transitivity of refinement we can conclude that $I \in [\![C_1 \boxtimes C_2]\!]_{\mathsf{beh}}$ which finishes the proof. □

What is still missing to form a specification theory for contracts is a notion of environment correctness which is preserved by contract refinement. It is a slight discrepancy that arises here in the theory for contracts: Environment correctness must entail composability, thus our only choice is dominability as environment correctness which is a symmetric condition rather than a non-symmetric one. The reason why this discrepancy arises here is that composability is a semantic rather than a syntactic condition. Preservation of environment correctness then amounts to preservation of dominability which was proven in Theorem 7.2.15.

Lastly we have to prove that contract composition is commutative and associative. Interestingly, associativity can only be proven if we impose the following assumption on environment correctness.

Assumption 4. *Let $S_1, S_2, E \in \mathfrak{S}$ be specifications. If $S_1 \to S_2 \otimes E$ and $S_2 \to S_1 \otimes E$, then $S_1 \otimes S_2 \to E$.*

The above assumption requires that we can check individually whether S_1 and S_2 feel well in the remaining environment $S_2 \otimes E$ and $S_1 \otimes E$, respectively, in order to conclude that $S_1 \otimes S_2$ feels well in E. We will see later that strong environment correctness (for both MIOs and MIODs) satisfies this assumption, in contrast to weak environment correctness which does not satisfy this property.

Lemma 7.2.24
Contract composition $\boxtimes$ is commutative and associative.

Proof. Commutativity simply follows form commutativity of conjunction $\wedge$ and composition $\otimes$. For proving associativity of $\boxtimes$, assume $(C_1 \boxtimes C_2) \boxtimes C_3$ for contracts $C_i = (A_i, G_i)$, $1 \le i \le 3$. We show that $C_1 \boxtimes (C_2 \boxtimes C_3)$ is defined and both assumptions and guarantees are equivalent. For the guarantees this is easy to see:

$$(G_1 \otimes G_2) \otimes G_3 = G_1 \otimes (G_2 \otimes G_3)$$

follows from associativity of composition. Let us consider the assumptions: we have to prove that

$$\mathsf{max}_{(G_1 \otimes G_2) \otimes G_3 \to}((\mathsf{max}_{G_1 \otimes G_2 \to}((A_1 /\!\!/ G_2) \wedge (A_2 /\!\!/ G_1)) /\!\!/ G_3) \wedge (A_3 /\!\!/ (G_1 \otimes G_2))) \quad (1)$$

is equivalent to

$$\mathsf{max}_{G_1 \otimes (G_2 \otimes G_3) \to}((A_1 /\!\!/ (G_2 \otimes G_3)) \wedge (\mathsf{max}_{G_2 \otimes G_3 \to}((A_2 /\!\!/ G_3) \wedge (A_3 /\!\!/ G_2)) /\!\!/ G_1)). \quad (2)$$

Let X be a specification refining (1). Then

- $G_1 \otimes G_2 \otimes G_3 \to X$,
- $G_2 \otimes G_3 \otimes X \le A_1$, and thus also $G_1 \to G_2 \otimes G_3 \otimes X$,
- $G_1 \otimes G_3 \otimes X \le A_2$, and thus also $G_2 \to G_1 \otimes G_3 \otimes X$,
- $G_1 \otimes G_2 \otimes X \le A_3$, and thus also $G_3 \to G_1 \otimes G_2 \otimes X$.

It follows that $G_1 \otimes X \le A_2 /\!\!/ G_3$ and $G_1 \otimes X \le A_3 /\!\!/ G_2$, thus $G_1 \otimes X \le (A_2 /\!\!/ G_3) \wedge (A_3 /\!\!/ G_2)$. From Assumption 4 it follows that $G_2 \otimes G_3 \to G_1 \otimes X$, hence $G_1 \otimes X \le \mathsf{max}_{G_2 \otimes G_3 \to}((A_2 /\!\!/ G_3) \wedge (A_3 /\!\!/ G_2))$. Since we also know that $G_2 \otimes G_3 \otimes X \le A_1$ and hence $X \le A_1 /\!\!/ (G_2 \otimes G_3)$, we can infer altogether that

$$X \le (A_1 /\!\!/ (G_2 \otimes G_3)) \wedge (\mathsf{max}_{G_2 \otimes G_3 \to}((A_2 /\!\!/ G_3) \wedge (A_3 /\!\!/ G_2)) /\!\!/ G_1).$$

Finally, X also refines (2) because $G_1 \otimes G_2 \otimes G_3 \to X$. The other direction can be proven analogously. □

In order to obtain a specification theory for contracts, we have to fix a subset of contracts that form the implementations, i.e. contracts that are final elements with respect to contract refinement. As we do not further study these final elements here, we just define the set of final elements semantically: A contract C is an implementation if whenever $C' \sqsubseteq C$ for some contract C', then $C \sqsubseteq C'$. By Corollary 7.2.11 a sufficient condition for a contract (A,G) to be an implementation w.r.t. $\sqsubseteq$ is that (1) (A,G) is in normal form, (2) G is an implementation and (3) for any $A' \in \mathfrak{S}$, whenever $A \le A'$ and $A' \not\le A$, then $G \not\to A'$.

Corollary 7.2.25
Contracts itself form again a specification theory, with environment correctness defined as existence of a dominating contract.

Proof. Theorem 7.2.15 shows that dominance is preserved contract refinement. Compositionality of contract refinement is shown in Theorem 7.2.23. Contract implementations are final elements with respect to contract refinement. □

Remark 7.2.26
Unfortunately, due to the discrepancy mentioned before, it is not possible to define a specification theory morphism from a specification theory to its derived contract theory because definedness of $S \otimes T$ in the underlying specification theory does not imply definedness of $(A_S, S) \boxtimes (A_T, T)$ in general where A_S and A_T are some (possibly maximal) correct environments of S and T.

7.3 Modal Contracts based on $Th^{d\mathbb{MIO}}_{strong}$

In this section we illustrate our generic constructions for moving from a specification theory to contracts by considering the complete specification theory $Th^{d\mathbb{MIO}}_{strong}$ for deterministic MIOs. We first justify this claim in Section 7.3.1 that one can indeed define conjunction, quotient and a maximal environment correctness operator for deterministic MIOs. After discussing normalization of contracts in Section 7.3.2 we study a small example in Section 7.3.3 that illustrates contracts based on deterministic MIOs.

7.3.1 Completeness of $Th^{d\mathbb{MIO}}_{strong}$

In Section 3.3 we have established the specification theory $Th^{d\mathbb{MIO}}_{strong}$ for deterministic MIOs based on strong modal refinement and strong environment correctness. We now show that it is a *complete* specification theory, i.e. the additional specification operators conjunction, quotient and a maximal environment operator can be defined. The assumption of determinism is in fact crucial for the maximality of the operators; see [69] for an example showing that in general no conjunction operator exists for non-deterministic modal transition systems.

Pruning

Before we introduce conjunction and quotient for deterministic MIOs, we have to introduce a pruning operator. When synthesizing new MIOs, it may happen, e.g. when conjoining MIOs, that there is a local inconsistency due to mismatching requirements. For instance, assume two MIOs S_1 and S_2 which describe the behaviour of the same component and they shall be conjoined to form a largest common refinement of both S_1 and S_2. Assume that there is a pair of states

(s_1, s_2) of $S_1 \wedge S_2$ with a must-transition enabled in s_1 labelled with α and there is *no* may-transition enabled in s_2 labelled with α, then there is obviously an inconsistency in the requirements. Thus, this pair of states (s_1, s_2) is marked as inconsistent during the construction of $S_1 \wedge S_2$ and all states, that *necessarily* lead to this state (s_1, s_2) in $S_1 \wedge S_2$ by a sequence of must-transitions are considered inconsistent as well. The pruning ρ takes then a MIO and a subset of inconsistent states and removes all those states and all states that necessarily lead to inconsistent states.

The pruning operator for a deterministic MIO $S = (St_S, s_0, \Sigma_S, \dashrightarrow_S, \longrightarrow_S)$ is formally defined as follows. First, the *must-predecessor* is defined, for a set of states $B \subseteq St_S$, by

$$\mathsf{mustPre}_S(B) \triangleq \{s \in St_S \mid \exists \alpha \in \bigcup \Sigma_S, s' \in B : s \xrightarrow{\alpha}_S s'\}$$

and $\mathsf{mustPre}_S^0(B) \triangleq B$, $\mathsf{mustPre}_S^{j+1}(B) \triangleq \mathsf{mustPre}_S(\mathsf{mustPre}_S^j(B))$ for $j \geq 0$. Given a set $\lightning \subseteq St_S$ of states of S, the pruning of $\lightning$ in S is defined by the tuple

$$(St_S^\rho, s_0, \Sigma_S, \dashrightarrow_\rho, \longrightarrow_\rho)$$

where

$$\begin{aligned}
St_S^\rho &= St_S \setminus \mathsf{mustPre}_S^{|St_S|}(\lightning),\\
\dashrightarrow_\rho &= \dashrightarrow \cap \big(St_S^\rho \times \bigcup \Sigma_S \times St_S^\rho\big), \text{ and}\\
\longrightarrow_\rho &= \longrightarrow \cap \big(St_S^\rho \times \bigcup \Sigma_S \times St_S^\rho\big).
\end{aligned}$$

and where $|St_S|$ denotes the number of states of S.

One can easily observe that $\rho_\lightning(S)$ is a deterministic MIO if and only if $s_0 \notin \mathsf{mustPre}_S^{|St_S|}(\lightning)$. Moreover, if $\rho_\lightning(S)$ is a deterministic MIO, the implementation semantics of S is preserved by $\rho_\lightning(S)$ when one considers only implementations I of S for which there exists a refinement relation $R \subseteq I \times (St_S \setminus \lightning)$.

Conjunction

Conjoining deterministic MIOs is the operation of computing a largest common refinement of two deterministic MIOs with the same action signature. Conjunction of (deterministic) modal transition systems was already defined by Larsen in 1989 [121].

Definition 7.3.1 (Conjunction)
Let $S, T \in d\mathbb{MIO}$ with the same action signature Σ and let $S \wedge^{pre} T$ be the tuple

$$\rho_\lightning\big((St_S \times St_T, (s_0, t_0), \Sigma, \dashrightarrow, \longrightarrow)\big)$$

where the set $↯ \subseteq St_S \times St_T$ *of inconsistent states and the transition relations* $\dashrightarrow$, $\longrightarrow$ *are defined by the following rules:*

$$\frac{s \xrightarrow{\alpha}_S \quad t \not\overset{\alpha}{\dashrightarrow}_T}{(s,t) \in ↯} \qquad \frac{s \not\overset{\alpha}{\dashrightarrow}_S \quad t \xrightarrow{\alpha}_T}{(s,t) \in ↯}$$

$$\frac{s \xrightarrow{\alpha}_S s' \quad t \overset{\alpha}{\dashrightarrow}_T t'}{(s,t) \xrightarrow{\alpha} (s',t')} \qquad \frac{s \overset{\alpha}{\dashrightarrow}_S s' \quad t \xrightarrow{\alpha}_T t'}{(s,t) \xrightarrow{\alpha} (s',t')} \qquad \frac{s \overset{\alpha}{\dashrightarrow}_S s' \quad t \overset{\alpha}{\dashrightarrow}_T t'}{(s,t) \overset{\alpha}{\dashrightarrow} (s',t')}$$

Conjunction $\wedge$ *is defined by*

$$S \wedge T \triangleq \begin{cases} S \wedge^{pre} T & \text{if } S \wedge^{pre} T \text{ is a deterministic MIO,} \\ \text{undefined} & \text{otherwise,} \end{cases}$$

for any $S,T \in d\mathbb{MIO}$.

Example 7.3.2

We illustrate conjunction of deterministic MIOs again with the vending machine. Consider two specifications S *and* T *with the same action signature, shown in Figure 7.7.* S *expresses that after dispensing coffee or tea, the machine must beep before any other action is performed.* T *is a vending machine that must be able to dispense coffee, and after dispensing a drink, the machine beeps. Additionally, the machine allows a coin being inserted twice in a row. The machine then dispenses either coffee or tea, and returns to the state* t_1.

We are interested in whether these two (loose) specifications are consistent in the sense that there exists a common refinement of S *and* T. *Applying the conjunction operator as defined in Definition 7.3.1 results in* $S \wedge T$, *shown in Figure 7.8. The state* (s_1,t_1) *is an "inconsistent" state because* s_1 *requires* beep, *however,* t_1 *does not allow* beep. *Thus, there is no common refinement of this pair of states* (s_1,t_1). $S \wedge T$ *is then the result of pruning all those states in which must-transitions lead to such an "inconsistent" state – in our case, we have to additionally remove the state* (s_0,t_2).

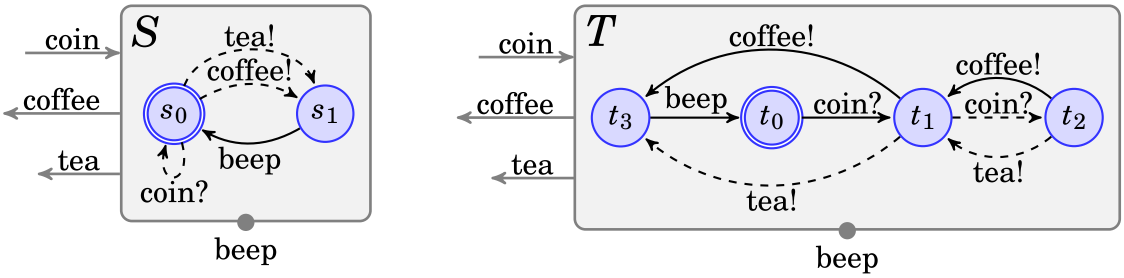


Figure 7.7: Two MIO specifications for a vending machine.

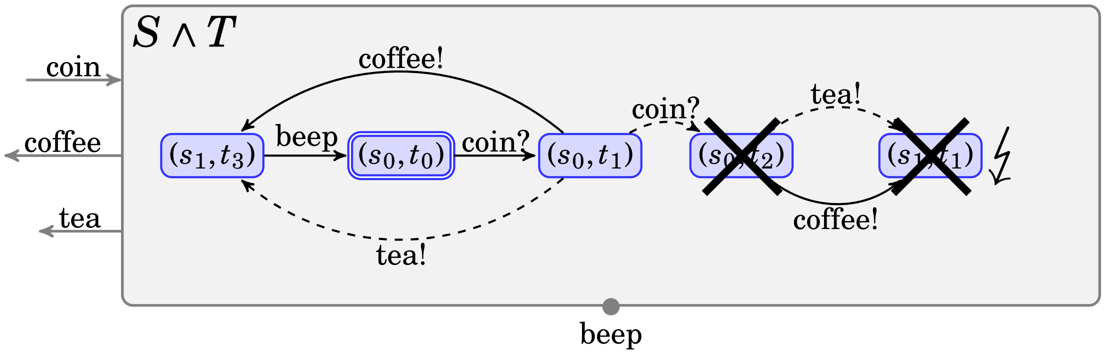


Figure 7.8: The conjunction of S and T from Figure 7.7.

Theorem 7.3.3
Let $S,T \in d\mathbb{MIO}$ with the same action signature Σ.

1. *$S \wedge T$ is defined if and only if there exists $X \in d\mathbb{MIO}$ with action signature Σ such that $X \leq_s S$ and $X \leq_s T$.*

2. *If $S \wedge T$ is defined, then for any $X \in d\mathbb{MIO}$ with action signature Σ, whenever $X \leq_s S$ and $X \leq_s T$, then $X \leq_s S \wedge T$.*

Proof. 1. If $S \wedge T$ is defined, then one can easily show that $S \wedge T \leq_s S$ and $S \wedge T \leq_s T$. For instance, $S \wedge T \leq_s S$ is shown by the refinement relation

$$\{((s,t),s) \mid s \in St_S, t \in St_T\}.$$

For the other direction, assume that $S \wedge T$ is undefined and assume that there is $X \in d\mathbb{MIO}$ such that $X \leq_s S$ and $X \leq_s T$. Then the pruning operator removed the initial state during the construction of $S \wedge T$. But this can only happen if there are must-transitions leading from the initial state to some inconsistent state $(s,t) \in \lightning$ in which w.l.o.g. $s \xrightarrow{\alpha}$ and $t \overset{\alpha}{\not\dashrightarrow}$. As S and T are deterministic, X would have to implement these must-transitions in S and T by a unique sequence of must-transitions in X, leading to a state $x \in St_X$ such that x is related to s and t. This is a contradiction, because x would have to implement $s \xrightarrow{\alpha}$ which is, however, not allowed because of $t \overset{\alpha}{\not\dashrightarrow}$.

2. Assume that $S \wedge T$ is defined, and assume that $X \leq_s S$ and $X \leq_s T$ demonstrated by refinement relations R_1 and R_2, respectively. The refinement $X \leq_s S \wedge T$ is shown by the refinement relation

$$\{(x,(s,t)) \mid (x,s) \in R_1, (x,t) \in R_2\}.$$

Corollary 7.3.4
Let $S,T \in d\mathbb{MIO}$ such that $S \wedge T$ is defined. Then $[\![S \wedge T]\!]_s = [\![S]\!]_s \cap [\![T]\!]_s$.

Quotient

Recall that the quotient operator $/\!\!/$ is adjoint to composition: $T /\!\!/ S$ is a most general solution X for the refinement statement $S \otimes X \leq T$. We first note that the quotient $T /\!\!/ S$ has the signature $\Sigma_T /\!\!/ \Sigma_S$, as explained in the following. Given the action signature Σ_T of T and Σ_S of S, Σ_T is *quotientable by* Σ_S if $(\bigcup \Sigma_S) \subseteq (\bigcup \Sigma_T)$ and $\Sigma_S^{int} \subseteq \Sigma_T^{int}$, $\Sigma_S^{out} \cap \Sigma_T^{ext} \subseteq \Sigma_T^{out}$ and $\Sigma_S^{in} \cap \Sigma_T^{ext} \subseteq \Sigma_T^{in}$.

Definition 7.3.5
Let Σ_T and Σ_S be two action signatures such that Σ_T is quotientable by Σ_S. The quotient of Σ_T by Σ_S *is defined as the action signature $\Sigma_T /\!\!/ \Sigma_S \triangleq (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ where*

$$\begin{aligned}
\Sigma^{in} &= (\Sigma_T^{in} \setminus \Sigma_S^{in}) \cup (\Sigma_S^{out} \setminus \Sigma_T^{out}),\\
\Sigma^{out} &= (\Sigma_T^{out} \setminus \Sigma_S^{out}) \cup (\Sigma_S^{in} \setminus \Sigma_T^{in}),\\
\Sigma^{int} &= \Sigma_T^{int} \setminus \left(\bigcup \Sigma_S\right).
\end{aligned}$$

It is easy to prove that $\Sigma_T /\!\!/ \Sigma_S$ is the unique solution Σ to the statement $\Sigma_S \otimes \Sigma = \Sigma_T$.

By the modal synchronous composition with any shared external action becoming an internal action, the performing of some actions of S is not visible to the quotient $T /\!\!/ S$: every external action of S not shared with $T /\!\!/ S$, and any internal action of S are not visible to T. The state space of the quotient $T /\!\!/ S$ consists, on the one hand, by elements of $\mathscr{P}_{\geq 1}(St_T \times St_S)$, which contains all the state pairs (t, s) with the meaning that T is currently in state t and S is in state s. Note that, since some actions of S are not visible for the quotient $T /\!\!/ S$, any reachable state of s' from s with actions not visible to $T /\!\!/ S$ must be taken into account as well, and any state in $\mathscr{P}_{\geq 1}(St_T \times St_S)$ must be closed with respect to this reachability property. On the other hand, the state space of the quotient $T /\!\!/ S$ contains a distinguished *universal state* univ in which the MIO can show arbitrary behaviour. The universal state is needed to achieve maximality of the quotient.

Example 7.3.6
Consider an example of a verification component T, that can receive a request by action check*, then performs an internal verification, action* verify*, then can either do the external action* ok *or* notOk*, and then goes back to the initial state with the internal action* ready*. Assume that we are given a component S, that already partially implements the component T: when receiving a new check request, then, it forwards this request to another component, and then goes back to the initial state with* ready*. The question is now whether there exists a component $T /\!\!/ S$ that, when composed with S, realizes (more precisely, refines) the given specification T. As expected, the quotient $T /\!\!/ S$ waits for a verification request, then answers with*

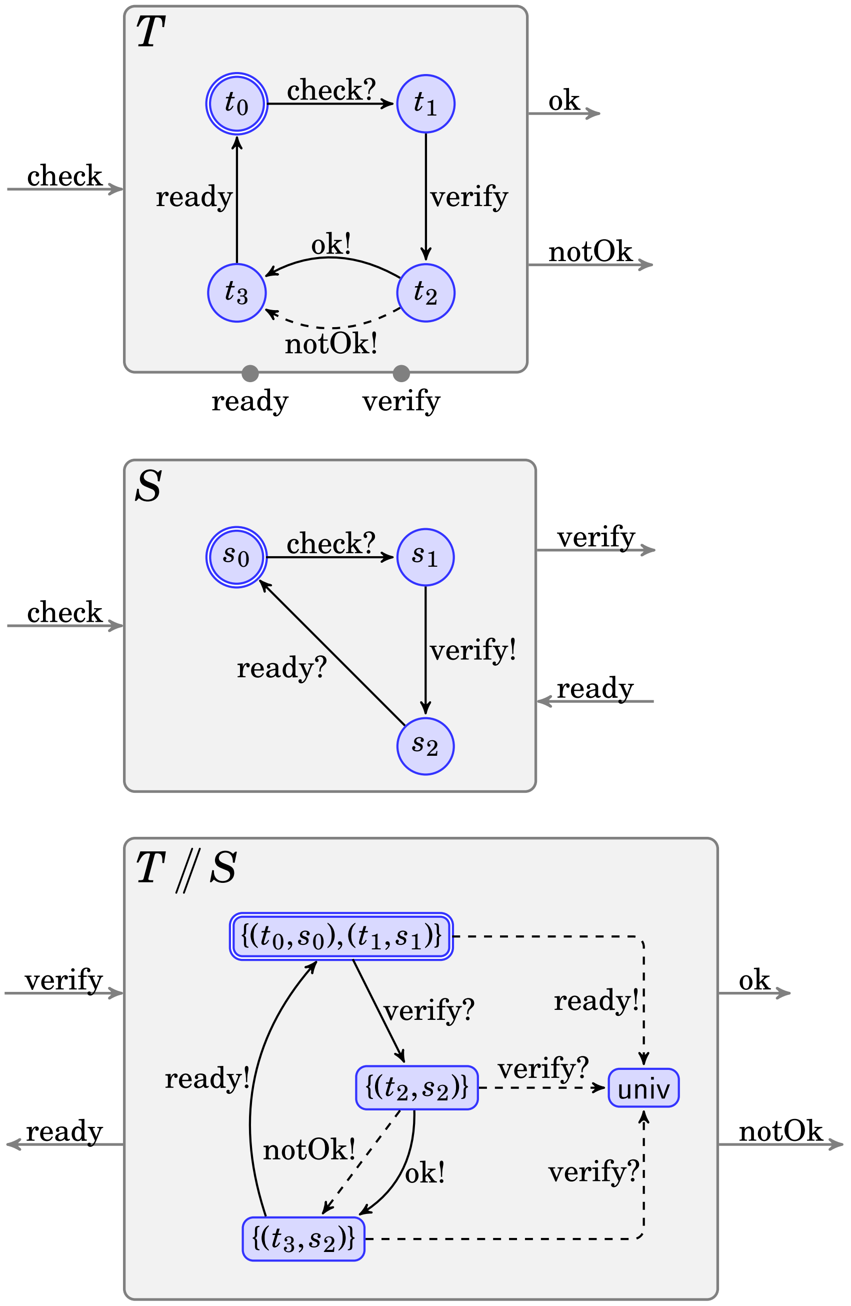


Figure 7.9: A simple example for the quotient operation

ok *or* notOk *and then synchronizes with S on the action* ready *to go back to the initial state. The additional may-transitions to the universal state* univ *indicate which transitions are additionally allowed (and which will not be synchronized when composing with S). In the following we show that the quotient operator* $/\!\!/$ *can be defined such that* $T /\!\!/ S$ *is a most general specification* X *that solves the problem* $S \otimes X \leq_s T$.

Definition 7.3.7 (Quotient)
Let $S, T \in d\mathbb{MIO}$ *such that* Σ_T *is quotientable by* Σ_S. *Let* $T /\!\!/^{pre} S$ *be the tuple*

$$\rho_{\lightning}\big((\mathscr{P}_{\geq 1}(St_T \times St_S) \cup \{\mathsf{univ}\}, \mathscr{R}(t_0, s_0), \Sigma_T /\!\!/ \Sigma_S, \dashrightarrow, \longrightarrow)\big)$$

in which, for any states $t \in St_T$, $s \in St_S$,

$$\mathscr{R}(t,s) \triangleq \{(t',s') \mid \exists \sigma \in (\bigcup \Sigma_T \setminus \bigcup(\Sigma_T /\!\!/ \Sigma_S))^* : t \overset{\sigma}{\dashrightarrow}_T t' \text{ and } s \overset{\sigma}{\dashrightarrow}_S s'\},$$

and in which the set of inconsistent states $\lightning \subseteq \mathscr{P}_{\geq 1}(St_T \times St_S)$ *is defined as follows: A set* $q \in \mathscr{P}_{\geq 1}(St_T \times St_S)$ *is in* $\lightning$ *if at least one of the following conditions is satisfied:*

1. *Let* $\alpha \in \bigcup \Sigma_S, \alpha \notin \bigcup(\Sigma_T /\!\!/ \Sigma_S)$.

   (a) $\exists (t,s) \in q : t \xrightarrow{\alpha}_T$ *and* $s \not\xrightarrow{\alpha}_S$,

   (b) $\exists (t,s) \in q : t \not\overset{\alpha}{\dashrightarrow}_T$ *and* $s \overset{\alpha}{\dashrightarrow}_S$,

2. *Let* $\alpha \notin \bigcup \Sigma_S, \alpha \in \bigcup(\Sigma_T /\!\!/ \Sigma_S)$.

   $\exists (t,s),(t',s') \in q : t \xrightarrow{\alpha}_T$ *and* $t' \not\overset{\alpha}{\dashrightarrow}_T$,

3. *Let* $\alpha \in \bigcup \Sigma_S, \alpha \in \bigcup(\Sigma_T /\!\!/ \Sigma_S)$.

   (a) $\exists (t,s) \in q : t \xrightarrow{\alpha}_T$ *and* $s \not\xrightarrow{\alpha}_S$,

   (b) $\exists (t,s),(t',s') \in q : t \xrightarrow{\alpha}_T$ *and* $s' \overset{\alpha}{\dashrightarrow}_S$ *and* $t' \not\overset{\alpha}{\dashrightarrow}_T$.

Moreover, the transition relations $\dashrightarrow, \longrightarrow$ *are defined by the following rules:*
Let $q \in \mathscr{P}_{\geq 1}(St_T \times St_S) \setminus \lightning$.

1. *For any* $\alpha \notin \bigcup \Sigma_S, \alpha \in \bigcup(\Sigma_T /\!\!/ \Sigma_S)$:

$$\frac{\exists (t,s) \in q : t \xrightarrow{\alpha}_T}{q \xrightarrow{\alpha} \bigcup\{\mathscr{R}(t',s) \mid \exists t \in T : (t,s) \in q, t \overset{\alpha}{\dashrightarrow}_T t'\}}$$

$$\frac{\forall (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T}{q \overset{\alpha}{\dashrightarrow} \bigcup\{\mathscr{R}(t',s) \mid \exists t \in T : (t,s) \in q, t \overset{\alpha}{\dashrightarrow}_T t'\}}$$

2. *For any $\alpha \in \bigcup \Sigma_S, \alpha \in \bigcup(\Sigma_T /\!\!/ \Sigma_S)$:*

$$\frac{\exists (t,s) \in q : t \xrightarrow{\alpha}_T}{q \xrightarrow{\alpha} \bigcup\{\mathscr{R}(t',s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}}$$

$$\frac{\exists (t,s) \in q : s \overset{\alpha}{\dashrightarrow}_S \qquad \forall (t,s) \in q : (s \overset{\alpha}{\dashrightarrow}_S \textit{ implies } t \overset{\alpha}{\dashrightarrow}_T)}{q \overset{\alpha}{\dashrightarrow} \bigcup\{\mathscr{R}(t',s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}}$$

$$\frac{\forall (t,s) \in q : s \overset{\alpha}{\not\dashrightarrow}_S}{q \overset{\alpha}{\dashrightarrow} \mathsf{univ}}$$

3. *For any $\alpha \in \bigcup(\Sigma_T /\!\!/ \Sigma_S)$:*

$$\frac{}{\mathsf{univ} \overset{\alpha}{\dashrightarrow} \mathsf{univ}}$$

The quotient operator $/\!\!/$ is defined by

$$T /\!\!/ S \triangleq \begin{cases} T /\!\!/^{pre} S & \textit{if } \Sigma_T \textit{ is quotientable by } \Sigma_S \textit{ and } T /\!\!/^{pre} S \in d\mathbb{MIO}, \\ \textit{undefined} & \textit{otherwise,} \end{cases}$$

for any $S, T \in d\mathbb{MIO}$.

The set of inconsistent states $\lightning$ contains those states $q \in \mathscr{P}_{\geq 1}(St_T \times St_S)$ for which there is a unsatisfiable requirement: either there is some behaviour of S that cannot be interfered by $T /\!\!/ S$ and does not refine T (conditions in 1.), or there are two states of T in q with a requirement for $T /\!\!/ S$ that are contradicting (condition 2.), or there is a requirement in T for an action shared between S and $T /\!\!/ S$ that is not satisfied by any state s that is possible according to q (conditions in 3.).

For the transition relations, note that the target states contain all those states that are reachable by the respective action, but must also contain states that S may perform without synchronizing with $T /\!\!/ S$. Also note the rule in (2.) that introduces transitions to the universal state univ: whenever a shared action $\alpha \in (\bigcup \Sigma_S) \cap (\bigcup(\Sigma_T /\!\!/ \Sigma_S))$ is not enabled in any possible state $s \in St_S$ (according to q), then this action will never synchronize and hence the quotient is allowed to show arbitrary behaviour.

Observe that, to ensure that the above definition is well-defined, one has to prove that the intermediate transition system in the above construction, before pruning is applied, is indeed a proper deterministic MIO, i.e. satisfying $\longrightarrow \subseteq \dashrightarrow$. This can be seen as follows: Let $q \in \mathscr{P}_{\geq 1}(St_T \times St_S)$, and let $\alpha \notin \Sigma_S$, then

$$\exists (t,s) \in q : t \xrightarrow{\alpha}_T \quad \text{implies} \quad \forall (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T$$

because of inconsistency rule (2) in Definition 7.3.7. Similarly, the consistency of transitions involving actions $\alpha \in \bigcup \Sigma_S$ is also satisfied: Assume $(t,s) \in q$ and $t \xrightarrow{\alpha}_T$, then by rule (3a) we know that $s \xrightarrow{\alpha}_S$, hence $\exists (t,s) \in q : s \overset{\alpha}{\dashrightarrow}_S$. Finally, from rule (3b),

$$\forall (t,s) \in q : (s \overset{\alpha}{\dashrightarrow}_S \text{ implies } t \overset{\alpha}{\dashrightarrow}_T)$$

follows.

The following theorem shows that the quotient operator $/\!\!/$ satisfies the abstract properties of a quotient defined in Section 7.1.

Theorem 7.3.8
Let $S, T \in d\mathbb{MIO}$.

1. *$T /\!\!/ S$ is defined if and only if there exists $X \in d\mathbb{MIO}$ such that $S \otimes X \leq_s T$.*
2. *If $T /\!\!/ S$ is defined, then $S \otimes (T /\!\!/ S) \leq_s T$ and for any $X \in d\mathbb{MIO}$, if $S \otimes X \leq_s T$, then $X \leq_s T /\!\!/ S$.*

Proof. 1. The implication from left to right is subsumed by the second statement. For the other direction, assume that there exists $X \in d\mathbb{MIO}$ such that $S \otimes X \leq_s T$. Assume that pruning removes the initial state $\mathscr{R}(t_0, s_0)$. This is the case if and only if there is a sequence of must-transitions in the "intermediate" (unpruned) tuple of the quotient construction that lead to an inconsistent state in $\lightning$. In other words, there are must-transitions leading to a state that cannot be realized such that, when composed with S, refines T. This contradicts with the fact that X is deterministic and solves the equation, as X would have to implement all the must-transitions, necessarily leading to this inconsistent state because of determinism. Thus, pruning does not remove the initial state $\mathscr{R}(t_0, s_0)$ and $T /\!\!/ S$ is defined. Note that by construction of the transition relations, $T /\!\!/ S$ is deterministic.

2. We first show that $S \otimes (T /\!\!/ S) \leq_s T$. We define a relation R by

$$R = \{((s,q),t) \in (St_S \times \mathscr{P}_{\geq 1}(St_T \times St_S)) \times St_T \mid (t,s) \in q\},$$

and we show that R demonstrates $S \otimes (T /\!\!/ S) \leq_s T$. Obviously, it holds that $((s_0, \mathscr{R}(t_0, s_0)), t_0) \in R$.

(a) Assume $(s,q) \overset{\alpha}{\dashrightarrow}_{S \otimes (T /\!\!/ S)} (s', q')$.

- If $\alpha \in \bigcup \Sigma_S$, $\alpha \notin \bigcup (\Sigma_T /\!\!/ \Sigma_S)$, then $s \overset{\alpha}{\dashrightarrow}_S s'$ and $q = q'$. We know that $(t,s) \in q$, and since $q \notin \lightning$, we have $t \overset{\alpha}{\dashrightarrow}_T t'$, and $(t', s') \in q$ by definition. Hence $((s', q), t') \in R$.

- If $\alpha \notin \bigcup \Sigma_S$, $\alpha \in \bigcup(\Sigma_T \sslash \Sigma_S)$, then $s = s'$ and $q \overset{\alpha}{\dashrightarrow}_{T\sslash S} q'$ where

  $$q' = \bigcup \{\mathscr{R}(t', s) \mid \exists t \in St_T : (t,s) \in q, t \overset{\alpha}{\dashrightarrow}_T t'\}.$$

  For this transition in $T \sslash S$ to be present, we must have $\forall (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T$. Hence there exists $t \overset{\alpha}{\dashrightarrow}_T t'$ such that $(t', s) \in q'$ by definition. It follows that $((s, q'), t') \in R$.
- If $\alpha \in \bigcup \Sigma_S$, $\alpha \in \bigcup(\Sigma_T \sslash \Sigma_S)$, then we have $s \overset{\alpha}{\dashrightarrow}_S s'$ and $q \overset{\alpha}{\dashrightarrow}_{T\sslash S} q'$ where

  $$q' = \bigcup \{\mathscr{R}(t', s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}.$$

  By the precondition of the (unique) rule which generates this transition, we know that there is $t \overset{\alpha}{\dashrightarrow}_T t'$ and hence $(t', s') \in q'$ and $((s', q'), t') \in R$.

(b) Assume $t \xrightarrow{\alpha}_T t'$.

- If $\alpha \in \bigcup \Sigma_S$, $\alpha \notin \bigcup(\Sigma_T \sslash \Sigma_S)$, then $s \xrightarrow{\alpha}_S s'$ because of $(t,s) \in q \notin \lightning$. Hence $(t', s') \in q$ and $((s', q), t') \in R$.
- If $\alpha \notin \bigcup \Sigma_S$, $\alpha \in \bigcup(\Sigma_T \sslash \Sigma_S)$, then from $q \notin \lightning$ it follows that for all $(\dot{t}, \dot{s}) \in q$, $\dot{t} \overset{\alpha}{\dashrightarrow}_T$. Hence there exists $q \xrightarrow{\alpha}_{T\sslash S} q'$ with

  $$q' = \bigcup \{\mathscr{R}(t', s) \mid \exists t \in T : (t,s) \in q, t \overset{\alpha}{\dashrightarrow}_T t'\}.$$

  Thus $(t', s) \in q'$, and $((s, q'), t') \in R$.
- If $\alpha \in \bigcup \Sigma_S$, $\alpha \in \bigcup(\Sigma_T \sslash \Sigma_S)$, then from $(t,s) \in q \notin \lightning$ we can infer that $s \xrightarrow{\alpha}_S s'$ and $q \xrightarrow{\alpha}_{T\sslash S} q'$ with

  $$q' = \bigcup \{\mathscr{R}(t', s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}.$$

  Hence $(t', s') \in q'$ and $((s', q'), t') \in R$.

We now show the second claim. Let S, X, and T as above, and assume that $S \otimes X \le_s T$ which is witnessed by a relation R. Then we define a relation R' by

$$\begin{aligned} R' = &\{(x,q) \in St_X \times \mathscr{P}_{\ge 1}(St_T \times St_S) \mid \forall (t,s) \in q : ((s,x),t) \in R\} \\ &\cup \{(x, \mathsf{univ}) \mid x \in St_X\} \end{aligned}$$

where univ is the unique universal state in the construction of $T \sslash S$. We show that R' demonstrates $X \le_s T \sslash S$. Obviously, it holds that

$$(x_0, \mathscr{R}(t_0, s_0)) \in R',$$

because $((s_0,x_0),t_0) \in R$, and for any state $(t,s) \in \mathscr{R}(t_0,s_0)$, (t,s) is reachable from (t_0,s_0) be actions in $\bigcup\Sigma_S \setminus \bigcup(\Sigma_T \,/\!\!/\, \Sigma_S)$, respectively. Since these transitions are interleaved in the deterministic MIO $S \otimes X$ and $((s_0,x_0),t_0) \in R$, we know that $((s,x_0),t) \in R$.

Now, let $(x,q) \in R'$.

(a) Assume $x \overset{\alpha}{\dashrightarrow}_X x'$.

- If $\alpha \in \bigcup(\Sigma_T \,/\!\!/\, \Sigma_S)$, $\alpha \notin \bigcup\Sigma_S$, then $t \overset{\alpha}{\dashrightarrow}_T$ for all $(t,s) \in q$. Hence
$$q \overset{\alpha}{\dashrightarrow} \bigcup\{\mathscr{R}(t',s) \mid \exists t \in St_T : (t,s) \in q, t \overset{\alpha}{\dashrightarrow}_T t'\}.$$
We still have to show that for all $(t'',s'') \in \bigcup\{\mathscr{R}(t',s) \mid \exists t \in St_T : (t,s) \in q, t \overset{\alpha}{\dashrightarrow}_T t'\}$, it holds $((s'',x'),t'') \in R$. By assumption, we know that $((s,x'),t') \in R$. Then any path with action sequence $\sigma \in (\bigcup\Sigma_T \setminus \bigcup(\Sigma_T /\!\!/ \Sigma_S))^*$, starting from s, must be matched with the unique path with the same action sequence σ, hence $((s'',x'),t'') \in R$.
- If $\alpha \in \bigcup(\Sigma_T \,/\!\!/\, \Sigma_S)$, $\alpha \in \bigcup\Sigma_S$, then we distinguish cases: If for all $(t,s) \in q$, $s \overset{\alpha}{\not\dashrightarrow}_S$, then $q \overset{\alpha}{\dashrightarrow} \mathsf{univ}$ and $(x,\mathsf{univ}) \in R'$. Assume now that there exists $(t,s) \in q$ such that $s \overset{\alpha}{\dashrightarrow}_S$. We can infer that for every $(t,s) \in q$, if $s \overset{\alpha}{\dashrightarrow}_S$ then $t \overset{\alpha}{\dashrightarrow}_T$. Thus, by the rules of the transition relation of the quotient, we get
$$q \overset{\alpha}{\dashrightarrow} \bigcup\{\mathscr{R}(t',s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}.$$
For every $(t'',s'') \in \bigcup\{\mathscr{R}(t',s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}$ we know that $((s'',x'),t'') \in R$, by the same argument as above, hence
$$(x', \bigcup\{\mathscr{R}(t',s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}) \in R'.$$

(b) Assume $q \xrightarrow{\alpha}_{T/\!\!/S} q'$.

- If $\alpha \in \bigcup(\Sigma_T \,/\!\!/\, \Sigma_S)$, $\alpha \notin \bigcup\Sigma_S$, then
$$q' = \bigcup\{\mathscr{R}(t',s) \mid \exists t \in St_T : (t,s) \in q, t \overset{\alpha}{\dashrightarrow}_T t'\}$$
and there exists $(t,s) \in q$ such that $t \xrightarrow{\alpha}_T t'$. Since $((s,x),t) \in R$, we know that $x \xrightarrow{\alpha}_X x'$ such that $((s,x'),t') \in R$. By the same argumentation as above, we get that $((s'',x'),t'') \in R$ for all $(t'',s'') \in q'$. Hence
$$(x',q') \in R'.$$

- If $\alpha \in \bigcup(\Sigma_T /\!\!/ \Sigma_S)$, $\alpha \in \bigcup \Sigma_S$, then

$$q' = \bigcup\{\mathscr{R}(t', s') \mid \exists (t,s) \in q : t \overset{\alpha}{\dashrightarrow}_T t', s \overset{\alpha}{\dashrightarrow}_S s'\}$$

and there is $(t,s) \in q$ such that $t \xrightarrow{\alpha}_T t'$. It follows that there exists $x \xrightarrow{\alpha}_X x'$ and $s \xrightarrow{\alpha}_S s'$ such that $((s',x'),t') \in R$. Again, $((s'',x'),t'') \in R$ for all $(t'',s'') \in \mathscr{R}(t',s')$, hence

$$(x', q') \in R'. \qquad \square$$

Maximal Environment Operator

We already defined a quotient and a conjunction operator for deterministic MIOs with strong modal refinement, so only a maximal environment operator for strong environment correctness is missing to achieve a complete specification theory.

We first define the maximal correct environment $\mathsf{max}_{S \rightarrow_s}(\Sigma)$ for a given action signature Σ and specification S.

$$\mathsf{max}_{S \rightarrow_s}(\Sigma) \triangleq (\mathscr{P}_{\geq 1}(S) \cup \{\mathsf{univ}\}, \mathscr{R}(s_0), \Sigma, \dashrightarrow, \longrightarrow)$$

in which for all states $t \in T$,

$$\mathscr{R}(s) \triangleq \left\{ s' \in S \mid \exists \sigma \in (\bigcup \Sigma_S \setminus \bigcup \Sigma)^* : s \overset{\sigma}{\dashrightarrow}_S s' \right\},$$

and the transition relations $\dashrightarrow$, $\rightarrow$ are defined by the following rules:

$$\frac{\exists s \in q : s \overset{\alpha!}{\dashrightarrow}_S}{\begin{array}{c} q \xrightarrow{\alpha?} \bigcup\{\mathscr{R}(s') \mid \exists s \in q : s \overset{\alpha!}{\dashrightarrow}_S s'\} \\ q \overset{\alpha?}{\dashrightarrow} \bigcup\{\mathscr{R}(s') \mid \exists s \in q : s \overset{\alpha!}{\dashrightarrow}_S s'\} \end{array}}(1) \qquad \frac{\forall s \in q : s \overset{\alpha!}{\not\dashrightarrow}_S}{q \overset{\alpha?}{\dashrightarrow} \mathsf{univ}}(2)$$

$$\frac{\exists s \in q : s \overset{\alpha?}{\dashrightarrow}_S}{q \overset{\alpha!}{\dashrightarrow} \bigcup\{\mathscr{R}(s') \mid \exists s \in q : s \overset{\alpha!}{\dashrightarrow}_S s'\}}(3) \qquad \frac{\forall s \in q : s \overset{\alpha?}{\not\dashrightarrow}_S}{q \overset{\alpha!}{\dashrightarrow} \mathsf{univ}}(4)$$

$$\frac{\alpha \in \bigcup \Sigma \quad \alpha \notin \bigcup \Sigma_S}{q \overset{\alpha}{\dashrightarrow} q}(5) \qquad \frac{\alpha \in \bigcup \Sigma}{\mathsf{univ} \overset{\alpha}{\dashrightarrow} \mathsf{univ}}(6)$$

Lemma 7.3.9
Let $S \in d\mathbb{MIO}$ and Σ be an action signature such that Σ_S and Σ are composable. Then $S \rightarrow_s \mathsf{max}_{S \rightarrow_s}(\Sigma)$, and for all $E \in d\mathbb{MIO}$ with action signature Σ, if $S \rightarrow_s E$, then $E \leq_s \mathsf{max}_{S \rightarrow_s}(\Sigma)$.

Proof. We first show that $S \rightarrow_s \mathsf{max}_{S\rightarrow_s}(\Sigma)$. Observe that for any reachable state (s,q) in $S \otimes \mathsf{max}_{S\rightarrow_s}(\Sigma)$ we have that $s \in q$. Assume that $s \overset{\alpha!}{\dashrightarrow}_S$ with $\alpha \in \Sigma^{out}_S \cap \Sigma^{in}$. Then, by rule (1) in Definition 7.3.10, we know that $q \xrightarrow{\alpha?}$ in $\mathsf{max}_{S\rightarrow_s}(\Sigma)$.

To prove the second claim, let $E \in d\mathbb{MIO}$ with action signature Σ and assume that $S \rightarrow_s E$. $E \leq_s \mathsf{max}_{S\rightarrow_s}(\Sigma)$ is proven by the relation

$$R = \{(e,q) \in St_E \times \mathscr{P}_{\geq 1}(St_S) \mid \forall s \in St_S : s \in q \text{ iff } (s,e) \text{ is reachable in } S \otimes E\} \\ \cup \{(s,\mathsf{univ}) \mid s \in St_S\}.$$

Clearly, we have that $(e_0,\mathscr{R}(s_0)) \in R$, because any state in $\mathscr{R}(s_0)$ is reachable from the initial state with actions not in Σ. Let $(e,q) \in R$. The only interesting case is to consider $q \xrightarrow{\alpha?} q'$ in $\mathsf{max}_{S\rightarrow_s}(\Sigma)$. Then, by rule (1) there must exist $s \in q$ such that $s \overset{\alpha!}{\dashrightarrow}_S s'$. Since $S \rightarrow_s E$, we must have $e \xrightarrow{\alpha?}_E e'$. By definition of q', we know that $(e',q') \in R$.

Definition 7.3.10 (Maximal environment operator)
The maximal environment operator for strong environment correctness is defined by

$$\mathsf{max}_{S\rightarrow_s}(E) \triangleq \begin{cases} E \wedge \mathsf{max}_{S\rightarrow_s}(\Sigma_E) & \text{if } S \otimes E \text{ and } E \wedge \mathsf{max}_{S\rightarrow_s}(\Sigma_E) \text{ are defined,} \\ \text{undefined} & \text{otherwise,} \end{cases}$$

for any $S,E \in d\mathbb{MIO}$.

The maximal environment operator w.r.t. strong environment correctness, as defined above, satisfies the conditions of a maximal environment operator from Section 7.1.

Theorem 7.3.11
Let $S,E \in d\mathbb{MIO}$.

1. $\mathsf{max}_{S\rightarrow_s}(E)$ *is defined if and only if there exists $E' \in d\mathbb{MIO}$ such that $E' \leq_s E$ and $S \rightarrow_s E'$.*

2. *If* $\mathsf{max}_{S\rightarrow_s}(E)$ *is defined, then* $\mathsf{max}_{S\rightarrow_s}(E) \leq_s E$, $S \rightarrow_s \mathsf{max}_{S\rightarrow_s}(E)$*, and for all $E' \in d\mathbb{MIO}$, if $E' \leq_s E$ and $S \rightarrow_s E'$, then $E' \leq_s \mathsf{max}_{S\rightarrow_s}(E)$.*

Proof. 1. Assume that $\mathsf{max}_{S\rightarrow_s}(E)$ is defined which is $E \wedge \mathsf{max}_{S\rightarrow_s}(\Sigma_E)$ by definition. Then we can take $E \wedge \mathsf{max}_{S\rightarrow_s}(\Sigma_E)$ as the E' that we are looking for. Since $\mathsf{max}_{S\rightarrow_s}(\Sigma_E)$ is a strongly correct environment for S (see Lemma 7.3.9), also $E \wedge \mathsf{max}_{S\rightarrow_s}(\Sigma_E)$ is a strongly correct environment for S by preservation of strong environment correctness. The refinement $E \wedge \mathsf{max}_{S\rightarrow_s}(\Sigma_E) \leq_s E$ follows from the properties of conjunction.

For the other direction assume $E' \in d\mathbb{MIO}$ such that $E' \leq_s E$ and $S \rightarrow_s E'$. Thus we know that $E' \leq_s \mathsf{max}_{S\rightarrow_s}(\Sigma_E)$ as well as $E' \leq_s E$. Hence $E' \leq_s \mathsf{max}_{S\rightarrow_s}(E)$ because conjunction is a greatest lower bound with respect to strong modal refinement.

2. This follows from very similar arguments as in the previous paragraph. □

Summing Up: A Complete Specification Theory for MIOs

Finally, we obtain a complete specification theory for deterministic MIOs.

Corollary 7.3.12
$Th^{d\mathbb{MIO}}_{strong}$ is a complete specification theory with quotient $/\!/$, conjunction $\wedge$ and the maximal environment operator $\mathsf{max}_{\longrightarrow_s}(\cdot)$ as defined above.

7.3.2 Modal Contracts

We assume a fixed global set *Act* of action labels. Such a fixed global set of action labels is needed because the assumption of a contract must encompass all actions of possible environments, i.e. all actions of contracts that are later considered for composition must be known in advance. This is caused by the fact that we use strong modal refinement which does not allow new internal actions in a refinement hence an assumption in a contract must already take into account all possible internal actions of any environments.

A *modal contract* is a pair (A,G) with $A,G \in d\mathbb{MIO}$ such that $G \rightarrow_s A$. We restrict modal contracts to those pairs (A,G) for which $\Sigma_A \otimes \Sigma_G = \Sigma_{Act}$ where Σ_{Act} is the action signature determined by $\Sigma^{in}_{Act} = \Sigma^{out}_{Act} = \emptyset$ and $\Sigma^{int}_{Act} = Act$. Note that due to this signature restriction and composability of A and G the external actions of A and G must be complementary, i.e. $\Sigma^{in}_A = \Sigma^{out}_G$ and $\Sigma^{out}_A = \Sigma^{in}_G$. Although our generic constructions in the previous section does not allow such restrictions on the signatures (because signatures of specifications are not considered), it is straightforward to see that contract composition will be closed under this property.

Before we go on with discussing semantics of modal contracts, we would like to stress that MIOs are very suitable for expressing assumptions. May-transitions allow to specify loose assumptions leaving open many possible design choices for the environment. Furthermore, must-transition can express local liveness properties that go beyond specifications based on single MIOs: with a modal contract, we can specify that an input of the guarantee G must be served by the environment by putting a must-transition with the corresponding output in A. Examples of modal contracts with MIOs are shown in the next section.

In order to provide a characterization of relativized refinement – written $S \leq_{s,E} T$ in the following – and hence in particular a characterization of the behaviour semantics of a contract, we define a weakening operator that is used to obtain the normal form of a contract. The weakening of a guarantee G by an assumption A, written $A \rhd G$, weakens G by taking into account A by modifying or adding, for each input action $\beta \in \Sigma^{in}_G$, may-transitions labelled with β to the universal state in G whenever A is not allowed to perform $\beta \in \Sigma^{out}_A$. Note that such a weakening is done only for input actions of G, not for output actions of G, because we want to have environment correctness $G \rightarrow_s A$ preserved. Formally, the weakening operator is defined as follows.

Definition 7.3.13 (Weakening)
Let $A, G \in d\mathbb{MIO}$ such that A and G are composable and $\Sigma^{in}_A = \Sigma^{out}_G$, $\Sigma^{out}_A = \Sigma^{in}_G$. The weakening of G by A *is defined by*

$$A \rhd G \triangleq ((\mathscr{P}_{\geq 1}(A) \times G) \cup \{\mathsf{univ}\}, (\mathscr{R}(a_0), g_0), \Sigma_G, \dashrightarrow, \longrightarrow)$$

where, for any $a \in A$,

$$\mathscr{R}(a) \triangleq \{a' \in A \mid a \overset{\tau^*}{\dashrightarrow}_A a'\}$$

and $\dashrightarrow, \longrightarrow$ are defined as the smallest relations satisfying the following rules:
for $\beta \in \Sigma^{int}_G$:

$$\frac{g \overset{\beta}{\dashrightarrow}_G g'}{(q,g) \overset{\beta}{\dashrightarrow} (q,g')}(1) \qquad \frac{g \xrightarrow{\beta}_G g'}{(q,g) \xrightarrow{\beta} (q,g')}(2)$$

for $\beta \in \Sigma^{ext}_G$:

$$\frac{\exists a \in q : a \overset{\beta}{\dashrightarrow}_A \quad g \overset{\beta}{\dashrightarrow}_G g'}{(q,g) \overset{\beta}{\dashrightarrow} (\bigcup\{\mathscr{R}(a') \mid \exists a \in q : a \overset{\beta}{\dashrightarrow}_A a'\}, g')}(3)$$

$$\frac{\exists a \in q : a \overset{\beta}{\dashrightarrow}_A \quad g \xrightarrow{\beta}_G g'}{(q,g) \xrightarrow{\beta} (\bigcup\{\mathscr{R}(a') \mid \exists a \in q : a \overset{\beta}{\dashrightarrow}_A a'\}, g')}(4)$$

$$\frac{\beta \in \Sigma^{in}_G \quad \forall a \in q : a \overset{\beta}{\not\dashrightarrow}_A}{(q,g) \overset{\beta}{\dashrightarrow} \mathsf{univ}}(5)$$

for $\beta \in \bigcup \Sigma_G$:

$$\frac{\beta \in \bigcup \Sigma_G}{\mathsf{univ} \overset{\beta}{\dashrightarrow} \mathsf{univ}}(6)$$

The weakening operator adds to every state in G a set of states of A in which A can possibly be in when G is composed with A. Rules (1) and (2) preserve internal transitions in G, rules (3) and (4) preserve also the external transitions in G provided that A can possibly perform this action as well. Rule (5) adds an input transition to the universal state whenever A cannot perform this action. Finally, rule (6) makes sure that in the universal state all actions are allowed, i.e. in the universal state $A \rhd G$ may show arbitrary behaviour.

The properties of the weakening operator are summarised in the following lemma.

Lemma 7.3.14
Let (A,G) be a modal contract. Then the following are satisfied:

(a) $A \rhd (A \rhd G) = A \rhd G$,

(b) $A \otimes G = A \otimes (A \rhd G)$,

(c) $A \rhd G \rightarrow_s A$.

(d) $A \rhd G = A^{max} \rhd G$, *with* A^{max} *is the maximal implementation of* A *with the transition relations defined by* $\rightarrow_{\tilde{A}} \triangleq \dashrightarrow_{\tilde{A}} \triangleq \dashrightarrow_A$.

Proof. We first observe that $q \in \mathscr{P}_{\geq 1}(A)$ in any reachable state (q,g) in $A \rhd G$ contains exactly all those states $a \in A$ for which (a,g) is a reachable state in $A \otimes G$.

(a) This is an immediate consequence from the observation above: $A \rhd G$ already takes into account all reachable states in A, and in the second weakening $A \rhd (A \rhd G)$ no input-transitions are added or modified; all states in $A \rhd (A \rhd G)$ are of the form $(q,(q,g))$ or univ.

(b) This is again an easy consequence of the fact that G and $A \rhd G$ only differ in input transitions that are not synchronized when composed with A, hence do not appear in the composition.

(c) Let $((q,g),a)$ be a reachable state in $(A \rhd G) \otimes A$ – note that (univ,a) cannot be a reachable state. Reachability of $((q,g),a)$ implies reachability of (g,a) in $G \otimes A$, in particular $a \in q$. If $(q,g) \overset{\beta!}{\dashrightarrow}_{A \rhd G}$, then also $g \overset{\beta!}{\dashrightarrow}_G$ by rule (3), hence $a \xrightarrow{\beta?}_A$ by $G \rightarrow_s A$.

(d) The construction of $A \rhd G$ only refers to the may-transition relations which can be easily seen when looking at the rules (1) – (6). □

Theorem 7.3.15
Let (A,G) be a modal contract. Then, for any $S \in d\mathbb{MIO}$, $S \leq_{s,A} G$ if and only if $S \leq_s A \rhd G$.

Proof. $\Rightarrow$: We assume $S \leq_{s,A} G$ and show $S \leq_s A \rhd G$. Let A^{max} be the maximal implementation of A (see Lemma 7.3.14). We know that $G \rightarrow_s A$, and since $A^{max} \leq_s A$, by Lemma 7.2.4 we can infer that $S \rightarrow_s A^{max}$ and $A^{max} \otimes S \leq_s A^{max} \otimes G$ which is demonstrated by a refinement relation R. Let us define a relation $Q \subseteq St_S \times (\mathscr{P}_{\geq 1}(St_{A^{max}}) \times St_G)$ by

$$Q = \{(s,(q,g)) \mid \forall a \in q : ((a,s),(a,g)) \in R\} \cup \{(s,\mathsf{univ}) \mid s \in St_S\}.$$

We show that Q proves $S \leq_s (A^{max} \rhd G)$, and since $A^{max} \rhd G = A \rhd G$ is satisfied by Lemma 7.3.14, we get $S \leq_s (A \rhd G)$.

First, we have to show that $(s_0,(\mathscr{R}(a_0),g_0)) \in Q$. We know that

$$((a_0,s_0),(a_0,g_0)) \in R,$$

and every state $a \in \mathscr{R}(a_0)$ is by definition reachable by internal transitions in A^{max}; it follows that (a,s_0) is reachable by internal transitions which are simulated by (a_0,g_0) to (a,g_0) such that $((a,s_0),(a,g_0)) \in R$. Thus $(s_0,(\mathscr{R}(a_0),g_0)) \in Q$.

Second, let $(s,(q,g)) \in Q$.

1. Assume $s \overset{\beta}{\dashrightarrow}_S s'$.

   Case $\beta \in \Sigma^{int}_G$. Then, for all $a \in q$, $(a,s) \overset{\beta}{\dashrightarrow}_{A\otimes S} (a,s')$, hence $(a,g) \overset{\beta}{\dashrightarrow}_{A\otimes G} (a,g')$ and $((a,s'),(a,g')) \in R$. It follows that $(s',(q,g')) \in Q$.

   Case $\beta \in \Sigma^{out}_G$. Then, because $S \to_s A^{max}$, we know that for all $a \in q$, we have $a \xrightarrow{\beta}_{A^{max}}$. Then $(a,s) \overset{\beta}{\dashrightarrow}_{A\otimes S} (a',s')$, hence $(a,g) \overset{\beta}{\dashrightarrow}_{A\otimes G} (a',g')$ and

   $$((a',s'),(a',g')) \in R.$$

   It follows that (by determinism of G) that there is a unique $g \overset{\beta}{\dashrightarrow}_G g'$, thus $(q,g) \overset{\beta}{\dashrightarrow}_{A^{max}\rhd G} (q',g')$ with

   $$q' = \bigcup\{\mathscr{R}(a') \mid \exists a \in q : a \overset{\beta}{\dashrightarrow}_{A^{max}} a'\}.$$

   The claim $(s',(q',g')) \in Q$ follows then with the same arguments used above in the demonstration of $(s_0,(\mathscr{R}(a_0),g_0)) \in Q$.

   Case $\beta \in \Sigma^{in}_G$. If for all $a \in q$, $a \overset{\beta!}{\not\dashrightarrow}_{A^{max}}$, then by rule (5) we can infer that $(q,g) \overset{\beta}{\dashrightarrow}_{A^{max}\rhd G}$ univ and then $(s',\text{univ}) \in Q$. If there exists $a \in q$ such that $a \overset{\beta!}{\dashrightarrow}_{A^{max}} a'$, then the proof of $(s',(q',g')) \in Q$ is analogous to the previous case.

2. Assume $(q,g) \xrightarrow{\beta}_{A^{max}\rhd G} (q',g')$. This transition must come from rule (3) thus there is some $a \in q$ such that $a \overset{\beta}{\dashrightarrow}_{A^{max}} a'$. Let $a \in q$ be an arbitrary state in A^{max} with $a \overset{\beta}{\dashrightarrow}_{A^{max}} a'$. Since A^{max} is an implementation, we also know that $a \xrightarrow{\beta}_{A^{max}} a'$. Then $(a,s) \xrightarrow{\beta}_{A\otimes S} (a',s')$, hence $(a,g) \xrightarrow{\beta}_{A\otimes G} (a',g')$ and $((a',s'),(a',g')) \in R$. It follows that (by determinism of S) that there is a unique $s \xrightarrow{\beta}_S s'$. The claim $(s',(q',g')) \in Q$ follows then with the same arguments used above in the demonstration of $(s_0,(\mathscr{R}(a_0),g_0)) \in Q$.

Finally, observe that conditions of a refinement relation are also satisfied for the pairs $(s, \mathsf{univ}) \in Q$. This finishes the proof that $S \leq_{s,A} G$ implies $S \leq_s A \rhd G$.

$\Leftarrow$: For the other direction assume Q as a refinement relation demonstrating $S \leq_s (A \rhd G)$. We can assume $\bar{A} \leq_s A$ with refinement relation P and $G \rightarrow_s \bar{A}$ (which already holds by assumption of the theorem).

We first show that $S \rightarrow_s \bar{A}$. By Lemma 7.3.14 we know that $A \rhd G \rightarrow_s A$, and since $S \leq_s A \rhd G$ we can infer by preservation of environment correctness that $S \rightarrow_s A$, and again $\bar{A} \leq_s A$ implies $S \rightarrow_s \bar{A}$.

Second, we show that $St_{\bar{A}} \otimes St_S \leq_s St_{\bar{A}} \otimes St_G$. We define $R \subseteq ((\bar{A} \times S) \times (\bar{A} \times G))$ by

$$R = \{((\bar{a}, s), (\bar{a}, g)) \mid \exists a \in St_A, q \in \mathscr{P}_{\geq 1}(St_A) : (\bar{a}, a) \in P \text{ and } a \in q \text{ and } (s, (q, g)) \in Q\}.$$

Clearly it holds that $((\bar{a}_0, s_0), (\bar{a}_0, g_0)) \in R$. Now let $((\bar{a}, s), (\bar{a}, g)) \in R$. We can assume that there is $a \in St_A$, $q \in \mathscr{P}_{\geq 1}(St_A)$ such that $(\bar{a}, a) \in P$, $a \in q$ and $(s, (q, g)) \in Q$.

1. Assume $(\bar{a}, g) \xrightarrow{\beta}_{\bar{A} \otimes G} (\bar{a}', g')$. We only consider the case when β is a shared action of A and G. Then $\bar{a} \xrightarrow{\beta}_{\bar{A}} \bar{a}'$ and $g \xrightarrow{\beta}_G g'$. Then $a \xrightarrow{\beta}_A a'$ such that $(\bar{a}', a') \in P$. Then $(q, g) \xrightarrow{\beta}_{A \rhd G} (q', g')$ and $a' \in q'$ according to rule (4), thus $s \xrightarrow{\beta}_S s'$ such that $(s', (q', g')) \in Q$. It follows that $(\bar{a}, s) \xrightarrow{\beta}_{\bar{A} \otimes S} (\bar{a}', s')$ such that $((\bar{a}', s'), (\bar{a}', s')) \in R$.

2. Assume $(\bar{a}, s) \overset{\beta}{\dashrightarrow}_{\bar{A} \otimes S} (\bar{a}', s')$. We only consider the case when β is a shared action of A and S. Then $\bar{a} \overset{\beta}{\dashrightarrow}_{\bar{A}} \bar{a}'$ and $s \overset{\beta}{\dashrightarrow}_S s'$. Then $a \overset{\beta}{\dashrightarrow}_A a'$ such that $(\bar{a}', a') \in P$, and from $(s, (q, g)) \in Q$ it follows that there is $(q, g) \overset{\beta}{\dashrightarrow}_{A \rhd G} (q', g')$, hence $g \overset{\beta}{\dashrightarrow}_G g'$. By definition of q' in rule (3) we know that $a' \in q'$. Then $(\bar{a}, g) \overset{\beta}{\dashrightarrow}_{\bar{A} \otimes G} (\bar{a}', g')$ and $((\bar{a}', s'), (\bar{a}', g')) \in R$. □

A consequence of the above theorem is that the weakening operator can be used to compute normal forms of modal contracts. This is shown in the following theorem.

Theorem 7.3.16 (Normal form)
Let (A, G) be a modal contract. Then $(A, A \rhd G)$ is a semantically equivalent modal contract in normal form. In the following we write G^{nf} for $A \rhd G$ if A is clear from the context.

Proof. First of all, $A \rhd G \rightarrow_s A$ follows from part (c) of Theorem 7.3.14.

Now, we have to show that $(A, A \rhd G)$ is semantically equivalent to (A, G), thus we have to show that

$$[\![(A,G)]\!]_{\mathrm{beh}} = [\![(A,G^{nf})]\!]_{\mathrm{beh}} \text{ and } [\![(A,G)]\!]_{\mathrm{env}} = [\![(A,G^{nf})]\!]_{\mathrm{env}}. \qquad \square$$

The latter equality is trivial. The former equality $[\![(A,G)]\!]_{\mathrm{beh}} = [\![(A,G^{nf})]\!]_{\mathrm{beh}}$ follows from part (a) of Lemma 7.3.14 and from Theorem 7.3.15 since

$$\begin{aligned} S \leq_{s,A} G &\iff S \leq_s (A \rhd G) \\ &\iff S \leq_s (A \rhd (A \rhd G)) \\ &\iff S \leq_{s,A} (A \rhd G). \end{aligned}$$

Thus, we have shown that $Th^{d\mathbb{MIO}}_{strong}$ has normal forms.

Example 7.3.17
We would like to exemplify the computation of a normal form of a modal contract by reconsidering the example of the introduction of this chapter. The modal contract (A,G) *is shown in Figure 7.10. As the environment is only allowed to send a message (*msg!*) once, any behaviour of* (A,G) *is allowed to implement* msg? *with a subsequent arbitrary behaviour. The normalized contract* (A,G^{nf}) *with* $G^{nf} = A \rhd G$ *is shown in Figure 7.11.*

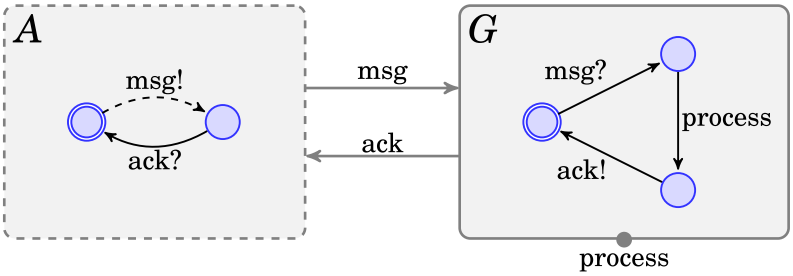


Figure 7.10: Modal contract (A,G), not in normal form

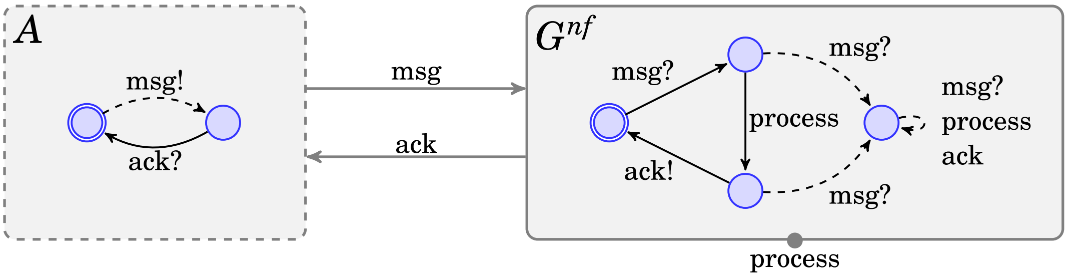


Figure 7.11: Modal contract (A,G^{nf}) in normal form

For a successful instantiation of the contract framework for $Th^{d\mathbb{MIO}}_{strong}$, we still have to show Assumption 4:

Lemma 7.3.18

Whenever $S_1 \rightarrow_s S_2 \otimes E$ and $S_2 \rightarrow_s S_1 \otimes E$, then $S_1 \otimes S_2 \rightarrow_s E$.

Proof. Let (s_1, s_2, e) be a reachable state in $S_1 \otimes S_2 \otimes E$. If $s_1 \overset{\alpha!}{\dashrightarrow}_{S_1}$ for some $\alpha \in \Sigma^{out}_{S_1} \cap \Sigma^{in}_{E}$, then according to $S_1 \rightarrow_s S_2 \otimes E$ we must have

$$(s_2, e) \xrightarrow{\alpha?}_{S_2 \otimes E} \text{ which entails } e \xrightarrow{\alpha?}_{E} .$$

If $s_2 \overset{\alpha!}{\dashrightarrow}_{S_2}$ for some $\alpha \in \Sigma^{out}_{S_2} \cap \Sigma^{in}_{E}$, then we can use similar arguments to conclude $e \xrightarrow{\alpha?}_{E}$. □

Importantly, we prove that variant 1 of finding a dominating modal contract, see page 163 results in a strongest dominating modal contract given that the decompositions are independent of each other, i.e. there are no shared actions between assumptions of the same modal contract. Under this condition of independent assumptions, we prove that the dominating modal contract constructed along the lines of variant 1 equals more precisely the dominating contract obtained by contract composition ⊠.

Corollary 7.3.19

Let $C_1 = (A'_1 \otimes A''_1, G_1)$ and $C_2 = (A'_2 \otimes A''_2, G_2)$ be two modal contracts. If

1. $(\bigcup \Sigma_{A'_1}) \cap (\bigcup \Sigma_{A''_1}) = \emptyset$ *and* $(\bigcup \Sigma_{A'_2}) \cap (\bigcup \Sigma_{A''_2}) = \emptyset$,
2. $A'_2 \otimes G^{nf}_2 \leq A''_1$ *and* $A'_1 \otimes G^{nf}_1 \leq A''_2$, *and*
3. $G^{nf}_1 \otimes G^{nf}_2 \rightarrow A'_1 \otimes A'_2$,

then

$$C_1 \boxtimes C_2 = (A'_1 \otimes A'_2, G^{nf}_1 \otimes G^{nf}_2).$$

Proof. We have to show that

$$\max_{G^{nf}_1 \otimes G^{nf}_2 \rightarrow}(((A'_1 \otimes A''_1) /\!\!/ G^{nf}_2) \wedge ((A'_2 \otimes A''_2) /\!\!/ G^{nf}_1)) = A'_1 \otimes A'_2.$$

First, using condition (1.) it is straightforward to prove that $(A'_1 \otimes A''_1) /\!\!/ G^{nf}_2$ is defined if and only if $A'_1 \otimes (A''_1 /\!\!/ G^{nf}_2)$ is defined, and if they are defined then $(A'_1 \otimes A''_1) /\!\!/ G^{nf}_2 = A'_1 \otimes (A''_1 /\!\!/ G^{nf}_2)$. Then $A'_1 \otimes (A''_1 /\!\!/ G^{nf}_2) = A'_1 \otimes X'_2$ for some $X'_2 \in d\mathbb{MIO}$ such that $A'_2 \leq_s X'_2$. Similarly, we can infer that there exists $X'_1 \in d\mathbb{MIO}$ such that $A'_2 \otimes (A''_2 /\!\!/ G^{nf}_1) = A'_2 \otimes X'_1$ and $A'_1 \leq_s X'_1$. Conditions (1.) and (2.) imply that A'_1 and A'_2 have no shared actions. Then one can show that

$$(A'_1 \otimes X'_2) \wedge (X'_1 \otimes A'_2) = (A'_1 \wedge X'_1) \otimes (X'_2 \wedge A'_2) = A'_1 \otimes A'_2.$$

Hence

$$C_1 \boxtimes C_2 = (A'_1 \otimes A'_2, G^{nf}_1 \otimes G^{nf}_2)$$

which is a valid modal contract because of condition (3.). □

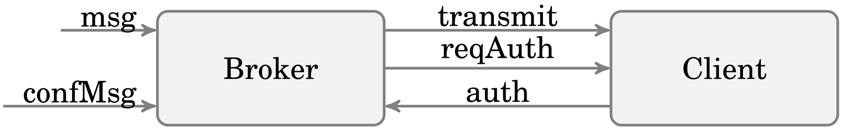


Figure 7.12: The static structure of the message transmission system

Broker		Client	
msg?	receive a message		
confMsg?	receive a confidential message		
transmit!	deliver the message to the client	transmit?	receive the message
reqAuth!	send out an authentication request	reqAuth?	receive an authentication request
auth?	receive the (valid) authentication information	auth!	send the authentication information

Table 7.1: Intuitive meaning of actions

7.3.3 Example: Message Transmission System

As an illustration we consider a simple message transmission system which consists of two components: a broker component delivers received messages to a client component. A standard message is immediately delivered while a confidential message is only delivered after successful authentication of the client. The static structure of this component system is shown in Figure 7.12. The meaning of the input and output actions is summarized in Table 7.1. The fixed global set *Act* of action names is given by $\{msg, confMsg, transmit, reqAuth, auth\}$.

In the following we specify broker and client component by different modal contracts $C^1_{Broker}, C^2_{Broker}$ and C^1_{Client}, C^2_{Client}, respectively, and illustrate the two variants of finding a (strongest) dominating contract, see Section 7.2.2.

Modal contracts for the broker component. Let us start with specifying the broker component by the modal contract

$$C^1_{Broker} = (A^{1,1}_{Broker} \otimes A^{1,2}_{Broker}, G^1_{Broker})$$

shown in Figure 7.13. Note that the assumption $A^{1,1}_{Broker} \otimes A^{1,2}_{Broker}$ is a composition of two individual and independent assumptions on the environment of the broker component. C^1_{Broker} models the following requirements for the behaviour and the environment.

- $A^{1,1}_{Broker}$ specifies that at any time the environment may send messages and confidential messages.
- G^1_{Broker} specifies that any standard message (*msg*) received by the environment is immediately sent to the client (*transmit*). Whenever the environment sends a confidential message (*confMsg*), then this message is only delivered to the client if the broker receives valid authentication information (*auth*) from the client after requesting it (*reqAuth*).

- $A^{1,2}_{Broker}$ is an assumption on the client component and specifies that delivered messages as well as authentication requests must be received at any time, and once an authentication request is received, the client must answer with sending the authentication information to the broker. Note that $A^{1,2}_{Broker}$ never disallows the sending of the authentication information.

Obviously, C^1_{Broker} is a valid modal contract, satisfying the following to requirements: First, the action signatures of assumption and guarantee compose to an action signature with the set *Act* of internal actions; second, $A^{1,1}_{Broker} \otimes A^{1,2}_{Broker}$ is a strongly correct environment for G^1_{Broker} since $A^{1,2}_{Broker}$ requires must-transitions for every input action *transmit?* and *reqAuth?* to be enabled in every state. Finally, note that C^1_{Broker} is in normal form since in the assumption $A^{1,1}_{Broker} \otimes A^{1,2}_{Broker}$, the outputs *msg!*, *confMsg!* and *auth!* are always allowed.

Consider another modal contract

$$C^2_{Broker} = (A^2_{Broker}, G^2_{Broker})$$

for the broker component shown in Figure 7.14. The main difference between C^2_{Broker} and C^1_{Broker} is that the assumption in C^2_{Broker} is specified by a single MIO. The guarantee G^{Broker}_2 is the same as the guarantee G^1_{Broker} of the previous modal contract C^1_{Broker}. The assumption A^2_{Broker} formulates requirements on the whole environment of the broker (including the client component), basically following the same control flow of the guarantee G^2_{Broker}. The assumption A^2_{Broker} expresses that

- any action is allowed in any state;

- once a confidential message was sent in the initial state, the environment is obliged to answer any authentication request as the transition labelled with *auth!* is a must-transition.

The modal contracts C^2_{Broker} is also in normal form as for every output action in the environment there is a transition enabled in every state of the assumption.

We finally remark that C^2_{Broker} is a refinement of C^1_{Broker}. As both contracts are in normal form, by Corollary 7.2.11 we only have to check that[1]

$$A^{1,1}_{Broker} \otimes A^{1,2}_{Broker} \leq_s A^2_{Broker} \quad \text{and} \\ G^2_{Broker} \leq_s G^1_{Broker}.$$

[1] Both refinements have been checked in the MIO Workbench, see also Chapter 8.

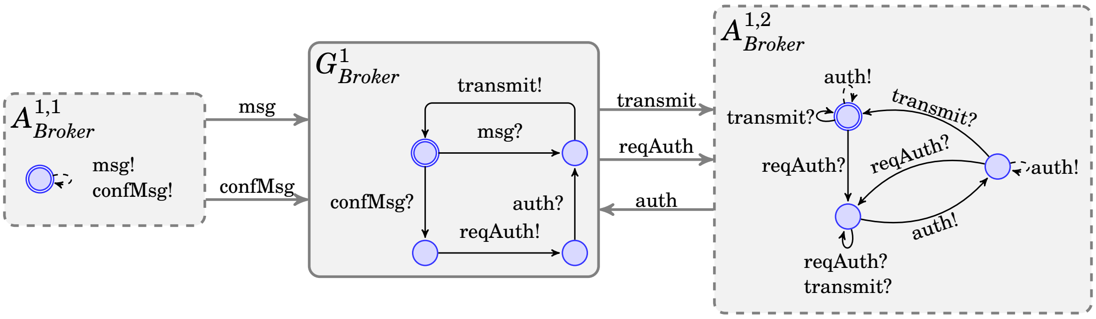


Figure 7.13: $C^{1}_{Broker} = (A^{1,1}_{Broker} \otimes A^{1,2}_{Broker}, G^{1}_{Broker})$

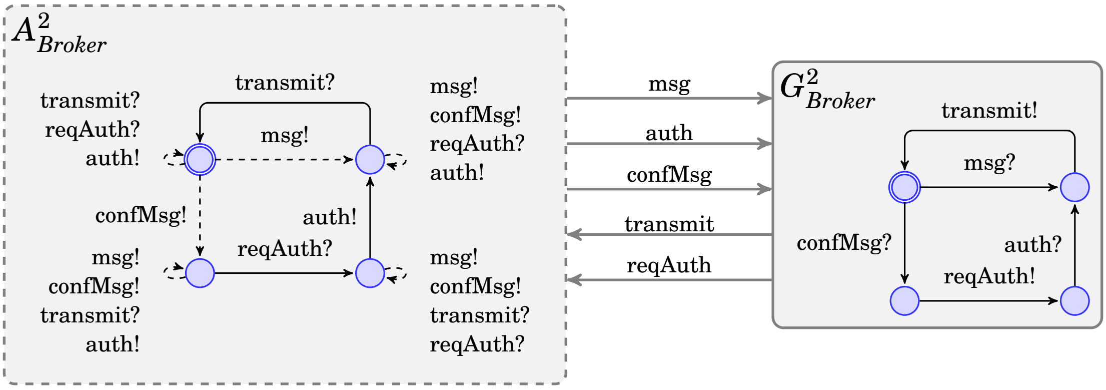


Figure 7.14: $C^{2}_{Broker} = (A^{2}_{Broker}, G^{2}_{Broker})$

Modal contracts for the client component. Similar to the broker component we consider two different variants of modal contracts for the client component. The first modal contract

$$C^1_{Client} = (A^1_{Client}, G^1_{Client})$$

is shown in Figure 7.15 and specifies that the client is always able to receive messages and authentication requests. Furthermore, the client must be able to answer authentication requests by sending the requested info to the broker. It is easy to see that the modal contract is valid again (i.e. satisfies the signature restriction, and $G^1_{Client} \rightarrow_s A^1_{Client}$). Moreover, as there is a must-transition enabled for any input action in any state of G^1_{Client}, the modal contract C^1_{Client} is in normal form.

Another contract $C^2_{Client} = (A^2_{Client}, G^2_{Client})$, shown in Figure 7.16, specifies a client which is not authorized to receive confidential messages and does not accept any authentication requests and never sends the authentication information to the broker. According to the assumption A^2_{Client}, the environment of the client may show arbitrary behaviour. The guarantee G^2_{Client} specifies that messages are always received from the broker (*transmit?*). Note that the modal contract is valid and is in normal form.

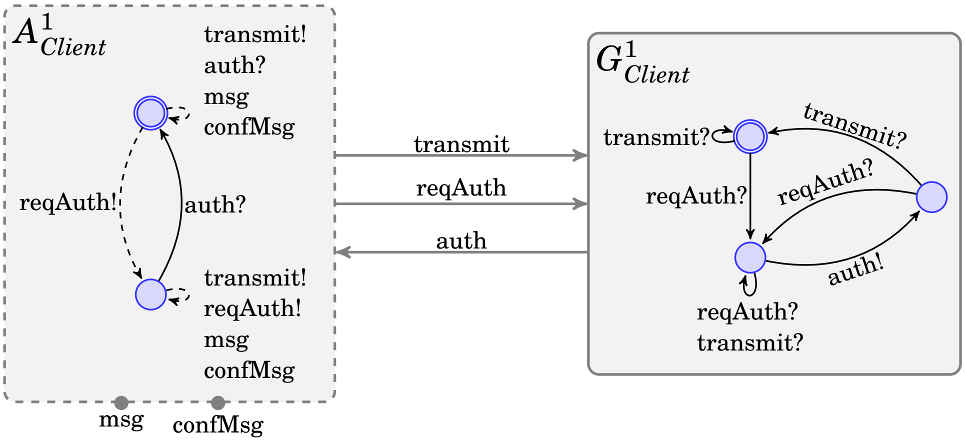


Figure 7.15: $C^1_{Client} = (A^1_{Client}, G^1_{Client})$

Contract composition. For the illustration of contract composition we consider the modal contracts C^1_{Broker}, C^2_{Broker}, C^1_{Client} and C^2_{Client} introduced above and discuss all four possible compositions.

$C^1_{Broker} \boxtimes C^1_{Client}$:

In this case, we construct $C^1_{Broker} \boxtimes C^1_{Client}$ by using Theorem 7.3.19: We can

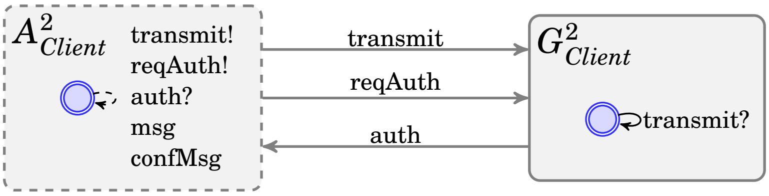


Figure 7.16: $C^2_{Client} = (A^2_{Client}, G^2_{Client})$

verify that

$$\begin{aligned}(\bigcup \Sigma_{A^{1,1}_{Broker}}) \cap (\bigcup \Sigma_{A^{1,2}_{Broker}}) = \emptyset \quad \text{and}\\ A^{1,1}_{Broker} \otimes G^1_{Broker} \leq_s A^1_{Client} \quad \text{and}\\ G^1_{Client} \leq_s A^{1,2}_{Broker} \quad \text{and}\\ G^1_{Broker} \otimes G^1_{Client} \to_s A^{1,1}_{Broker}\end{aligned}$$

are satisfied, hence

$$C^1_{Broker} \boxtimes C^1_{Client} = (A^{1,1}_{Broker}, G^1_{Broker} \otimes G^1_{Client}).$$

The contract composition $C^1_{Broker} \boxtimes C^1_{Client}$ is shown in Figure 7.17.

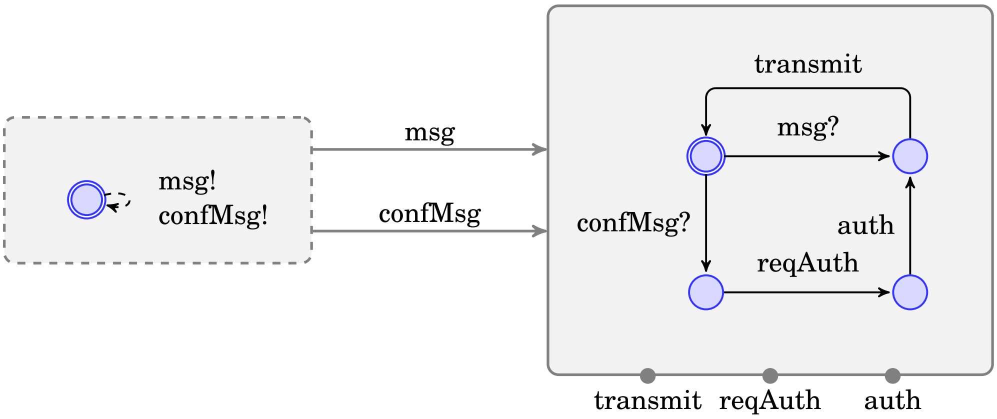


Figure 7.17: $C^1_{Broker} \boxtimes C^1_{Client}$

$C^1_{Broker} \boxtimes C^2_{Client}$:

In this case, we show that C^1_{Broker} and C^2_{Client} are not dominable, hence the contract composition $C^1_{Broker} \boxtimes C^2_{Client}$ is undefined. Assume that there exists $E \in d\mathbb{MIO}$ with action signature $\Sigma_{A^{1,1}_{Broker}} \otimes \Sigma_{G^1_{Broker}}$ such that

$$E \otimes G^2_{Client} \leq_s A^{1,1}_{Broker} \otimes A^{1,2}_{Broker}. \tag{$\star$}$$

However, $A^{1,1}_{Broker} \otimes A^{1,2}_{Broker}$ requires that in the initial state there is a must-transition enabled labelled with *reqAuth?* – as G^2_{Client} disallows this action to happen, the refinement statement ($\star$) cannot hold. Thus, by Theorem 7.2.17, C^1_{Broker} and C^2_{Client} are not dominable.

$C^2_{Broker} \boxtimes C^1_{Client}$:
Since the assumption of C^2_{Broker} is given as a single MIO for which no decomposition is given a priori, we cannot make use of Theorem 7.3.19 for a syntactic construction of $C^2_{Broker} \boxtimes C^1_{Client}$. Nevertheless, as the specification theory $Th^{d\mathbb{MIO}}_{strong}$ is complete, we can compute the contract composition $C^2_{Broker} \boxtimes C^1_{Client}$ according to its definition which yields the same contract $C^1_{Broker} \boxtimes C^1_{Client}$ as shown in Figure 7.17.

$C^2_{Broker} \boxtimes C^2_{Client}$:
Recall that C^2_{Client} specifies a client that never accepts authentication requests from the broker and never sends authentication information to the broker. Again, similar to the previous case, the assumption of C^2_{Broker} is not given by the composition of two independent assumptions, hence we have to compute $C^2_{Broker} \boxtimes C^2_{Client}$ by using the definition of contract composition $\boxtimes$. The result of $C^2_{Broker} \boxtimes C^2_{Client}$ is shown in Figure 7.18. As expected, confidential messages may not be sent by the environment, because this would lead to an unsatisfied assumption of the broker component which assumes the authentication code to be received once it is requested. The reader may verify that indeed $C^2_{Broker} \boxtimes C^2_{Client}$ is dominating C^2_{Broker} and C^2_{Client}, in particular, if $A_\boxtimes$ denotes the assumption of $C^2_{Broker} \boxtimes C^2_{Client}$ then

$$A_\boxtimes \otimes G^2_{Client} \leq_s A^2_{Broker}$$
$$A_\boxtimes \otimes G^2_{Broker} \leq_s A^2_{Client}.$$

Observe that the modal contract $C^2_{Broker} \boxtimes C^2_{Client}$ in Figure 7.18 is not in normal form, thus $C^2_{Broker} \boxtimes C^2_{Client}$ is an example that contract composition $\boxtimes$ not necessarily results in a modal contract in normal form.

7.4 Modal Contracts based on $Th^{d\mathbb{MIOD}}_{strong}$

As the second illustration, we instantiate our generic contract framework for the specification theory $Th^{d\mathbb{MIOD}}_{strong}$, with strong modal refinement and strong environment correctness, see Section 4.3. We do not introduce a complete specification theory for MIODs– we are convinced that conjunction and a maximal environment operator can be defined (though the latter would be very technical),

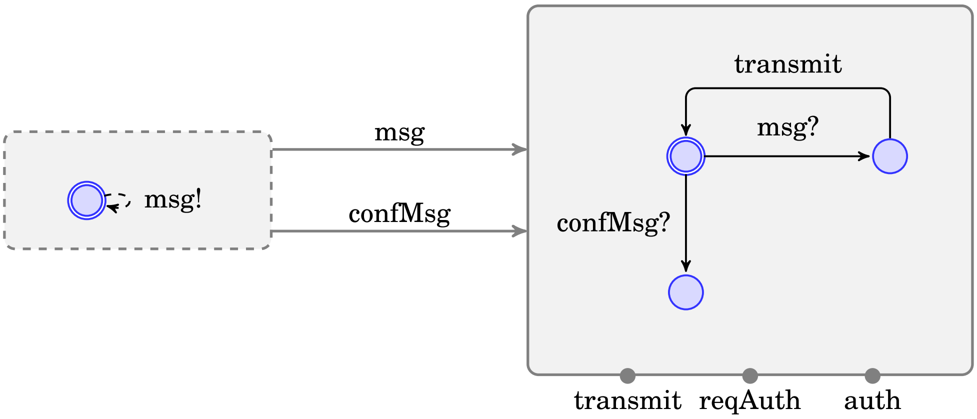


Figure 7.18: $C^2_{Broker} \boxtimes C^2_{Client}$

however, whether quotienting is possible for MIODs is unknown so far (even for deterministic MIODs) and is subject of future work.[2]

7.4.1 Modal Contracts

Similar to the instantiation of contracts for MIOs, we assume a fixed global set Act of action labels. *Modal contracts* are defined as pairs (A,G) with $A,G \in d\mathbb{MIOD}$ such that

- $G \to^d_s A$
- and $A \otimes G$ is a *closed* MIOD, i.e.
  - $\Sigma_A \otimes \Sigma_G = \Sigma_{Act}$ where Σ_{Act} is the action signature determined by $\Sigma^{in}_{Act} = \Sigma^{out}_{Act} = \emptyset$ and $\Sigma^{int}_{Act} = \emptyset$;
  - $V^{req}_A = V^{prov}_G$ and $V^{prov}_A = V^{req}_G$.

In order to illustrate the semantics of modal contracts that are using MIODs for specifying assumptions and guarantees, a *weakening operator* $\rhd$ may provide a characterization of relativized refinement for MIODs as well as obtaining normal forms of modal contracts, similarly to the case of MIOs. Crucially, in comparison to the MIO case, the weakening operator $\rhd$ not just replaces or adds may-transitions to the universal state for input actions that cannot be served by the environment, but also adds may-transitions to the universal state for output actions, with preconditions φ that are unsatisfiable by the environment (i.e. not satisfied by all possible data states of the environment). Furthermore, the weakening $A \rhd G$ yields MIODs with infinitely many states and transitions even if A

[2] In [25] we have define a quotient operator for restricted MIODs where postconditions do not refer to the previous data states.

and G are finite MIODs. The weakening $A \rhd G$ can be assured to be finite by restricting the assumptions A to finite MIODs with postconditions not referring to the previous data state. As a consequence normalization of modal contracts based on finite MIODs is not guaranteed to result in finite MIODs. The definition of the weakening operator as well as a further discussion on sufficient conditions for obtaining finite normalized contracts is out of the scope of this work.

The question of finding environments for modal contracts with MIODs can only be answered here for situations as in variant 1, see page 7.2.2. More precisely, when given two modal contracts in normal form and with decomposed assumptions, then Theorem 7.2.18 offers a way to finding a dominating modal contract. Variant 2, see page 7.2.2 would require to show that $Th_{strong}^{d\mathbb{MIOD}}$ is a complete specification theory (which is left for future work).

In what follows, we describe in detail the behaviour semantics $[\![C]\!]_{\mathrm{beh}}$ of modal contracts C based on MIODs. As said before, we do not define the weakening operator for MIODs, but propose *relativized refinement relations* that enable us to characterize relativized refinement derived from strong modal refinement $\leq_s^d$.

Relativized refinement relations are very related to usual refinement relations for strong modal refinement, cf. Definition 4.3.1 in Chapter 4, however with the following differences:

- A must-transition in G with external action must only be simulated if the environment A does have a may-transition that can synchronize with that transition. Otherwise, there is no requirement because this transition does not occur in the composition $A \otimes^d G$ (see condition 1 in Definition 7.4.1).

- Similarly, any may-transition in S with external action must only be simulated if the environment A does have a may-transition that can synchronize with that transition. Otherwise, there is no requirement because this transition does not occur in the composition $A \otimes^d S$ (see condition 2 in Definition 7.4.1).

A relativized refinement relation R for the refining MIOD S, the context MIOD A and the refined MIOD G, is a relation in $St_S \times St_A \times \mathcal{D}(V_A^{prov}) \times St_G \times \mathcal{D}(V_G^{prov})$. To take into account all possible states of the environment A, R must be closed under may-transitions of A (see condition 3 in Definition 7.4.1). Formally, relativized refinement relations are defined as follows.

Definition 7.4.1 (Relativized Refinement Relation)
Let $S,G \in d\mathbb{MIOD}$ with the same extended action signature $\Sigma = (\Sigma^{in}, \Sigma^{out}, \Sigma^{int})$ and state signature $V = (V^{prov}, V^{req})$. Let $A \in d\mathbb{MIOD}$ with action signature $\Sigma_A = (\Sigma_A^{in}, \Sigma_A^{out}, \Sigma_A^{int})$ such that $\Sigma_A \otimes \Sigma = \Sigma_{Act}$, and with state signature $V_A = (V_A^{prov}, V_A^{req})$ such that $V_A^{prov} = V^{req}$, $V_A^{req} = V^{prov}$.

A relation $R \subseteq St_S \times St_A \times \mathcal{D}(V_A^{prov}) \times St_G \times \mathcal{D}(V_G^{prov})$ is a relativized refinement relation *for (S,A,G) if $\models_\forall \varphi_{0,S} \Rightarrow \varphi_{0,G}$, $(s_0, a_0, \delta_{0,A}, g_0, \delta_0) \in R$ for all $\delta_0 \in \mathcal{D}(\varphi_{0,S})$, $\delta_{0,A} \in \mathcal{D}(\varphi_{0,A})$, and for all $(s,a,\delta_A,g,\delta) \in R$:*

1. ***(from abstract to concrete)***
Let $g \xrightarrow{[\varphi_G]\alpha[\pi_G]}_G g'$ and $(\delta \cdot \delta_A;\rho) \models \varphi_G$ with $\rho \in \mathscr{D}(par(\beta))$.

Case $\beta \in \Sigma_G^{int}$**.** *Then there exists*

$$s \xrightarrow{[\varphi_S]\beta[\pi_S]}_S s'$$

such that $(\delta \cdot \delta_A;\rho) \models \varphi_S$, and for all $\delta' \in \mathscr{D}(\mathrm{V}_G^{prov})$, if

- $(\delta \cdot \delta_A,\delta';\rho) \models \pi_S$

then

- *$(\delta \cdot \delta_A,\delta';\rho) \models \pi_G$ and*
- *$(s',a,\delta_A,g',\delta') \in R$.*

Case $\beta \in \Sigma_G^{ext}$**.** *If there exists*

$$a \overset{[\varphi_A]\beta[\pi_A]}{\dashrightarrow}_A a'$$

such that $(\delta \cdot \delta_A;\rho) \models \varphi_A$, then there exists

$$s \xrightarrow{[\varphi_S]\beta[\pi_S]}_S s'$$

such that $(\delta \cdot \delta_A;\rho) \models \varphi_S$, and for all $\delta'_A \in \mathscr{D}(\mathrm{V}_A^{prov})$, for all $\delta' \in \mathscr{D}(\mathrm{V}_G^{prov})$, if

- *$(\delta \cdot \delta_A,\delta'_A;\rho) \models \pi_A$ and*
- $(\delta \cdot \delta_A,\delta';\rho) \models \pi_S$

then

- *$(\delta \cdot \delta_A,\delta';\rho) \models \pi_G$ and*
- *$(s',a',\delta'_A,g',\delta') \in R$.*

2. ***(from concrete to abstract)***
Let $s \overset{[\varphi_S]\alpha[\pi_S]}{\dashrightarrow}_S s'$ and $(\delta \cdot \delta_A,\delta';\rho) \models \varphi_G \wedge \pi_G$ with $\rho \in \mathscr{D}(par(\beta))$ and $\delta' \in \mathscr{D}(\mathrm{V}_G^{prov})$.

Case $\beta \in \Sigma_G^{int}$**.** *Then there exists*

$$g \overset{[\varphi_G]\beta[\pi_G]}{\dashrightarrow}_G g'$$

such that $(\delta \cdot \delta_A,\delta';\rho) \models \varphi_G \wedge \pi_G$ and $(s',a,\delta_A,g',\delta') \in R$.

Case $\beta \in \Sigma_G^{ext}$**.** *If there exist* $\delta_A \in \mathscr{D}(V_A^{prov})$ *and*

$$a \overset{[\varphi_A]\beta[\pi_A]}{\dashrightarrow}_A a'$$

such that $(\delta \cdot \delta_A, \delta'_A; \rho) \models \varphi_A \wedge \pi_A$*, then there exists*

$$g \overset{[\varphi_G]\beta[\pi_G]}{\dashrightarrow}_G g'$$

such that $(\delta \cdot \delta_A, \delta'; \rho) \models \varphi_G \wedge \pi_G$ *and* $(s', a', \delta'_A, g', \delta') \in R$.

3. ***(context)***
   Let $a \overset{[\varphi_A]\beta[\pi_A]}{\dashrightarrow}_A a'$ *and* $(\delta \cdot \delta_A, \delta'_A; \rho) \models \varphi_A \wedge \pi_A$ *then* $(s, a', \delta'_A, g, \delta) \in R$.

The following lemma shows a characterization of relativized refinement relations that will be used to formulate a characterization of the behaviour semantics of modal contracts.

Lemma 7.4.2
Let S, A, G *be MIODs as in Definition 7.4.1. Then there exists a relativized refinement relation* R *for* (S, A, G) *if and only if for all* $A' \leq_s^d A$, $(A' \otimes^d S) \leq_s^d (A' \otimes^d G)$*.*

Proof. $\Rightarrow$: Assume that there exists a relativized refinement relation R for (S, A, G). Assume that $A' \leq_s^d A$, demonstrated by a refinement relation P. We define a relation $Q \subseteq (A' \times S) \times (A' \times G) \times \mathscr{D}(V_A^{prov} \cup V_G^{prov})$ by

$$Q = \{((a', s), (a', g), \delta_A \cdot \delta) \mid \exists a \in A : (a', a, \delta_A) \in P, (s, a, \delta_A, g, \delta) \in R\}.$$

Clearly $((a'_0, s_0), (a'_0, g_0), \delta_{0,A} \cdot \delta_0) \in Q$ for all $\delta_{0,A} \cdot \delta_0 \in \mathscr{D}(V_A^{prov} \cup V_G^{prov})$ such that $\delta_{0,A} \cdot \delta_0 \models \varphi_{0,A'} \wedge \varphi_{0,S}$. Now let $((a', s), (a', g), \delta_A \cdot \delta) \in Q$. By definition of Q there exists $a \in A$ such that $(a', a, \delta_A) \in P$ and $(s, a, \delta_A, g, \delta) \in R$.

1. Assume that $(a', g) \xrightarrow{[\varphi]\alpha[\pi]} (\hat{a}', \hat{g})$ such that $(\delta \cdot \delta_A; \rho) \models \varphi$.

   Case $\beta \in \Sigma_{A \otimes^d S}^{int}$**.** If $\beta \in \Sigma_A^{int}$ then $a' \xrightarrow{[\varphi]\beta[\pi]} \hat{a}'$ and $g = \hat{g}$. From $(a', a, \delta_A) \in P$ it follows that for all $\delta'_A \in \mathscr{D}(V_A^{prov})$ with $(\delta_A \cdot \delta, \delta'_A; \rho) \models \varphi \wedge \pi$ there is

   $$a \overset{[\varphi_A]\beta[\pi_A]}{\dashrightarrow} \hat{a}$$

   such that $(\delta \cdot \delta_A, \delta'_A; \rho) \models \varphi_A \wedge \pi_A$ and $(\hat{a}', \hat{a}, \delta'_A) \in P$. By definition of R we can infer that $(s, \hat{a}, \delta'_A, g, \delta) \in R$. All together, there is

   $$(a', s) \xrightarrow{[\varphi]\alpha[\pi]} (\hat{a}', s)$$

such that $(\hat{a}',\hat{a},\delta'_A)\in P$ and $(s,\hat{a},\delta'_A,g,\delta)\in R$ implying $((\hat{a}',s),(\hat{a}',g),\delta'_A\cdot\delta)\in Q$.

If $\beta\in\Sigma^{int}_S=\Sigma^{int}_G$. Then $\hat{a}'=a'$ and

$$g\xrightarrow{[\varphi]\beta[\pi]}\hat{g}.$$

From $(s,a,\delta_A,g,\delta)\in R$ it follows that there is

$$s\xrightarrow{[\varphi_S]\beta[\pi_S]}\hat{s}$$

such that $(\delta\cdot\delta_A;\rho)\models\varphi_S$ and for all $\delta'\in\mathscr{D}(V^{prov}_S)$, if $(\delta\cdot\delta_A,\delta';\rho)\models\pi_S$ then $(\delta\cdot\delta_A,\delta';\rho)\models\pi$ and $(\hat{s},a,\delta_A,\hat{g},\delta')\in R$. Thus

$$(a',s)\xrightarrow{[\varphi_S]\beta[\pi_S]}(a',\hat{s})$$

and also $((a',\hat{s}),(a',\hat{g}),\delta_A\cdot\delta')\in Q$ follows.

Case $\beta\in\Sigma^{ext}_{A\otimes^d S}$**.** Then

$$a'\xrightarrow{[\varphi_{A'}]\beta[\pi_{A'}]}\hat{a}'$$

and

$$g\overset{[\varphi_G]\beta[\pi_G]}{\dashrightarrow}\hat{g}$$

such that $\varphi'=\varphi_{A'}\wedge\varphi_G$ and $\pi'=\pi_{A'}\wedge\pi_G$. Then

$$a\overset{[\varphi_A]\beta[\pi_A]}{\dashrightarrow}\hat{a}$$

and $(\hat{a}',\hat{a},\delta'_A)\in P$ for any $\delta'_A\in\mathscr{D}(V^{prov}_A)$ such that $(\delta_A\cdot\delta,\delta'_A;\rho)\models\pi_{A'}$. Then from $(s,a,\delta_A,g,\delta)\in R$ it follows that

$$s\xrightarrow{[\varphi_S]\beta[\pi_S]}\hat{s}$$

such that $(\delta\cdot\delta_A;\rho)\models\varphi_S$ and for all $\delta'\in\mathscr{D}(V^{prov}_S)$, if $(\delta\cdot\delta_A,\delta';\rho)\models\pi_S$ then $(\delta\cdot\delta_A,\delta';\rho)\models\pi_G$ and $(\hat{s},\hat{a},\delta'_A,\hat{g},\delta')\in R$ for all $\delta'_A\in\mathscr{D}(V^{prov}_A)$ such that $(\delta_A\cdot\delta,\delta'_A;\rho)\models\pi_A$. Finally,

$$(a',s)\xrightarrow{[\varphi_{A'}\wedge\varphi_S]\beta[\pi_{A'}\wedge\pi_S]}(\hat{a}',\hat{s})$$

such that $(\delta\cdot\delta_A;\rho)\models\varphi_{A'}\wedge\varphi_S$, and for all $\delta'_A\in\mathscr{D}(V^{prov}_A)$, for all $\delta'\in\mathscr{D}(V^{prov}_S)$, if $(\delta\cdot\delta_A,\delta'\cdot\delta'_A;\rho)\models\pi_{A'}\wedge\pi_S$ then $(\delta\cdot\delta_A,\delta'\cdot\delta'_A;\rho)\models\pi_{A'}\wedge\pi_G$ and $((\hat{a}',\hat{s}),(\hat{a}',\hat{g}),\delta'\cdot\delta'_A)\in Q$.

2. Assume that $(a', s) \overset{[\varphi']\beta[\pi']}{\dashrightarrow} (\hat{a}', s')$ such that $(\delta \cdot \delta_A, \delta' \cdot \delta'_A; \rho) \models \varphi' \wedge \pi'$ with $\delta' \in \mathscr{D}(V_S^{prov})$, $\delta'_A \in \mathscr{D}(V_A^{prov})$ and $\rho \in \mathscr{D}(par(\beta))$.

   Case $\beta \in \Sigma^{int}_{A \otimes^d S}$**.** The case $\beta \in \Sigma^{int}_A$ is very similar to the corresponding case for must-transitions above.
   Let $\beta \in \Sigma^{int}_S = \Sigma^{int}_G$. Then $\hat{a}' = a'$ and

   $$s \overset{[\varphi']\beta[\pi']}{\dashrightarrow} \hat{s}.$$

   Then

   $$g \overset{[\varphi_G]\beta[\pi_G]}{\dashrightarrow} \hat{g}$$

   such that $(\hat{s}, a, \delta_A, \hat{g}, \delta') \in R$. Hence

   $$(a', g) \overset{[\varphi]\alpha[\pi]}{\dashrightarrow} (a', \hat{g})$$

   and $((a', \hat{s}), (a', \hat{g}), \delta_A \cdot \delta') \in Q$.

   Case $\beta \in \Sigma^{ext}_{A \otimes^d S}$**.** Then

   $$a' \overset{[\varphi_{A'}]\beta[\pi_{A'}]}{\dashrightarrow} \hat{a}'$$

   and

   $$s \overset{[\varphi_S]\beta[\pi_S]}{\dashrightarrow} \hat{s}$$

   such that $\varphi' = \varphi_{A'} \wedge \varphi_S$ and $\pi' = \pi_{A'} \wedge \pi_S$. Then there exists

   $$a \overset{[\varphi_A]\beta[\pi_A]}{\dashrightarrow} \hat{a}$$

   such that $(\delta_A \cdot \delta, \delta'_A; \rho) \models \pi_A$ and $(\hat{a}', \hat{a}, \delta'_A) \in P$. Then from $(s, a, \delta_A, g, \delta) \in R$ it follows that

   $$g \overset{[\varphi_G]\beta[\pi_G]}{\dashrightarrow} \hat{g}$$

   such that $(\delta \cdot \delta_A, \delta' \cdot \delta'_A; \rho) \models \varphi_G \wedge \pi_G$ and $(\hat{s}, \hat{a}, \delta'_A, \hat{g}, \delta') \in R$. Finally,

   $$(a', g) \overset{[\varphi_{A'} \wedge \varphi_G]\beta[\pi_{A'} \wedge \pi_G]}{\dashrightarrow} (\hat{a}', \hat{g})$$

   such that $(\delta \cdot \delta_A, \delta' \cdot \delta'_A; \rho) \models (\varphi_{A'} \wedge \varphi_G) \wedge (\pi_{A'} \wedge \pi_G)$ and $((\hat{a}', \hat{s}), (\hat{a}', \hat{g}), \delta' \cdot \delta'_A) \in Q$.

$\Leftarrow$: Let A^{max} be A with $\dashrightarrow_{A^{max}} \triangleq \longrightarrow_{A^{max}} \triangleq \dashrightarrow_A$. We can assume a refinement relation Q demonstrating $A^{max} \otimes^d S \leq^d_s (A^{max} \otimes^d G)$. We define

$$R = \{(s, a, \delta_A, g, \delta) \mid ((a, s), (a, g), \delta_A \cdot \delta) \in Q\}.$$

We show that R is a relativized refinement relation for (S,A,G). First, observe that $(s_0,a_0,\delta_{0,A},g_0,\delta_0)\in R$ for all $\delta_{0,A}\in\mathscr{D}(\varphi_{0,A})$, $\delta_0\in\mathscr{D}(\varphi_{0,S})$. Let $(s,a,\delta_A,g,\delta)\in R$. We only prove condition (1.) of Definition 7.4.1, the proofs of the other conditions are similar. Assume that

$$g\xrightarrow{[\varphi_G]\beta[\pi_G]}_G g'$$

and $(\delta\cdot\delta_A;\rho)\models\varphi_G$ with $\rho\in\mathscr{D}(par(\beta))$. If $\beta\in\Sigma_G^{int}$, then

$$(a,g)\xrightarrow{[\varphi_G]\beta[\pi_G]}_{A^{max}\otimes^d G}(a,g')$$

which implies

$$(a,s)\xrightarrow{[\varphi_S]\beta[\pi_S]}_{A^{max}\otimes^d S}(a,s')$$

such that $((a,s'),(a,g'),\delta_A\cdot\delta')\in Q$ for all $\delta'\in\mathscr{D}(V_S^{prov})$ such that $(\delta\cdot\delta_A,\delta';\rho)\models\pi_S$; in particular, $(\delta\cdot\delta_A,\delta';\rho)\models\pi_S$ implies $(\delta\cdot\delta_A,\delta';\rho)\models\pi_G$. Then $(s',a,\delta_A,g',\delta')\in R$ for all $\delta'\in\mathscr{D}(V_S^{prov})$ such that $(\delta\cdot\delta_A,\delta';\rho)\models\pi_S$.

If $\beta\in\Sigma_G^{ext}$ and there exists

$$a\overset{[\varphi_A]\beta[\pi_A]}{\dashrightarrow}_{A^{max}}a'$$

such that $(\delta\cdot\delta_A;\rho)\models\varphi_A$, then

$$a\xrightarrow{[\varphi_A]\beta[\pi_A]}_{A^{max}}a'$$

because in A^{max} every may-transition is also a must-transition. Then

$$(a,g)\xrightarrow{[\varphi_A\wedge\varphi_G]\beta[\pi_A\wedge\pi_G]}_{A^{max}\otimes^d G}(a',g')$$

which implies (by determinism of A^{max})

$$(a,s)\xrightarrow{[\varphi_A\wedge\varphi_S]\beta[\pi_A\wedge\pi_S]}_{A^{max}\otimes^d S}(a',s')$$

and $((a',s'),(a',g'),\delta'_A\cdot\delta')\in Q$ for all $\delta'_A\in\mathscr{D}(V_A^{prov})$, $\delta'\in\mathscr{D}(V_G^{prov})$ such that $(\delta\cdot\delta_A,\delta'\cdot\delta'_A;\rho)\models\pi_A\wedge\pi_S$. In particular, for every $\delta'\in\mathscr{D}(V_S^{prov})$, $(\delta\cdot\delta_A,\delta';\rho)\models\pi_S$ implies $(\delta\cdot\delta_A,\delta';\rho)\models\pi_G$. Then

$$s\xrightarrow{[\varphi_S]\beta[\pi_S]}_S s'$$

such that $(s',a',\delta'_A,g',\delta')\in R$ for all $\delta'_A\in\mathscr{D}(V_A^{prov})$, $\delta'\in\mathscr{D}(V_G^{prov})$ such that $(\delta\cdot\delta_A,\delta'\cdot\delta'_A;\rho)\models\pi_A\wedge\pi_S$. □

We now propose a characterization of behaviour semantics of modal contracts.

Theorem 7.4.3
Let (A,G) be a modal contract, and let $S \in d\mathbb{MIOD}$. Then $S \in [\![(A,G)]\!]_{\text{beh}}$ is equivalent to

1. $S \rightarrow^d_s A$ *and*
2. *there exists a relativized refinement relation for* (S,A,G)*.*

Proof. First, assume that $S \in [\![(A,G)]\!]_{\text{beh}}$. (1.) Since $G \rightarrow^d_s A$ and $A \leq A$, also $S \rightarrow^d_s A$ by definition of relativized refinement. (2.) By Lemma 7.4.2 we only have to show that for all $A' \leq^d_s A$, $A' \otimes^d S \leq^d_s A' \otimes^d G$. Let $A' \in d\mathbb{MIOD}$ such that $A' \leq^d_s A$. By preservation of environment correctness also $G \rightarrow^d_s A'$. By definition of relativized refinement we can infer that $A' \otimes^d S \leq^d_s A' \otimes^d G$.

Second, assume that $S \rightarrow^d_s A$ and that there exists a relativized refinement relation R for (S,A,G). Let $A' \in d\mathbb{MIOD}$ such that $A' \leq^d_s A$ and $G \rightarrow^d_s A'$. Then $S \rightarrow^d_s A'$ by preservation of environment correctness and $S \rightarrow^d_s A$. By Lemma 7.4.2 we can conclude that $A' \otimes^d S \leq^d_s A' \otimes^d G$ which was to be shown. □

7.4.2 Example: Bank Account

Let us consider an example of a modal contract using MIODs. The data universe is given by the set of all integers, i.e. $\mathcal{U} \triangleq \mathbb{Z}$.

The modal contract $C_{Account} = (A_{Account}, G_{Account})$ in Figure 7.19 specifies a bank account which offers operations for money to be paid in money to be withdrawn as long as the account's balance is high enough. The assumption $A_{Account}$ allows outputs $deposit(x)!$ for $x \geq 0$, and $withdraw(x)!$ for $0 \leq x \leq bal$. The guarantee $G_{Account}$ ensures that $withdraw(x)?$ must be possible exactly when $0 \leq x \leq bal$, and the next provided data state decreases bal by x.

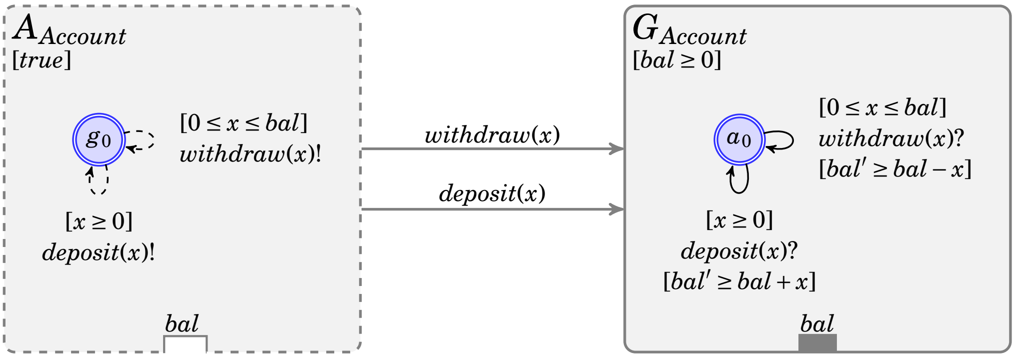


Figure 7.19: $C_{Account} = (A_{Account}, G_{Account})$

Preconditions in assumptions can be assumed by the implementor of a modal contract, i.e. any behaviour can assume that the corresponding action is only performed by the environment if the precondition is met. For instance, the precondition $0 \leq x \leq bal$ can be assumed by any behaviour whenever a synchronization on

$withdraw(x)$ between $A_{Account}$ and $G_{Account}$ happens because the environment is only allowed to issue the output $withdraw(x)!$ for $0 \leq x \leq bal$. A correct behaviour $S_{Account}$ of $C_{Account}$ is shown in Figure 7.20. Observe that as long as the environment satisfies $A_{Account}$ the specified guaranteed behaviour $G_{Account}$ is respected by $S_{Account}$. However, if the environment does not respect $A_{Account}$ – for instance, if the environment issues $withdraw(x)!$ for $x > bal$ – then there is no guaranteed behaviour anymore. As exemplified by $S_{Account}$, after $withdraw(x)?$ for $x > bal$ arbitrary behaviour is possible.

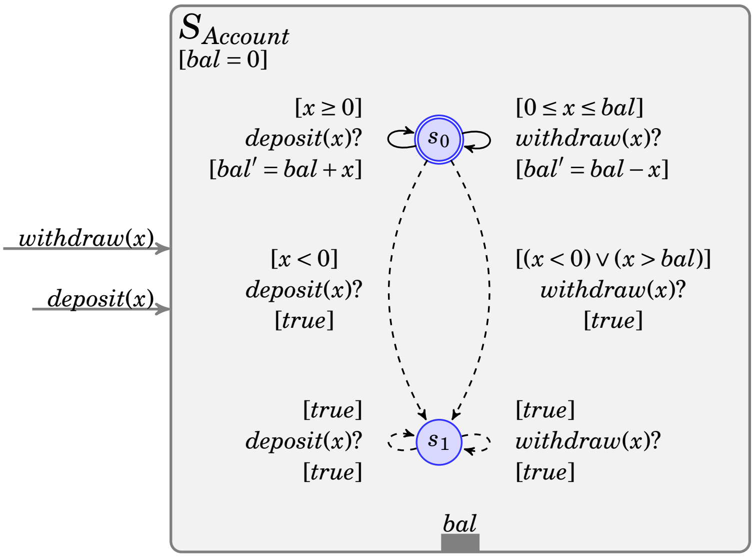


Figure 7.20: $S_{Account}$

To show that indeed $S_{Account} \in [\![C_{Account}]\!]_{\mathrm{beh}}$ is satisfied we can use the characterization proved in Theorem 7.4.3. First, $S_{Account} \rightarrow^{d}_{s} A_{Account}$ trivially holds because $S_{Account}$ has no output actions. Second, the relation

$$\{(s_0, a_0, \varepsilon, g_0, \delta) \mid \delta \in \mathcal{D}(\{bal\})\}$$

is a relativized refinement relation for $(S_{Account}, A_{Account}, G_{Account})$ where ε is the trivial data state for the empty set of variables.

Remark 7.4.4
Behaviour semantics of modal contracts are in fact closely related to the semantics of method specifications in object-oriented programs following the idea of Design by Contract [140] where methods are specified by pre- and postconditions. The precondition formulates requirements on the data state that must hold when the method body is entered; the postcondition expresses requirements on the data state that must hold at the end of the method body given that the preconditions was satisfied upon entry of the method body. These semantics of pre- and postconditions is also commonly considered for OCL operation specifications [103].

Consider the following OCL operation specification for a withdraw operation for a bank account.

```
context Account::withdraw(x : Integer)
pre: 0 <= a <= bal
post: bal = bal@pre-x
```

An implementation of withdraw(x) satisfies the above OCL operation specification if the postcondition is true whenever the precondition was true when the operation was called. In our example, the postcondition only has to be established by the operation body if the precondition is met at the entry of the operation body, i.e. if $0 \leq x \leq bal$ *is satisfied for the current value of x and bal. The contract* $C_{Account}$ *would be the natural representation of the above OCL operation specification.*

So far, we have seen how preconditions in assumptions are interpreted and how they are used to considerably enlarge the behaviour semantics of modal contracts based on MIODs. In the following we take a closer look at postconditions in assumptions that specify how the environment changes its data state when performing an action (i.e., from the viewpoint of the guarantee, how does the required data state change).

Postconditions in assumptions also relax the component behaviour semantics. To illustrate this let us consider the modal contract $C_{Client} = (A_{Client}, G_{Client})$ for a client of a bank account, with $A_{Client} = G_{Account}$ and $G_{Client} = A_{Account}$, see Figure 7.21. Whenever some behaviour of C_{Client} issues on output $deposit(x)!$ with $x \geq 0$ then it must synchronize with the corresponding transition in A_{Client}, labelled with $deposit(x)?$ and postcondition $bal' \geq bal + x$. Thus, in this case any behaviour can assume that the balance of the account has increased by at least x.

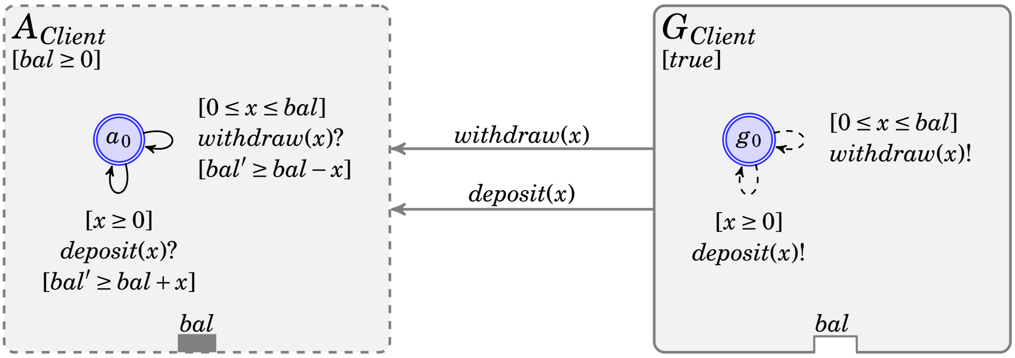


Figure 7.21: $C_{Client} = (A_{Client}, G_{Client})$

This interpretation gives rise to correct behaviours of the contract C_{Client} that may show additional outputs that were not specified in G_{Client}. As an example consider the implementation I_{Client} in Figure 7.22 which is a component behaviour of C_{Client}. Clearly, the implementation I_{Client} does not literally refine the guarantee G_{Client} because the precondition $x = 50$ for $withdraw(x)$ does not satisfy $0 \le x \le bal$ in general. However, I_{Client} refines G_{Client} in the context of A_{Client}: whenever I_{Client} has deposited an amount of 50 on the bank account, then the implementation can rely on the postcondition $bal' \ge bal + 50$, thus it is safe to issue $withdraw(x)!$ with $x = 50$; the client can be sure to have a balance of at least 50. Hence I_{Client} is a component behaviour of C_{Client}, i.e.

$$I_{Client} \in [\![(A_{Client}, G_{Client})]\!]_{\mathrm{beh}}.$$

By Theorem 7.4.3 we can formally show that I_{Client} is in $[\![(A_{Client}, G_{Client})]\!]_{\mathrm{beh}}$ by using the following relativized refinement relation R for $(I_{Client}, A_{Client}, G_{Client})$:

$$\begin{aligned} R = {} & \{(i_0, a_0, \delta_A, g_0, \varepsilon) \mid \delta_A \in \mathscr{D}(bal \ge 0)\} \\ & \cup \{(i_1, a_0, \delta_A, g_0, \varepsilon) \mid \delta_A \in \mathscr{D}(bal \ge 50)\} \end{aligned}$$

with ε the function with empty domain.

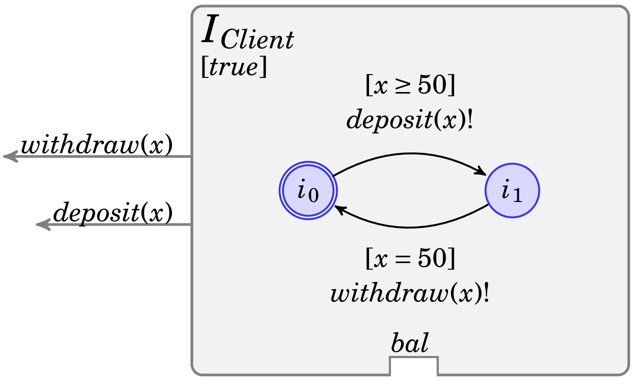


Figure 7.22: Behaviour I_{Client} of C_{Client}

7.5 Limitations of the Approach

We have shown in Corollary 7.2.25 that contracts also form a specification theory if environment correctness for contracts is defined as dominability. However, this so-defined specification theory for contracts does not fit very well in the intended meaning of the notions of composition, refinement and environment correctness. Environment correctness is in general a non-symmetric condition, however, dominability is a symmetric one; moreover, contract composition only works for dominable contracts. Thus, the theory as defined in this thesis

- lacks a syntactical composition operator that also works for contracts that are not dominable, and

- lacks a proper notion of environment correctness for contracts which is preserved by contract refinement.

One possibility to resolve these issues would be to pursue further the idea of define contract composition just for specific cases in which assumptions are already decomposed according to a suitable partition of action signatures. This idea of decomposed assumptions can be developed further arriving at so-called component contracts that consist of a behaviour (or component frame) F together with a set of port contracts (A_P, G_P) for each port P where the ports are basically a partition of the action signature of F. Port contracts describe the assumptions and guarantees at this specific interaction point of the component.

Such an approach would also solve the problem that instantiating contracts by MIOs with weak modal refinement and weak environment correctness is not possible so far. The reason for this is twofold. Firstly, specification theories for MIOs with weak modal refinement and weak environment correctness are in general not complete.[3] Secondly, contracts of the form (A, G) with a single assumption A do not seem adequate for weak environment correctness. Consider the following example as shown in Figure 7.23. Intuitively, guarantee G together with assumptions A_1 and A_2 should be valid contract because $A_1 \otimes G \rightarrow_w A_2$ and $A_2 \otimes G \rightarrow_w A_1$. But since $G \not\rightarrow_w A_1 \otimes A_2$, $(A_1 \otimes A_2, G)$ is not a contract. It seems obvious that contracts should be capable to explicitly express several assumptions.

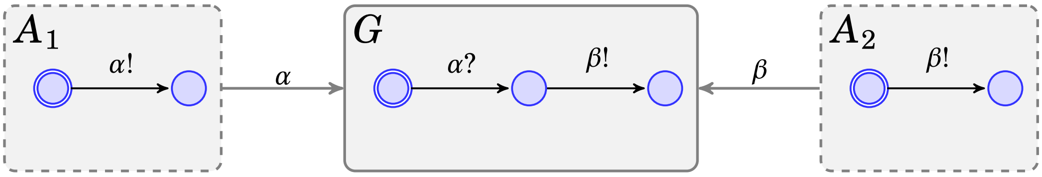


Figure 7.23: Counterexample showing that weak environment correctness is not adequately supported by the current contract framework

7.6 Related Work

The general framework for contracts is inspired by the work of Benveniste et al. [33]. They have chosen a trace-based approach, essentially following the tagged signal model of Lee and Sangiovanni-Vincentelli [128], to represent interfaces which in fact is a specification theory in our sense and an instance of our proposed abstract contract framework – except that they do not consider environment correctness and environment semantics. In [33] no definition of dominance was needed because their trace-based specification theory is complete

[3] The example in [69] can be easily adapted to show that in general there does not exist a conjunction operator for MIOs with weak modal refinement.

which makes a direct definition of contract composition possible. They go further then what is covered in this chapter by defining conjunction and disjunction of contracts which we believe can be integrated into our work, too.

The notion of relativized refinement goes back to Larsen's PhD thesis [120] in which he generalized bisimulation to take into account contexts to alleviate the stepwise refinement process.

The idea to equip a specification with behaviour and environment semantics has been used already in [17] where UML protocol state machines were considered as specifications of component interfaces.

Modal contracts have already been introduced and investigated in several previous works, including [90, 154]. Raclet and Goessler [90] propose an implementation semantics that is slightly different to ours. In their paper, an implementation I satisfies a contract (A,G) if $A \wedge I \leq_m G$ whenever $A \wedge I$ is defined, which is in fact equivalent to our definition of contract satisfaction, but only for implementations. Our satisfaction relation works for arbitrary MIOs. Refinement and composition is only syntactically defined, without any semantic considerations as we do it here, hence they lack the universal property for contract compositions.

In [154], Quinton and Graf define an abstract framework of contracts which however tends to be technical and complex due to the integration of parameterized composition operators. Besides this difference, they do not consider environment semantics; their contract semantics coincide with our component behaviour semantics. Our notion of (semantic) dominance is inspired by their (syntactical) definition of dominance, but still their work lacks a careful discussion on dominance and the universal property of contract composition.

In summary, in comparison to both works [90, 154], we consider our work as "more semantical" as behaviour and environment semantics of contracts are carefully taken into account for the definition of contracts and contract operators.

De Alfaro and Henzinger introduced a new optimistic view on compatibility and composition in their work on interface automata [61] and interface theories [62]: Communication errors in a composition should be pruned by removing input transitions that necessarily lead to those error states. The pruned interface shows then the maximal subspecification such that no communication errors occur. Larsen et al. [124] transferred de Alfaro and Henzinger's idea of optimistic compatibility and composition to modal input/output automata which was later refined by Raclet et al. in [157]. Their approach is in-fact very close to the idea of contracts, except that the optimistic approach to compatibility and composition has the disadvantage that the external and internal view on the component is mixed in a single specification with implicit assumptions. Once optimistic composition is computed, internal details of a specification are lost. The use of contracts is a possibility to separate assumptions on the environment from guarantees of the component while the latter can still keep details about the component specification (or implementation). Also, moving from specifications to contracts usually results in a more expressive specification than with single specifications because the set of correct environments can be further restricted with the assumptions.

The complete specification theory for deterministic MIOs reuses several results on conjunction and quotient that can be found in the literature. The quotient operator for modal transition systems was first defined by Raclet in [156], however without considering internal actions (which makes our powerset construction necessary). Conjunction of modal transition systems was already defined by Larsen in 1989 [121].

Publication history. A first version of the present contract framework, without the integration of environment correctness, has been presented in [13]. As an instantiation we presented in [13] modal contracts based on pure deterministic modal transition systems without input/output actions.

7.7 Summary

In this chapter, we have shown how to construct a contract framework on top of any specification theory. We have defined when a contract is dominating two other contracts, and introduced contract composition as the strongest dominating contract. Whenever the specification theory is complete (i.e. offers quotient, conjunction, and a maximal compatibility operator) then we have shown that a contract composition can be constructively defined. The general constructions on the basis of abstract specification theories has been illustrated with modal input/output automata which provide a complete specification theory (with strong modal refinement and strong environment correctness). We also briefly discussed the instantiation of the framework to modal input/output automata with data constraints and studied contract semantics in this specific setting.

Chapter 8

Tool Support: The MIO Workbench

The *MIO Workbench* [144] is a tool for the design and verification of modal input/output automata (MIOs) and modal contracts with MIOs, implementing the following theories:

- the specification theory $Th^{\mathbb{MIO}}_{strong}$, see Section 3.3,
- the specification theory $Th^{\mathbb{MIO}}_{weak}$, see Section 3.4,
- the complete specification theory $Th^{d\mathbb{MIO}}_{strong}$ including conjunction, quotient and the maximal environment operator for deterministic MIOs, see Section 7.3.1,
- modal contracts for the specification theory $Th^{d\mathbb{MIO}}_{strong}$, i.e., the generic contract framework instantiated for $Th^{d\mathbb{MIO}}_{strong}$, see Section 7.3.

MIOs and modal contracts can be specified either graphically or by simple textual syntax in an integrated text editor. The MIO Workbench supports the verification of strong and weak modal refinement as well as strong and weak environment correctness in the context of the aforementioned specification theories. These verification tasks can either be performed in a *verification view* that graphically illustrates answers to modal refinement and environment correctness checks, or in a *shell* (a command-line interpreter) that returns a text-based answer. In this chapter, we highlight the different editors and views of the MIO Workbench and describe their basic usage.

Technically, the MIO Workbench is based on the Eclipse platform [75] and offers a comfortable workbench for user interaction. Eclipse itself became known as an open-source integrated development environment for Java applications, but has grown to a general purpose software development framework. In particular, the Eclipse platform is highly extensible by Eclipse-plugins reusing basic functionality the Eclipse core offers, like resource management. The MIO Workbench is implemented as a set of Eclipse-plugins that use the Eclipse core framework and extend it by GUI elements used to design MIOs and drive their verification. The MIO Workbench is open source and can be downloaded from the website `http://www.miowb.net/`. The website of the tool additionally provides a basic usage tutorial as well as several examples.

Outline. In Section 8.1 we give an overview of the implemented relations and operations for MIOs and modal contracts. The different editors and views of the MIO Workbench are described in more detail in Section 8.2. The input language of the MIO Workbench that is used in both text editor and shell is described in Section 8.3. We compare the MIO Workbench to other related tools in Section 8.4.

8.1 Features

The MIO Workbench currently supports the specification theories $Th^{\mathbb{MIO}}_{strong}$ and $Th^{\mathbb{MIO}}_{weak}$, the complete specification theory $Th^{d\mathbb{MIO}}_{strong}$ including conjunction $\wedge$, quotient $/\!\!/$ and the maximal environment operator $\mathsf{max}_{\cdot\rightarrow}(\cdot)$, and modal contracts for the complete specification theory $Th^{d\mathbb{MIO}}_{strong}$. In addition, there are four other notions of modal refinement and compatibility for MIOs implemented in the MIO Workbench that are not further discussed here. Table 8.1 gives a detailed overview of the relations and operations that are implemented in the MIO Workbench.

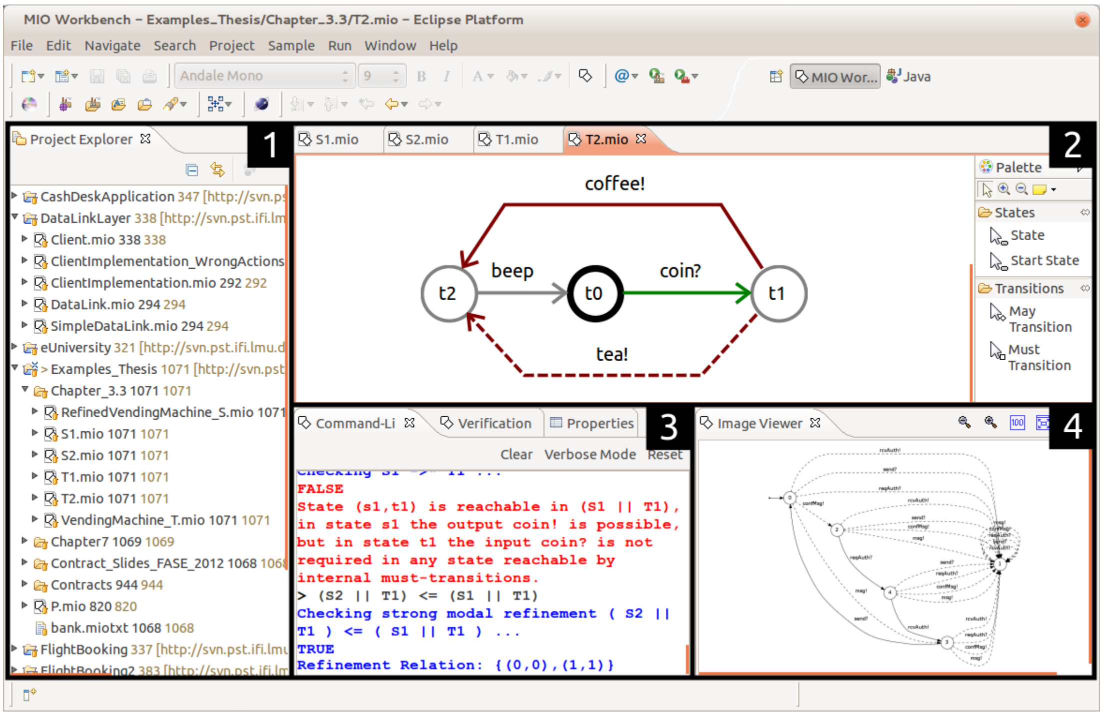
We note that contract normalization is implemented by using the weakening operator, as described in Section 7.3.2. Contract refinement is implemented by using Corollary 7.2.11, see Section 7.2.1: to verify whether $(A_1,G_1) \sqsubseteq (A_2,G_2)$ for two modal contracts (A_1,G_1) and (A_2,G_2), we first compute their normal forms (A_1,G_1^{nf}) and (A_2,G_2^{nf}), respectively, and then check whether both $A_2 \leq_s A_1$ and $G_1^{nf} \leq_s G_2^{nf}$ are satisfied.

8.2 User Interface

An overview of the graphical user interface of the MIO Workbench is shown in Figure 8.1. The package explorer (1) is a standard view from the Eclipse framework and shows all available projects and their files in the current workspace. The editor area (2) can either show a graphical editor for MIOs, an editor for modal contracts, or a text editor with syntax highlighting for textual specification of MIOs and modal contracts. The verification view, the shell and the properties view for editing selected elements in the graphical editor are gathered in area

	Verification view	Shell
Modal refinement		
Strong modal refinement $\leq_s$	✓	✓
Weak modal refinement $\leq_w$	✓	✓
May-weak modal refinement [125, 27]	✓	
Strict-observational modal refinement [139]	✓	
Environment Correctness / Compatibility		
Strong environment correctness $\rightarrow_s$	✓	✓
Weak environment correctness $\rightarrow_w$	✓	✓
Strict-observational I/O compatibility [139]	✓	
Ultra-weak compatibility [112]	✓	
Operations		
Modal synchronous composition $\otimes$		✓
Conjunction $\wedge$ for $d\mathbb{MIO}$		✓
Quotient $/\!\!/$ for $d\mathbb{MIO}$		✓
Maximal environment operator $\mathsf{max}_{\cdot\rightarrow}(\cdot)$ for $d\mathbb{MIO}$		✓
Modal contracts (generic contract framework instantiated for $Th^{d\mathbb{MIO}}_{strong}$)		
Contract refinement $\sqsubseteq$		✓
Contract composition $\boxtimes$		✓
Contract normalization (via weakening operator)		✓

Table 8.1: Relations and operations implemented in the MIO Workbench


MIO Workbench - Examples_Thesis/Chapter_3.3/T2.mio - Eclipse Platform
File Edit Navigate Search Project Sample Run Window Help
Andale Mono
MIO Wor...
Java
Project Explorer
1
CashDeskApplication 347 [http://svn.ps
DataLinkLayer 338 [http://svn.pst.ifi.lm
Client.mio 338 338
ClientImplementation_WrongActions
ClientImplementation.mio 292 292
DataLink.mio 294 294
SimpleDataLink.mio 294 294
eUniversity 321 [http://svn.pst.ifi.lmu.d
> Examples_Thesis 1071 [http://svn.pst
Chapter_3.3 1071 1071
RefinedVendingMachine_S.mio 1071
S1.mio 1071 1071
S2.mio 1071 1071
T1.mio 1071 1071
T2.mio 1071 1071
VendingMachine_T.mio 1071 1071
Chapter7 1069 1069
Contract_Slides_FASE_2012 1068 1068
Contracts 944 944
P.mio 820 820
bank.miotxt 1068 1068
FlightBooking 337 [http://svn.pst.ifi.lmu
FlightBooking2 383 [http://svn.pst.ifi.lm
S1.mio
S2.mio
T1.mio
T2.mio
2
coffee!
beep
coin?
t2
t0
t1
tea!
Palette
States
State
Start State
Transitions
May Transition
Must Transition
Command-Li
Verification
Properties
3
Clear
Verbose Mode
Reset
Image Viewer
4

```
Checking S1 ->* T1 ...
FALSE
State (s1,t1) is reachable in (S1 || T1),
in state s1 the output coin! is possible,
but in state t1 the input coin? is not
required in any state reachable by
internal must-transitions.
> (S2 || T1) <= (S1 || T1)
Checking strong modal refinement ( S2 ||
T1 ) <= ( S1 || T1 ) ...
TRUE
Refinement Relation: {(0,0),(1,1)}
```

Figure 8.1: MIO Workbench default perspective

(3). Finally, the user can send (possibly newly constructed) MIOs from the shell to the image view (4) that draws the MIOs using the layout program dot from the open source graph visualization software Graphviz [94].[1] We now describe areas (2), (3) and their editors and views in more detail.

8.2.1 Editors (Area 2)

Graphical MIO Editor

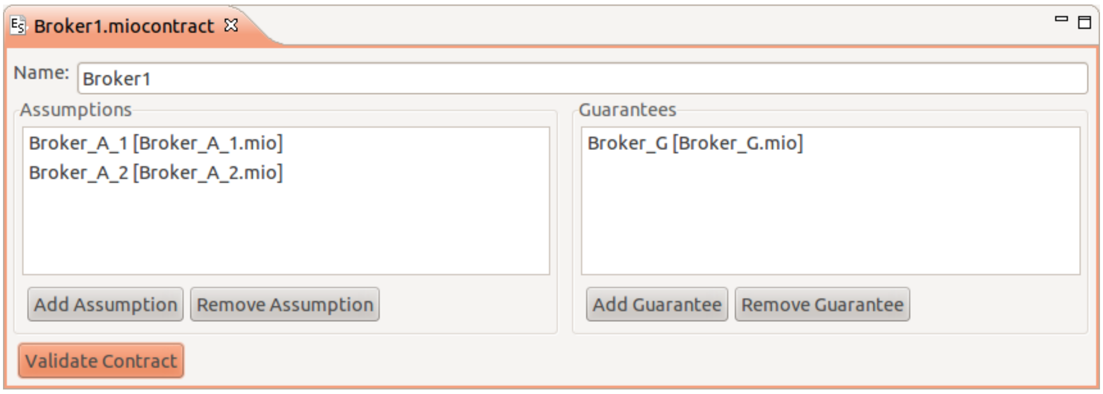
The graphical editor of the MIO Workbench displays a MIO in the classical way as a graph, with states as nodes and transitions as directed edges – may-transitions are dashed and are omitted if there is a corresponding must-transition connecting the respective states under the same action. To highlight the action type, input actions are suffixed with a question mark (?) and output actions are suffixed with an exclamation mark (!). In addition, transitions labelled with an input, output and internal action are coloured green, red and gray, respectively. The palette on the right hand side offers the creation of states and transitions. All other editing facilities are either accessible by context menus, by direct editing (e.g. of action names) in place, or by the properties view of the workbench.

[1] How to properly install the MIO Workbench and Graphviz is described on the webpage of the MIO Workbench [144].

The layout can be manually changed simply by moving states and transitions. A basic automatic layout algorithm is available in the context menu. A new MIO can be created by using the corresponding wizard; a MIO is stored in an XML-file with file extension *.mio*.

Modal Contract Editor

For the design of modal contracts, the MIO Workbench offers a custom editor to group together MIOs for the specification of the environment (the assumption) and for the component itself (the guarantee), see Figure 8.2. The user can specify two lists of MIOs modelling the assumptions and guarantees, respectively. The assumption of the modal contract is then given by the conjunction of all MIOs in the list on the left hand side, and similarly, the guarantee of the modal contract is given by the conjunction of all MIOs in the list on the right hand side. A validation check is available in the contract editor that checks whether the list of assumptions and the list of guarantees can actually be conjoined, respectively, and if the conjoined assumptions form a correct environment for the conjoined guarantees. All MIOs in modal contracts are references to the individual *.mio* files, and each MIO can can be opened in the graphical MIO editor by a double click. A modal contract is stored in an XML-file with extension *.miocontract* that is basically a list of the MIO references.



Broker1.miocontract

Name: Broker1

Assumptions

Broker_A_1 [Broker_A_1.mio]
Broker_A_2 [Broker_A_2.mio]

Guarantees

Broker_G [Broker_G.mio]

Figure 8.2: Modal contract editor

Text Editor

The MIO Workbench features a text editor offering a text-based input. The input language, described in more detail in Section 8.3, covers the definition of MIOs and modal contracts as well as relations and operators (see Table 8.1). The text editor should be used to write larger designs (or scripts) consisting of several MIOs or modal contracts as well as refinement and environment correctness checks. The text editor offers a syntax highlighting for the input language, with error indication whenever a statement cannot be parsed. Technically, the text editor together with the underlying parser is generated from the grammar using Xtext [171], see Section 8.3. The scripts need to have the file extension *.miotxt* – the text editor is automatically opened for files with this extension.

The scripts are written in the text editor and executed in the shell, a command-line interpreter, which is described below.

8.2.2 Views (Area 3)

Verification View

The verification view offers a side-by-side view of two MIOs. MIOs from the project explorer can be put on one of the sides by a simple drag-and-drop. The verification view supports refinement and environment correctness checks, see Table 8.1. The desired refinement or environment correctness notion can be selected in the middle panel of the view. For refinement checks, the verification view shows in the positive case a refinement relation by drawing links between related states, in the negative one error path (marked in red) is displayed.[2] Figure 8.3 shows a sample of a negative refinement check. For environment correctness checks, the verification view illustrates in the positive case all reachable states of the composition by drawing links between state pairs. In the negative case, one reachable state pair and an output transition is marked in red that cause the environment correctness not to be satisfied.

Shell

The shell offers a powerful text-based interface to the MIO Workbench, for specification of MIOs and modal contracts as well as the checking of refinement and environment correctness. The shell can be seen in area (3) in Figure 8.1. Commands can either be executed by typing them directly in the view, or all commands of a *.miotxt* file can be executed by drag-and-drop of the file from the project explorer onto the shell. Such *.miotxt* files should be created with the text editor, see above. The shell reuses the parser generated from the grammar of the input language (see Section 8.3), the obtained abstract syntax tree is manually traversed and evaluated. In addition to *.miotxt* files, the shell similarly accepts *.mio* and *.miocontract* files storing the described MIO and modal contract, respectively, under their names.

The shell gives a text-based answer to every command, for instance, to every refinement check the shell answers either with a refinement relation in the

[2] The illustration of negative answers to refinement checks is subject of future work, see Section 9.2. For deterministic MIOs displaying one error path is fine, however, for non-deterministic MIOs one would have to depict a set of paths which causes the refinement not to hold.

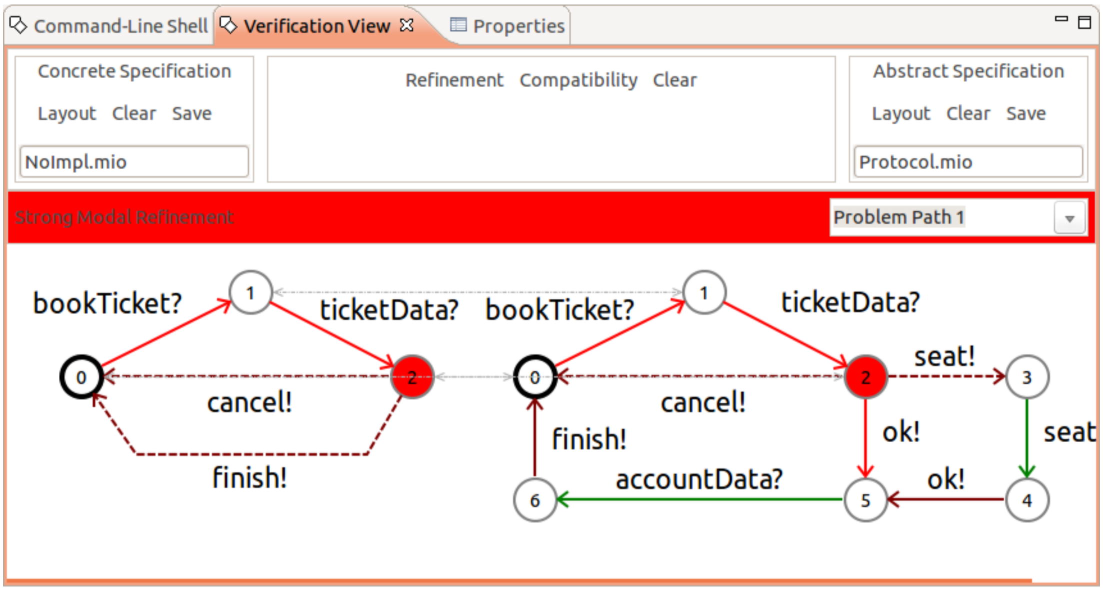


Figure 8.3: Negative strong modal refinement check in the verification view

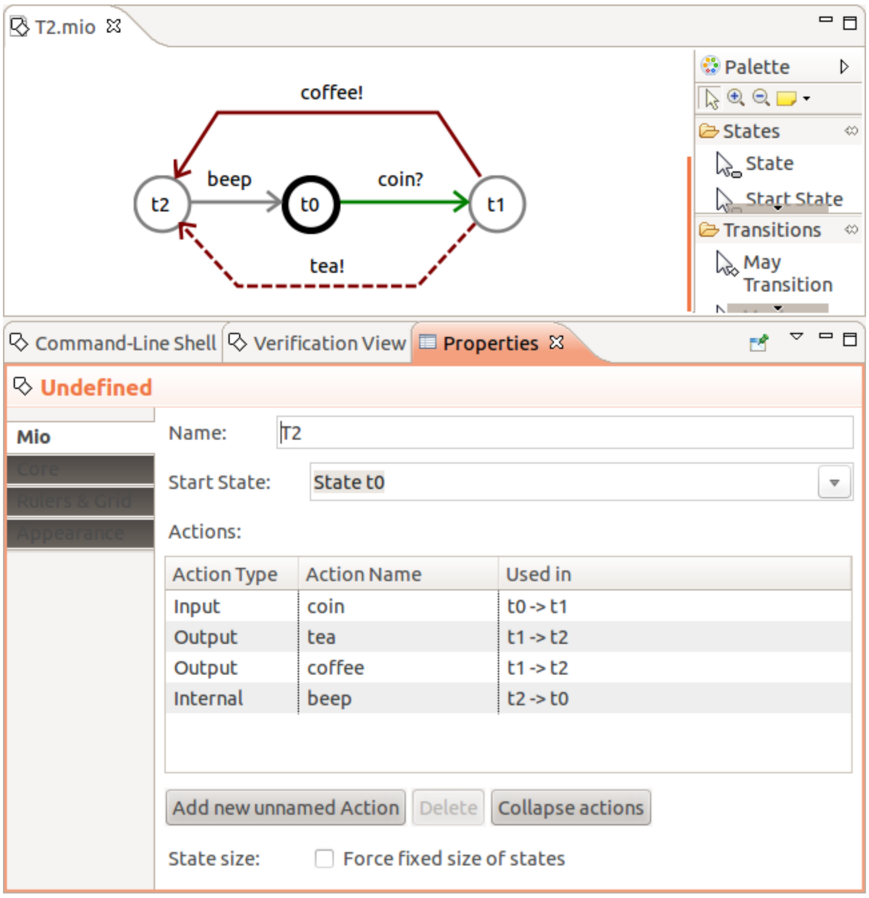
positive case, in the negative no further information is returned; to every environment correctness check the shell answers positively, or in the negative case it prints one communication error with the involved states and action. The input language is described in detail in the Section 8.3.

Properties View

The properties view can be found in the area (3). It is only important when the graphical MIO editor is used, then the properties view allows to edit a state (the name) or a transition (the action and the modality) that is selected in the graphical MIO editor. When no state or transition is selected in the graphical MIO editor, then the properties view shows the properties of the whole MIO, i.e. the name, the start state, a list of actions, and a checkbox to force fixed size of the states. This latter situation is shown in Figure 8.4.

8.3 Input Language

The input language for the MIO Workbench has been designed with Xtext [171], a framework that simplifies the creation of domain-specific languages. We have specified a grammar for which Xtext has generated a parser as well as a text editor with syntax highlighting (described above). The complete grammar for the input language is shown in Section 8.3.2.



Name: T2

Start State: State t0

Actions:

Action Type	Action Name	Used in
Input	coin	t0 -> t1
Output	tea	t1 -> t2
Output	coffee	t1 -> t2
Internal	beep	t2 -> t0

State size: □ Force fixed size of states

Figure 8.4: The graphical MIO editor and the properties view

8.3.1 Short Tutorial on the Input Language

The input language allows to specify MIOs with an intuitive syntax, for instance, the MIO shown in Figure 8.4 can be specified by:

```
mio T2 {
        inputs coin
        outputs coffee, tea
        internals beep
        states t0, t1, t2
        start t0
        mayTransitions
                t1 -> t2 [tea]
        mustTransitions
                t0 -> t1 [coin],
                t1 -> t2 [coffee],
                t2 -> t0 [beep]
}
```

Note that, similarly to the graphical representation, may-transitions underlying must-transitions need not be explicitly mentioned. The assignment

```
S := T
```

defines a new MIO S such that S equals T element-wise. The right hand side of an assignment may use modal synchronous composition $\otimes$ (denoted `||` in the shell), conjunction $\wedge$ (denoted `&&`), quotient $/\!\!/$ (denoted `--`) and weakening $\triangleright$ (denoted `>>`). Moreover, $\mathsf{max}_{T_\rightarrow}(E)$ is expressed by `max(T,E)`. For example, the MIO $S = (T_1 \otimes T_2) /\!\!/ T_3$ can be defined by the command

```
S := (T_1 || T_2) -- T_3
```

The command

```
S <= T
```

checks whether S is a strong modal refinement of T. Weak modal refinement is written `<=*`. The equivalence relations $\approx_s$ and $\approx_w$ induced by strong and weak modal refinement are written `equiv<=` and `equiv<=*`. A strong environment correctness statement $S \rightarrow_s E$ can be checked by executing

```
S -> E
```

Weak environment correctness is written `->*`, and the induced symmetric compatibility notions $\leftrightarrows_s$ and $\leftrightarrows_w$ are written `<-->` and `<-->*`, respectively.

A modal contract $C = (A, G)$ for existing MIOs A and G can be defined by the command

```
C := [ A, G ]
```

Note that, similar to the assignment of MIOs, the assumption and guarantee of `C` are copies of `A` and `G`, respectively, to avoid side-effects when normalizing contracts. New contracts can be constructed by composition or normalization. Contract composition $\boxtimes$, see Definition 7.2.21, is expressed by `||`, e.g. two contracts `C_1` and `C_2` are composed by `C_1 || C_2`. Normalization of a contract `C`, see Theorem 7.3.16, is done by the command `normalize(C)`. For instance, to define a new contract `C` by the normal form of the contract composition of `C_1` and `C_2`, we can type

```
C := normalize(C_1 || C_2)
```

Refinement of two contracts `C_1` and `C_2` is expressed by

```
C_1 <= C_2
```

The are a few other auxiliary commands. The command `list` prints a list of all MIOs and modal contracts that are currently stored in the shell. The command `save S` opens a save dialogue to save the MIO (or modal contract) `S` to a *.mio* (or *.miocontract*) file. There are two different kinds of comments available:

```
// single-line comments
/*
  multi-line comments
*/
```

Finally, the command `view S`, where `S` is the name of a MIO, sends the MIO `S` to the image view where it is depicted using the program *dot* from the Graphviz package [94]. We remark that the `view` command is only available in the shell for technical reasons (in the text editor, a syntax error is indicated by the syntax highlighting).

8.3.2 Grammar of the Input Language

```
Statement     =   ’list’ | ’save’ UId | ’view’ UId
                | Assignment | BinaryCheck | AutomatonDef ;
Assignment    =   UId ’:=’ ( Mio | Contract ) ;
BinaryCheck   =   Mio ( Refinement | EnvCorrect ) Mio
                | Contract ’<=’ Contract ;
Refinement    =   ’<=’ | ’<=*’ | ’equiv<=’ | ’equiv<=*’
EnvCorrect    =   ’->’ | ’->*’ | ’<-->’ | ’<-->*’

Mio           =   UId
                | Contract ’.’ ( ’env’ | ’spec’ )
                | ’(’ Mio Composition Mio ’)’ ;
Composition   =   ’||’ | ’&&’ ;

AutomatonDef  =   ’mio’ UId ’{’
                  ( ’inputs’ LId ( ’,’ LId )* )?
                  ( ’outputs’ LId ( ’,’ LId )* )?
                  ( ’internals’ LId ( ’,’ LId )* )?
                  ’states’ LId ( ’,’ LId )*
                  ’start’ LId
                  ( ’may’ Transition ( ’,’ Transition)* )?
                  ( ’must’ Transition ( ’,’ Transition)* )?
                  ’}’ ;
Transition    =   LId ’->’ LId ’[’ LId ( ’,’ LId)* ’]’ ;

Contract      =   Id | ContractDef ;
ContractDef   =   ’[’ Mio ’,’ Mio ’]’ ;

UId           =   (’A’..’Z’) (’a’..’z’|’A’..’Z’|’_’|’0’..’9’)* ;
LId           =   (’a’..’z’) (’a’..’z’|’A’..’Z’|’_’|’0’..’9’)* ;
```

8.4 Related Work

The MIO Workbench was the first tool that implemented a modal specification theory with operators like conjunction and quotient, with a focus on checking of environment correctness. Unique to the MIO Workbench is the explicit support of modal contracts.

The tool TICC [2] implements the interface theory of sociable interfaces [58], that are interface automata extended with variables, and focuses on interface compatibility and composition. It also supports the model checking of CTL formulae. A very related tool is CHIC [37] that similarly focuses on interface compatibility and composition and supports various interface theories like synchronous assume/guarantee interfaces [62], resource interfaces [52] or web service interfaces [38].

ECDAR [56] builds on UPPAAL [174] and implements the timed interface theory of [57]. The tool supports composition and conjunction and can verify refinement. Interface compatibility is not explicitly supported, but simple timed temporal logic properties (subset of TCTL formulae) can be model checked.

TAPAs [46] is a tool for the specification and the analysis of concurrent system. It supports of a basic process algebra as input language, internally translates process terms to labelled transition systems, and is capable to verify various equivalences as well as to model check μ-calculus formulae. The interesting point at TAPAs is that its aim is to be used as a tool in teaching process algebra. In particular, it supports a representation of processes both as a term and as a graph at the same time while being consistent with each other. Our verification view grew out of a similar idea: make results of checks visible and understandable for beginners.

The most related tool is MTSA [72, 71]. It is based on the LTSA tool [137] and integrated into the Eclipse platform – similar to the MIO Workbench – with a sophisticated GUI. MTSA supports the construction of modal transition systems using an adapted FSP (process algebra) syntax [137]. Although modal refinement can be checked on modal transition systems, MTSA focuses on synthesizing modal transition systems from scenarios/use cases or from requirement specifications expressed in fluent linear temporal logic [87]. Checks for compatibility of specifications like environment correctness are not implemented.

Recently, Mica [44] has been released implementing modal interfaces [158]. In particular, (optimistic) composition, conjunction, quotient as well as refinement are supported. The user can interact with Mica through a shell. Visual representation of modal interfaces is provided by using Graphviz [94], however, the design of modal interfaces can only be done in the shell by textual input.

Tool and publication history. The MIO Workbench [144] was initiated in August 2009 by Philip Mayer and was first published in [27]. This first version of the MIO Workbench already contained strong and weak modal refinement, symmetric compatibility notions $\leftrightarrows_s$ and $\leftrightarrows_w$ [27], the graphical editor and the verification view. Mayer used the MIO Workbench in his thesis [138] for the verification of protocols and orchestrations of web services for which MIOs have been used as a formal semantics. In the course of this thesis the MIO Workbench has been extensively improved by the additional features described above. Some of the improvements were presented in [26], in particular, the shell and the additional MIO operations quotient and conjunction. Since then, the support of modal contracts has been added.

Chapter 9

Conclusion

9.1 Contributions

This thesis provides a comprehensive study of formal behavioural specification of reactive components in the context of modal specification theories. The abstract framework of a specification theory formalizes essential ingredients and properties of any formal theory supporting the compositional design and refinement of component interfaces with a focus on component compatibility. Specification theories have served as the common basis throughout the thesis in order to allow for detailed comparison of the introduced theories by morphisms and embeddings.

We have considered modal input/output automata (MIOs) as a flexible formalism for specifying the behaviour of communicating components. The *may* and *must* modalities for transitions support loose specifications and a stepwise refinement approach developing component-based systems in a top-down manner, from abstract specifications to implementations.

We have defined novel specification theories for MIOs, both a "strong" variant (based on strong modal refinement) and a "weak" observational variant (based on weak modal refinement). To accommodate current needs for rich heterogeneous formalisms for component-based design, we have proposed important extensions of MIOs that take into account data (MIODs) and quantitative aspects ($\mathscr{K}$-WMIOs) and we have shown how to obtain specification theories for these extensions (again distinguishing between "strong" and "weak" variant). Thus we have obtained a hierarchy of specification theories for (deterministic) MIOs, MIODs, $\mathscr{K}$-WMIOs and $\mathscr{K}$-WMIODs, see Figure 9.1, in which all introduced modal specification theories are related to each other by (reflective) embeddings.

In more detail, the hierarchy of modal specification theories in Figure 9.1 is the result of the following key contributions of this thesis:

- The modal specification theories $Th^{\mathbb{MIO}}_{strong}$ and $Th^{d\mathbb{MIO}}_{strong}$ have been inspired by related work in the literature [124, 158] being the starting point of this

$$\begin{tikzcd}
 & & Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{weak} & \\
Th^{\mathbb{MIOD}}_{weak} \arrow[urr, {Hooks[right]}-{Triangle[open]}] & Th^{\mathbb{MIO}}_{weak} \arrow[l, {Hooks[right]}-{Triangle[open]}] \arrow[ur, {Hooks[right]}-{Triangle[open]}] \arrow[rr, {Hooks[right]}-{Triangle[open]}] & & Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak} \arrow[ul, {Hooks[right]}-{Triangle[open]}] \\
 & & Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong} \arrow[uu, hook] & \\
Th^{\mathbb{MIOD}}_{strong} \arrow[uu, hook] \arrow[urr, {Hooks[right]}-{Triangle[open]}] & Th^{\mathbb{MIO}}_{strong} \arrow[uu, hook] \arrow[l, {Hooks[right]}-{Triangle[open]}] \arrow[ur, {Hooks[right]}-{Triangle[open]}] \arrow[rr, {Hooks[right]}-{Triangle[open]}] & & Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[uu, hook] \arrow[ul, {Hooks[right]}-{Triangle[open]}] \\
 & & Th^{d\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong} \arrow[uu, {Hooks[right]}-{Triangle[open]}] & \\
Th^{d\mathbb{MIOD}}_{strong} \arrow[uu, {Hooks[right]}-{Triangle[open]}] \arrow[urr, {Hooks[right]}-{Triangle[open]}] & Th^{d\mathbb{MIO}}_{strong} \arrow[uu, {Hooks[right]}-{Triangle[open]}] \arrow[l, {Hooks[right]}-{Triangle[open]}] \arrow[ur, {Hooks[right]}-{Triangle[open]}] \arrow[rr, {Hooks[right]}-{Triangle[open]}] & & Th^{d\mathcal{K}\text{-}\mathbb{WMIO}}_{strong} \arrow[uu, {Hooks[right]}-{Triangle[open]}] \arrow[ul, {Hooks[right]}-{Triangle[open]}]
\end{tikzcd}$$

Figure 9.1: The hierarchy of modal specification theories introduced in this thesis

thesis. We have shown that in $Th^{d\mathbb{MIO}}_{strong}$ strong modal refinement is equivalent to thorough refinement defined by inclusion of implementation semantics. Furthermore, we have shown in Chapter 7 that $Th^{d\mathbb{MIO}}_{strong}$ is a *complete* specification theory supporting quotient, conjunction and a maximal environment correctness operator; this result has been particularly useful to derive modal contracts with an explicit contract composition operator (see below).

- For obtaining the observational specification theory $Th^{\mathbb{MIO}}_{weak}$ for MIOs with weak modal refinement, we have introduced the novel notion of *weak environment correctness* that has been the solution to the problem that strong environment correctness (from $Th^{\mathbb{MIO}}_{strong}$ and inspired by interface compatibility of interface automata [61]) is not preserved by weak modal refinement. Weak environment correctness requires that the environment guarantees to receive any outputs, possibly after some internal must-transitions.

- We have proposed MIOs with data constraints (MIODs) that enrich MIOs with provided and required state variables. The specification of data has been integrated by adding pre- and postconditions to transitions describing in which data states a transition is enabled and to which next (provided) data states the transition can lead. We extensively studied how modal refinement, environment correctness, and modal synchronous composition can be adapted accordingly to take into account pre- and postconditions.

The result has been the specification theories $Th^{\mathbb{MIOD}}_{strong}$, $Th^{d\mathbb{MIOD}}_{strong}$ and $Th^{\mathbb{MIOD}}_{weak}$ for MIODs. For the specification theories $Th^{\mathbb{MIOD}}_{strong}$, $Th^{d\mathbb{MIOD}}_{strong}$, we have studied two additional topics:

- We have introduced predicate abstraction as a possible verification technique for strong modal refinement for *finite* MIODs but with *infinite* variable domains.
- Moreover, we have proposed a denotational semantics of implementations of MIODs, given by input/output automata with data states (IODs) that are *implementation models* with provided data states in each state and required data states on transitions. We have shown that the denotational semantics preserves both strong environment correctness and synchronous composition.

- We have also investigated MIOs extended with transition weights from an abstract, partially ordered weight structure $\mathcal{K}$. The obtained $\mathcal{K}$-weighted MIOs ($\mathcal{K}$-WMIOs) are capable of expressing constraints on non-functional properties such as resource consumption (power, fuel) or costs. We studied both strong and weak modal refinement for $\mathcal{K}$-WMIOs in which (additionally to the usual modal refinement of may- and must-transitions) transition weights can be refined by the partial order of $\mathcal{K}$. The result has been the specification theories $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{strong}$, $Th^{d\mathcal{K}\text{-}\mathbb{WMIO}}_{strong}$ and $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{weak}$ for $\mathcal{K}$-WMIOs.

  The notion of strong modal refinement of $\mathcal{K}$-WMIOs, that has been used for the definition of the above modal specification theories, can be generalized to modal refinement distances. We have studied how strong modal refinement can be lifted to modal refinement distances in the context of the weight structure $\mathcal{K}_{intv}$ of integer intervals. Modal refinement distances are functions from pairs of $\mathcal{K}_{intv}$-WMIOs to $\mathbb{R}_{\geq 0}$ measuring how "well" the refinement holds (with 0 meaning that strong modal refinement holds). We have shown how compositionality can be reformulated for modal refinement distances.

- The specification theories $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$, $Th^{d\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$ and $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{weak}$ have been the result of the integration of data aspects and quantitative aspects into a single formalism.

On the abstract level of specification theories we have studied component contracts as a useful methodology in component-based design. Contracts explicitly distinguish between assumptions on the environment and guarantees of a component, strictly following the principle of separation of concerns. We have shown how one can build a contract theory in a generic way on top of any specification theory leading to generic definitions of contract semantics, contract refinement and contract dominance which defines necessary conditions for a contract to describe the structural combination of two contracts. For *complete* specification theories that feature conjunction, quotient and a maximal environment correctness operator, we have shown how to obtain a contract composition, i.e. a strongest dominating contract. Thus, for any instance of a (complete) specification theory one can obtain a corresponding contract theory "for free". This has been exemplified by instantiating the generic contract framework for the complete specification theory $Th^{d\mathbb{MIO}}_{strong}$ and for the specification theory $Th^{d\mathbb{MIOD}}_{strong}$, giving rise to *modal contracts*. Several examples have shown that modalities turn out to be very convenient for formulating loose environment assumptions.

Finally, the MIO Workbench is a tool for the verification of MIOs and modal contracts, implementing the specification theories $Th^{\mathbb{MIO}}_{strong}$, $Th^{d\mathbb{MIO}}_{strong}$, $Th^{\mathbb{MIO}}_{weak}$ as well as modal contracts obtained by instantiating the contract framework for $Th^{d\mathbb{MIO}}_{strong}$. The MIO Workbench is based on the Eclipse framework [75] and features an intuitive and easy-to-use graphical user interface. It supports a graphical MIO editor and modal contract editor, a graphical verification view for illustration of modal refinement and environment correctness checks, and features an expressive input language for the execution of verification tasks.

9.2 Discussion and Future Work

In this section we discuss findings together with future research directions on the basis of this thesis.

Specification theories, component languages and assemblies. The algebraic framework of a specification theory provides a common abstraction for all the formal theories introduced in this thesis. Specification theories have been inspired by de Alfaro and Henzinger's work on interface theories [62, 73]. We have distinguished between specification theory embeddings and reflective embeddings. Further work on this abstract framework may include a formal definition of products of specification theories to form new specification theories from existing ones, similar to what we have already done in this thesis: combining $Th^{\mathbb{MIOD}}_{strong}$ with $Th^{\mathcal{K}\text{-}\mathbb{WMIO}}_{strong}$ resulting in the specification theory $Th^{\mathcal{K}\text{-}\mathbb{WMIOD}}_{strong}$.

In the future more fine-grained features of component-based systems like signatures, channels, ports or connectors should be neatly integrated into the abstract framework of specification theories, similar to the work of Janisch [112] or the work of Allen and Garlan on Wright [4]. Furthermore, specification theories should explicitly support assemblies of components, with notions of assembly refinement and assembly compatibility. A first step into this direction has been developed by Hennicker and Knapp [102].

Modalities and variability. The formal theories presented in this thesis are based on variants of modal transition systems. The use of modalities in modelling the behaviour of component interfaces offers a greater specification variability and essentially increases the power of stepwise refinement from abstract specifications towards implementations.[1] However, modal transition systems only provide means to specify local variability, in the sense that a transition may be implemented or dropped in a later refinement step. There are no global constraints on these choices possible, e.g. one might want to express that whenever one optional feature is implemented, then the other feature has to be there, too. Another example is the persistence of choices which cannot be expressed in modal transition systems, i.e. if we choose to implement an optional behaviour, we should do so whenever we reach that same state again. Recently, Beneš et al. [30] introduced parametric modal transition systems that try to resolve this problem by allowing constraints on parameters that regulate the possible choices of transitions to be implemented.

It would be interesting to integrate such extensions with respect to variability aspects into our work. We believe that environment correctness can be integrated in a straightforward way. Whether weak modal refinement can be defined, for instance for parametric modal transition systems [30], is an open issue.

Communication and compatibility. With regard to communication and compatibility of component interfaces (in this thesis defined via environment correctness), we have restricted our specification theories to modal transition systems with input, output and internal actions with binary synchronous communication only, i.e. component interfaces are composed by synchronizing on matching inputs and outputs. Environment correctness is closely connected to this communication scheme, and requires that whenever an output action is enabled, then the environment must be able to receive that output. This basic idea of environment correctness has also been at the core of compatibility in the related approach of interface automata [61], however, with the different *optimistic* view on compatibility.

More general communication schemes and associated compatibility notions are clearly desirable. We believe that adding the support for arbitrary synchronization schemes like in the BIP framework [40] or in Reo [8], in particular allowing $1 : n$ communication like broadcast, would be a valuable effort. As compatibility is always tightly related to the desired communication scheme, a more general approach to communication errors is needed then. One possibility would be to lift the notion of stuck-freedom [86] of Fournet et al.– originally developed for CCS [142] – to the level of modal transition systems. Stuck-freedom is a kind of deadlock freedom, however, each process (or interface in our case) should not

[1] Modelling with variability essentially follows the idea of loose algebraic specifications, see e.g. [162].

get stuck (or deadlock) as long as external actions are enabled meaning that the process is still willing to interact with its environment. We claim that output-compatibility for certain subclasses of MIOs could be equally analyzed with the more general approach using stuck-freedom. An alternative would be to use logics like the modal μ-calculus [118] interpreted over modal transition systems like in [96]. Generalizing the satisfaction relation of [96] to a *relativized* satisfaction relation taking into account a given context would allow to express environment correctness and other compatibility notions by suitable formulae. Of course, refinement of the specification or the context would need to preserve the relativized satisfaction relation. It would also be interesting to integrate a logic into our abstract framework of a specification theory.

Another interesting direction of research is the integration of asynchronous compatibility. In [112, 20, 101] a specification theory was developed that defines asynchronous compatibility in the context of parallel composition in which component interfaces communicate via buffers. In principle, all of the introduced variants of MIOs in this thesis have to be reconsidered and studied to figure out how they work with asynchronous communication. In particular, it would be interesting to study asynchronous environment correctness for MIODs which was inherently synchronous so far: in the denotational semantics of implementations of MIODs, a transition labelled with $[v]\alpha(\rho)$ is executed if v is the data state of the environment and the environment offers a corresponding transition for α. So far it was assumed that the check for the environment's data state and the execution of α happens in one atomic step – which in the presence of asynchronous communication would not be possible anymore.

Modal refinement distances. We have presented a quantitative approach to system specification based on K_{intv}-WMIOs and modal refinement distances. One possible direction for future work is the integration of a suitable temporal logic that can express properties on the weights that are visited during a system run, e.g. one would like to decide whether there is a run such that the accumulated resource consumption stays below a given bound. A preliminary step is presented in our recent work [24] where we propose a subset of CTL extended with propositions reasoning about accumulated weights, and show various (un-)decidability results for the logic.

Contracts. In Chapter 7, we have shown how to derive a contract theory for a given (complete) specification theory. In particular, we required Assumption 4 to hold: whenever $S_1 \to S_2 \otimes E$ and $S_2 \to S_1 \otimes E$, then $S_1 \otimes S_2 \to E$. Specification theories for MIOs with weak environment correctness and weak modal refinement are neither complete nor satisfy Assumption 4. To overcome these problems, we believe that one has to integrate more structural elements into a contract and arrive at component contracts that consist of a component behaviour (or component frame) F together with a set of port contracts (A_P, G_P) for each port P that describe the assumptions and guarantees at this specific interaction point of the component. This extension is clearly closely related with the previously mentioned issue of integrating structural component concepts into specification theories. A first step into this direction is presented in our recent work [21], however, there is still missing a notion of dominance that semantically formulates requirements of contract compositions, and hence the formal treatment of deriving most permissive assumptions for proper contract compositions is to be added.

Tool support. So far, the MIO Workbench provides tool support for the specification theories $Th^{\mathbb{MIO}}_{strong}$, $Th^{d\mathbb{MIO}}_{strong}$, $Th^{\mathbb{MIO}}_{weak}$ as well as modal contracts obtained by instantiating the contract framework for $Th^{d\mathbb{MIO}}_{strong}$. Clearly, all other modal specification theories of Figure 9.1 should be added, including a support for modal refinement distances for $\mathcal{K}_{intv}$-WMIOs. The implementation of the modal specification theories for MIODs, in particular predicate abstract for strong modal refinement, would be very worthwhile in order to model and work with larger case studies. Error reporting for modal refinement and environment correctness for (small) *deterministic* MIOs is conveniently realized by the verification view. However, for *non-deterministic* MIOs, the verification view cannot handle negative answers in a satisfactory manner. For instance, if a modal refinement check for non-deterministic MIOs is negative, the verification view can only present a path to a state pair that does not satisfy the conditions required by the refinement relation. As MIOs are in general non-deterministic, there could be many such paths that are better illustrated by a *modal refinement game*: the user should be able to play the modal refinement game (possibly directly on the graphical representation of MIOs) against the computer that uses a winning strategy for showing that the modal refinement does not hold.

Moreover, integrating hierarchical models, assemblies of specifications, in the sense of [102], and finally port-based components is also desirable. We strongly believe that having a sophisticated tool support is an indispensable prerequisite for handling large, industrial case studies.

Software produce lines. In a larger context, modal transition systems have gained a lot of interest in the software product lines community. Software product lines or software product families are sets of related software products with many similarities but varying functionality. Software product line engineering is often used as a term for the process of managing the development of software product families, see [152] for an overview. For instance, when embedded systems are realized with similar hardware devices the software usually differs only in some variation points depending on the functionality of the hardware device used. Over the last years, modal transition systems have been used to model behavioural variability with a prospective application to software product line modelling [84, 124, 77, 78]. This line of research is of high practical interest with a promising application area for all the modal specification theories studied in this thesis.

Bibliography

[1] J. Adamek and F. Plasil. Behavior protocols capturing errors and updates. In *Proceedings of the Second International Workshop on Unanticipated Software Evolution (USE 2003)*, page 17–25. University of Warsaw, Poland, 2003. Online available: `http://www.iai.uni-bonn.de/~gk/use/2003/Papers/18500032.pdf`, last accessed on August 27, 2012.

[2] B. T. Adler, L. de Alfaro, L. D. da Silva, M. Faella, A. Legay, V. Raman, and P. Roy. Ticc: A Tool for Interface Compatibility and Composition. In T. Ball and R. B. Jones, editors, *18th International Conference on Computer Aided Verification (CAV 2006)*, volume 4144 of *Lecture Notes in Computer Science*, pages 59–62. Springer, 2006.

[3] R. P. Agarwal, M. Meehan, and D. O'Regan. *Fixed Point Theory and Applications*. Cambridge Tracts in Mathematics. Cambridge University Press, 2009.

[4] R. Allen and D. Garlan. A formal basis for architectural connection. *ACM Trans. Softw. Eng. Methodol.*, 6(3):213–249, 1997.

[5] R. Alur and D. L. Dill. The Theory of Timed Automata. In J. de Bakker, C. Huizing, W. P. de Roever, and G. Rozenberg, editors, *Real Time: Theory in Practice*, volume 600 of *Lecture Notes in Computer Science*, pages 45–74. Springer, Berlin, 1991.

[6] A. Antonik and M. Huth. Efficient Patterns for Model Checking Partial State Spaces in CTL intersection LTL. *Electronic Notes in Theoretical Computer Science*, 158(0):41 – 57, 2006.

[7] A. Antonik, M. Huth, K. G. Larsen, U. Nyman, and A. Wasowski. 20 years of modal and mixed specifications. *Bulletin of the EATCS*, 95:94–129, 2008.

[8] F. Arbab. Reo: a channel-based coordination model for component composition. *Mathematical Structures in Computer Science*, 14(3):329–366, 2004.

[9] F. Arbab and J. J. M. M. Rutten. A coinductive calculus of component connectors. In M. Wirsing, D. Pattinson, and R. Hennicker, editors, *16th International Workshop on Recent Trends in Algebraic Development Techniques (WADT 2002)*, volume 2755 of *Lecture Notes in Computer Science*, pages 34–55. Springer, 2002.

[10] P. Asirelli, M. H. ter Beek, S. Gnesi, and A. Fantechi. Formal description of variability in product families. In E. S. de Almeida, T. Kishi, C. Schwanninger, I. John, and K. Schmid, editors, *15th International Conference on Software Product Lines (SPLC 2011)*, pages 130–139. IEEE, 2011.

[11] C. Baier, M. Sirjani, F. Arbab, and J. J. M. M. Rutten. Modeling component connectors in Reo by constraint automata. *Sci. Comput. Program.*, 61(2):75–113, 2006.

[12] T. Barros, R. Ameur-Boulifa, A. Cansado, L. Henrio, and E. Madelaine. Behavioural models for distributed fractal components. *Annales des Télécommunications*, 64(1-2):25–43, 2009.

[13] S. Bauer, A. David, R. Hennicker, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. Moving from specifications to contracts in component-based design. In J. de Lara and A. Zisman, editors, *15th International Conference on Fundamental Approaches to Software Engineering (FASE 2012)*, volume 7212 of *Lecture Notes in Computer Science*, pages 43–58. Springer, 2012.

[14] S. Bauer, U. Fahrenberg, L. Juhl, K. G. Larsen, A. Legay, and C. R. Thrane. Quantitative refinement for weighted modal transition systems. In *36th International Symposium on Mathematical Foundations of Computer Science (MFCS 2011)*, volume 6907 of *Lecture Notes in Computer Science*, pages 60–71. Springer, 2011.

[15] S. Bauer, U. Fahrenberg, L. Juhl, K. G. Larsen, A. Legay, and C. R. Thrane. Weighted modal transition systems. *Formal Methods in System Design*, 2012. To appear.

[16] S. Bauer, U. Fahrenberg, A. Legay, and C. Thrane. General quantitative specification theories with modalities. In *7th International Computer Science Symposium in Russia (CSR 2012)*, volume 7353 of *Lecture Notes in Computer Science*, pages 18–30. Springer, 2012.

[17] S. Bauer and R. Hennicker. Views on behaviour protocols and their semantic foundation. In *Third International Conference on Algebra and Coalgebra in Computer Science (CALCO 2009)*, volume 5728 of *Lecture Notes in Computer Science*, pages 367–382. Springer, 2009.

[18] S. Bauer, R. Hennicker, and M. Bidoit. A modal interface theory with data constraints. In J. Davies, L. Silva, and A. da Silva Simão, editors, *13th Brazilian Symposium on Formal Methods (SBMF 2010), Revised Selected Papers*, volume 6527 of *Lecture Notes in Computer Science*, pages 80–95. Springer, 2011.

[19] S. Bauer, R. Hennicker, and S. Janisch. Behaviour protocols for interacting stateful components. *Electr. Notes Theor. Comput. Sci.*, 263:47–66, 2010.

[20] S. Bauer, R. Hennicker, and S. Janisch. Interface Theories for (A)synchronously Communicating Modal I/O-Transition Systems. In *Third Workshop on Foundations for Interface Technologies (FIT 2010)*, volume 46 of *Electronic Proceedings in Theoretical Computer Science*, pages 1–8, 2010.

[21] S. Bauer, R. Hennicker, and A. Legay. Component interfaces with contracts on ports. In *9th International Symposium on Formal Aspects of Component Software (FACS 2012)*, Lecture Notes in Computer Science. Springer, 2012. To appear.

[22] S. Bauer, R. Hennicker, and M. Wirsing. Interface theories for concurrency and data. *Theor. Comput. Sci.*, 412(28):3101–3121, 2011.

[23] S. Bauer, L. Juhl, K. G. Larsen, A. Legay, and J. Srba. Extending modal transition systems with structured labels. *Mathematical Structures in Computer Science*, 22(4):581–617, 2012.

[24] S. Bauer, L. Juhl, K. G. Larsen, J. Srba, and A. Legay. A logic for accumulated-weight reasoning on multiweighted modal automata. In T. Margaria, Z. Qiu, and H. Yang, editors, *6th International Symposium on Theoretical Aspects of Software Engineering (TASE 2012)*, pages 77–84. IEEE, 2012.

[25] S. Bauer, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. A modal specification theory for components with data. In *8th International Symposium on Formal Aspects of Component Software (FACS 2011)*, Lecture Notes in Computer Science. Springer, 2011. To appear.

[26] S. Bauer, P. Mayer, and A. Legay. MIO Workbench: A Tool for Compositional Design with Modal Input/Output Interfaces. In T. Bultan and P.-A. Hsiung, editors, *9th International Symposium on Automated Technology for Verification and Analysis (ATVA 2011)*, volume 6996 of *Lecture Notes in Computer Science*, pages 418–421. Springer, 2011.

[27] S. Bauer, P. Mayer, A. Schroeder, and R. Hennicker. On Weak Modal Compatibility, Refinement, and the MIO Workbench. In *16th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2010)*, volume 6015 of *Lecture Notes in Computer Science*, pages 175–189. Springer, 2010.

[28] N. Beneš, I. Cerná, and J. Kretínský. Modal Transition Systems: Composition and LTL Model Checking. In T. Bultan and P.-A. Hsiung, editors, *9th International Symposium on Automated Technology for Verification and Analysis (ATVA 2011)*, volume 6996 of *Lecture Notes in Computer Science*, pages 228–242. Springer, 2011.

[29] N. Beneš and J. Křetínský. Process algebra for modal transition systemses. In L. Matyska, M. Kozubek, T. Vojnar, P. Zemcík, and D. Antos, editors, *MEMICS*, volume 16 of *OASICS*, pages 9–18. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2010.

[30] N. Beneš, J. Křetínský, K. Larsen, M. Møller, and J. Srba. Parametric modal transition systems. In *Proceedings of the 9th International Symposium on Automated Technology for Verification and Analysis (ATVA 2011)*, volume 6996 of *Lecture Notes in Computer Science*, pages 275–289. Springer-Verlag, 2011.

[31] N. Beneš, J. Křetínský, K. G. Larsen, and J. Srba. Checking thorough refinement on modal transition systems is EXPTIME-complete. In *6th International Colloquium on Theoretical Aspects of Computing (ICTAC 2009)*, volume 5684 of *Lecture Notes in Computer Science*, pages 112–126. Springer, 2009.

[32] N. Beneš, J. Křetínský, K. G. Larsen, and J. Srba. On determinism in modal transition systems. *Theoretical Computer Science*, 410(41):4026–4043, 2009.

[33] A. Benveniste, B. Caillaud, A. Ferrari, L. Mangeruca, R. Passerone, and C. Sofronis. Multiple viewpoint contract-based specification and design. In *6th International Symposium on Formal Methods for Components and Objects (FMCO 2007)*, volume 5382 of *Lecture Notes in Computer Science*, pages 200–225. Springer, 2007.

[34] J. A. Bergstra and J. W. Klop. Process algebra for synchronous communication. *Information and Control*, 60(1-3):109–137, 1984.

[35] N. Bertrand, A. Legay, S. Pinchinat, and J.-B. Raclet. A compositional approach on modal specifications for timed systems. In K. Breitman and

A. Cavalcanti, editors, *11th International Conference on Formal Engineering Methods (ICFEM 2009)*, volume 5885 of *Lecture Notes in Computer Science*, pages 679–697. Springer, 2009.

[36] N. Bertrand, S. Pinchinat, and J.-B. Raclet. Refinement and consistency of timed modal specifications. In A. H. Dediu, A.-M. Ionescu, and C. Martín-Vide, editors, *Third International Conference on Language and Automata Theory and Applications (LATA 2009)*, volume 5457 of *Lecture Notes in Computer Science*, pages 152–163. Springer, 2009.

[37] D. Beyer, A. Chakrabarti, K. Chatterjee, L. de Alfaro, T. Henzinger, M. Jurdzinski, F. Mang, and C. Song. CHIC: Checking Interface Compatibility. Technical report, UC Berkeley, December 2007.

[38] D. Beyer, A. Chakrabarti, and T. A. Henzinger. Web service interfaces. In A. Ellis and T. Hagino, editors, *14th International Conference on World Wide Web (WWW 2005)*, pages 148–159. ACM, 2005.

[39] M. Bidoit, R. Hennicker, A. Knapp, and H. Baumeister. Glass-box and black-box views on object-oriented specifications. In *Second IEEE International Conference on Software Engineering and Formal Methods (SEFM 2004)*, pages 208–217. IEEE Computer Society Press, 2004.

[40] S. Bliudze and J. Sifakis. The algebra of connectors: structuring interaction in BIP. In C. M. Kirsch and R. Wilhelm, editors, *7th ACM & IEEE International conference on Embedded software (EMSOFT 2007)*, pages 11–20. ACM, 2007.

[41] E. Bruneton, T. Coupaye, M. Leclercq, V. Quéma, and J.-B. Stefani. An Open Component Model and Its Support in Java. In I. Crnkovic, J. A. Stafford, H. W. Schmidt, and K. C. Wallnau, editors, *7th International Symposium on Component-Based Software Engineering (CBSE 2004)*, volume 3054 of *Lecture Notes in Computer Science*, pages 7–22. Springer, 2004.

[42] G. Bruns and P. Godefroid. Generalized model checking: Reasoning about partial state spaces. In C. Palamidessi, editor, *11th International Conference on Concurrency Theory (CONCUR 2000)*, volume 1877 of *Lecture Notes in Computer Science*, pages 168–182. Springer, 2000.

[43] T. Bures, M. Decký, P. Hnetynka, J. Kofron, P. Parizek, F. Plasil, T. Poch, O. Sery, and P. Tuma. CoCoME in SOFA. In A. Rausch, R. Reussner, R. Mirandola, and F. Plasil, editors, *The Common Component Modeling Example: Comparing Software Component Models*, volume 5153 of *Lecture Notes in Computer Science*, pages 388–417. Springer, 2007.

[44] B. Caillaud. Mica – a modal interface compositional analysis library, 2011. Online available: `http://www.irisa.fr/s4/tools/mica/`, last accessed on August 27, 2012.

[45] B. Caillaud, B. Delahaye, K. G. Larsen, A. Legay, M. L. Pedersen, and A. Wasowski. Constraint markov chains. *Theor. Comput. Sci.*, 412(34):4373–4404, 2011.

[46] F. Calzolai, R. D. Nicola, M. Loreti, and F. Tiezzi. Tapas: A tool for the analysis of process algebras. *T. Petri Nets and Other Models of Concurrency*, 1:54–70, 2008.

[47] C. Carrez, A. Fantechi, and E. Najm. Behavioural contracts for a sound composition of components. In H. König, M. Heiner, and A. Wolisz, editors, *Formal Techniques for Networked and Distributed Systems (FORTE 2003)*, volume 2767 of *Lecture Notes in Computer Science*, pages 111–126. Springer, Berlin, Germany, Sept. 2003.

[48] C. Carrez, A. Fantechi, and E. Najm. Assembling components with behavioural contracts. *Annales des Télécommunications*, 60(7-8):989–1022, 2005.

[49] K. Cerans, J. C. Godskesen, and K. G. Larsen. Timed modal specification - theory and tools. In C. Courcoubetis, editor, *5th International Conference on Computer Aided Verification (CAV 1993)*, volume 697 of *Lecture Notes in Computer Science*, pages 253–267. Springer, 1993.

[50] I. Cerná, P. Vareková, and B. Zimmerova. Component substitutability via equivalencies of component-interaction automata. *Electr. Notes Theor. Comput. Sci.*, 182:39–55, 2007.

[51] P. Cerný, T. A. Henzinger, and A. Radhakrishna. Simulation distances. *Theor. Comput. Sci.*, 413(1):21–35, 2012.

[52] A. Chakrabarti, L. de Alfaro, T. A. Henzinger, and M. Stoelinga. Resource interfaces. In *Third International Conference on Embedded Software (EMSOFT 2003)*, volume 2855 of *Lecture Notes in Computer Science*, pages 117–133. Springer, 2003.

[53] A. Classen, P. Heymans, P.-Y. Schobbens, A. Legay, and J.-F. Raskin. Model checking lots of systems: efficient verification of temporal properties in software product lines. In J. Kramer, J. Bishop, P. T. Devanbu, and S. Uchitel, editors, *32nd ACM/IEEE International Conference on Software Engineering - Proceedings Volume 1*, pages 335–344. ACM, 2010.

[54] I. Claßen. Revised ACT ONE: Categorical Constructions for an Algebraic Specification Language. In H. Ehrig, H. Herrlich, H.-J. Kreowski, and G. Preuß, editors, *Categorial Methods in Computer Science*, volume 393 of *Lecture Notes in Computer Science*, pages 124–141. Springer, 1988.

[55] P. Clements and L. Northrop. *Software Product Lines: Practices and Patterns*. Addison-Wesley, 2001.

[56] A. David, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. Ecdar: An environment for compositional design and analysis of real time systems. In A. Bouajjani and W.-N. Chin, editors, *8th International Symposium on Automated Technology for Verification and Analysis (ATVA 2010)*, volume 6252 of *Lecture Notes in Computer Science*, pages 365–370, 2010.

[57] A. David, K. G. Larsen, A. Legay, U. Nyman, and A. Wasowski. Timed I/O automata: a complete specification theory for real-time systems. In *13th ACM International Conference on Hybrid Systems: Computation and Control (HSCC 2010)*, pages 91–100. ACM, 2010.

[58] L. de Alfaro, L. D. da Silva, M. Faella, A. Legay, P. Roy, and M. Sorea. Sociable interfaces. In *5th International Conference on Frontiers of Combining Systems (FROCOS 2005)*, volume 3717 of *Lecture Notes in Computer Science*, pages 81–105. Springer, 2005.

[59] L. de Alfaro, M. Faella, and M. Stoelinga. Linear and branching metrics for quantitative transition systems. In J. Díaz, J. Karhumäki, A. Lepistö, and D. Sannella, editors, *31st International Colloquium on Automata, Languages and Programming (ICALP 2004)*, volume 3142 of *Lecture Notes in Computer Science*, pages 97–109. Springer, 2004.

[60] L. de Alfaro, M. Faella, and M. Stoelinga. Linear and branching system metrics. *IEEE Trans. Software Eng.*, 35(2):258–273, 2009.

[61] L. de Alfaro and T. A. Henzinger. Interface automata. In *8th European Software Engineering Conference held jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 109–120. ACM Press, 2001.

[62] L. de Alfaro and T. A. Henzinger. Interface theories for component-based design. In *First International Workshop on Embedded Software (EMSOFT 2001)*, volume 2211 of *Lecture Notes in Computer Science*, pages 148–165. Springer, 2001.

[63] L. de Alfaro and T. A. Henzinger. Interface-based Design. In M. Broy, J. Grünbauer, D. Harel, and C. A. R. Hoare, editors, *Engineering Theories*

of Software-intensive Systems, volume 195 of *NATO Science Series: Mathematics, Physics, and Chemistry*, pages 83–104. Springer, 2005.

[64] L. de Alfaro, T. A. Henzinger, and R. Majumdar. Discounting the future in systems theory. In J. C. M. Baeten, J. K. Lenstra, J. Parrow, and G. J. Woeginger, editors, *30th International Colloquium on Automata, Languages and Programming (ICALP 2003)*, volume 2719 of *Lecture Notes in Computer Science*, pages 1022–1037. Springer, 2003.

[65] L. de Alfaro, T. A. Henzinger, and M. I. A. Stoelinga. Timed interfaces. In *Second International Conference on Embedded Software (EMSOFT 2002)*, volume 2491 of *Lecture Notes in Computer Science*, pages 108–122. Springer, 2002.

[66] L. de Alfaro and R. Majumdar. Quantitative solution of omega-regular games. In J. S. Vitter, P. G. Spirakis, and M. Yannakakis, editors, *33rd Annual ACM Symposium on Theory of Computing (STOC 2001)*, pages 675–683. ACM, 2001.

[67] B. Delahaye, B. Caillaud, and A. 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, 2011.

[68] B. Delahaye, J.-P. Katoen, K. G. Larsen, A. Legay, M. L. Pedersen, F. Sher, and A. Wasowski. Abstract probabilistic automata. In *12th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI 2011)*, volume 6538 of *Lecture Notes in Computer Science*, pages 324–339. Springer, 2011.

[69] B. Delahaye, K. Larsen, A. Legay, and A. Wasowski. On greatest lower bound of modal transition systems. Technical report, INRIA, 2010. Online available: `http://delahaye.benoit.free.fr/rapports/modalIPL.pdf`, last accessed on August 27, 2012.

[70] E. W. Dijkstra. Guarded commands, nondeterminacy and formal derivation of programs. *Commun. ACM*, 18(8):453–457, 1975.

[71] N. D'Ippolito, D. Fischbein, M. Chechik, and S. Uchitel. MTSA: The Modal Transition System Analyser. In *23rd International Conference on Automated Software Engineering (ASE 2008)*, pages 475–476. IEEE, 2008.

[72] N. D'Ippolito, D. Fischbein, H. Foster, and S. Uchitel. MTSA: Eclipse support for modal transition systems construction, analysis and elaboration. In L.-T. Cheng, A. Orso, and M. P. Robillard, editors, *OOPSLA Workshop Eclipse Technology eXchange (ETX 2007)*, pages 6–10. ACM Press, 2007.

[73] L. Doyen, T. A. Henzinger, B. Jobstmann, and T. Petrov. Interface theories with component reuse. In L. de Alfaro and J. Palsberg, editors, *8th International Conference on Embedded software (EMSOFT 2008)*, pages 79–88. ACM Press, 2008.

[74] M. Droste, W. Kuich, and H. Vogler. *Handbook of Weighted Automata*. Monographs in Theoretical Computer Science, EATCS, 2009.

[75] Eclipse. `http://www.eclipse.org/`.

[76] M. Emmi, D. Giannakopoulou, and C. S. Pasareanu. Assume-guarantee verification for interface automata. In J. Cuéllar, T. S. E. Maibaum, and K. Sere, editors, *15th International Symposium on Formal Methods (FM 2008)*, volume 5014 of *Lecture Notes in Computer Science*, pages 116–131. Springer, 2008.

[77] A. Fantechi and S. Gnesi. A behavioural model for product families. In I. Crnkovic and A. Bertolino, editors, *Proceedings of 6th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 521–524. ACM, 2007.

[78] A. Fantechi, S. Gnesi, A. Lapadula, F. Mazzanti, R. Pugliese, and F. Tiezzi. A model checking approach for verifying COWS specifications. In J. L. Fiadeiro and P. Inverardi, editors, *11th International Conference on Fundamental Approaches to Software Engineering (FASE 2008)*, volume 4961 of *Lecture Notes in Computer Science*, pages 230–245. Springer, 2008.

[79] H. Fecher and H. Schmidt. Comparing disjunctive modal transition systems with an one-selecting variant. *Journal of Logic and Algebraic Programming*, 77(1-2):20–39, 2008.

[80] F. Fernandes and J.-C. Royer. The STSLib project: Towards a formal component model based on STS. *Electr. Notes Theor. Comput. Sci.*, 215:131–149, 2008.

[81] J. L. Fiadeiro and T. S. E. Maibaum. Interconnecting formalisms: Supporting modularity, reuse and incrementality. In *Third ACM SIGSOFT Symposium on Foundations of Software Engineering*, pages 72–80. ACM, 1995.

[82] J. L. Fiadeiro and T. S. E. Maibaum. Categorical semantics of parallel program design. *Sci. Comput. Program.*, 28(2-3):111–138, 1997.

[83] D. Fischbein, V. A. Braberman, and S. Uchitel. A sound observational semantics for modal transition systems. In *6th International Colloquium on Theoretical Aspects of Computing (ICTAC 2009)*, volume 5684 of *Lecture Notes in Computer Science*, pages 215–230. Springer, 2009.

[84] D. Fischbein, S. Uchitel, and V. Braberman. A foundation for behavioural conformance in software product line architectures. In *Proceedings of the ISSTA 2006 Workshop on Role of Software Architecture for Testing and Analysis (ROSATEA'06)*, pages 39–48. ACM, 2006.

[85] C. Fischer. CSP-OZ: A combination of Object-Z and CSP. In H. Bowman and J. Derrick, editors, *Formal Methods for Open Object-Based Distributed Systems (FMOODS)*, pages 423–438, Canterbury, UK, 1997. Chapman and Hall, London.

[86] C. Fournet, C. A. R. Hoare, S. K. Rajamani, and J. Rehof. Stuck-free conformance. In R. Alur and D. Peled, editors, *16th International Conference on Computer Aided Verification (CAV 2004)*, volume 3114 of *Lecture Notes in Computer Science*, pages 242–254. Springer, 2004.

[87] D. Giannakopoulou and J. Magee. Fluent model checking for event-based systems. In *Proceedings of the 11th ACM SIGSOFT Symposium on Foundations of Software Engineering 2003 held jointly with 9th European Software Engineering Conference*, pages 257–266. ACM, 2003.

[88] R. v. Glabbeek. The linear time – branching time spectrum II; the semantics of sequential systems with silent moves. Manuscript. Preliminary version available at `http://boole.stanford.edu/pub/spectrum.ps.gz`, 1993. Extended abstract in E. Best, editor: Proceedings *CONCUR 1993*, 4th International Conference on Concurrency Theory, Hildesheim, Germany, August 1993, Lecture Notes in Computer Science, volume 715, Springer, pp. 66–81.

[89] P. Godefroid, M. Huth, and R. Jagadeesan. Abstraction-based model checking using modal transition systems. In K. G. Larsen and M. Nielsen, editors, *12th International Conference on Concurrency Theory (CONCUR 2001)*, volume 2154 of *Lecture Notes in Computer Science*, pages 426–440. Springer, 2001.

[90] G. Goessler and J.-B. Raclet. Modal contracts for component-based design. In *Seventh IEEE International Conference on Software Engineering and Formal Methods (SEFM 2009)*, pages 295–303. IEEE Computer Society, 2009.

[91] J. Goguen and R. Burstall. Institutions: abstract model theory for specification and programming. *Journal of ACM*, 39(1):95–146, 1992.

[92] J. A. Goguen and R. M. Burstall. Introducing institutions. In E. M. Clarke and D. Kozen, editors, *Logic of Programs*, volume 164 of *Lecture Notes in Computer Science*, pages 221–256. Springer, 1983.

[93] S. Graf and H. Saïdi. Construction of abstract state graphs with pvs. In O. Grumberg, editor, *9th International Conference on Computer Aided Verification (CAV 1997)*, volume 1254 of *Lecture Notes in Computer Science*, pages 72–83. Springer, 1997.

[94] Graphviz - Graph Visualization Software, 2012. `http://www.graphviz.org/`.

[95] J. F. Groote and A. Ponse. Proof theory for mucrl: A language for processes with data. In D. J. Andrews, J. F. Groote, and C. A. Middelburg, editors, *Semantics of Specification Languages*, Workshops in Computing, pages 232–251. Springer, 1993.

[96] O. Grumberg, M. Lange, M. Leucker, and S. Shoham. *Don't Know* in the μ-Calculus. In R. Cousot, editor, *6th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI 2005)*, volume 3385 of *Lecture Notes in Computer Science*, pages 233–249. Springer, 2005.

[97] R. J. Hall. Feature interactions in electronic mail. In M. Calder and E. H. Magill, editors, *Feature Interactions in Telecommunications and Software Systems VI*, pages 67–82. IOS Press, 2000.

[98] D. Harel. Statecharts: A visual formulation for complex systems. *Sci. Comput. Program.*, 8(3):231–274, 1987.

[99] D. Harel and A. Pnueli. On the development of reactive systems. In K. R. Apt, editor, *Logic and Model of Concurrent Systems*, volume 13 of *NATO ASI*, pages 477–498. Springer, Oct. 1985.

[100] M. Henessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of ACM*, pages 137–161, 1985.

[101] R. Hennicker, S. Janisch, and A. Knapp. On the observable behaviour of composite components. *Electr. Notes Theor. Comput. Sci.*, 260:125–153, 2010.

[102] R. Hennicker and A. Knapp. Modal interface theories for communication-safe component assemblies. In A. Cerone and P. Pihlajasaari, editors, *8th International Colloquium on Theoretical Aspects of Computing (ICTAC*

2011), volume 6916 of *Lecture Notes in Computer Science*, pages 135–153. Springer, 2011.

[103] R. Hennicker, A. Knapp, and H. Baumeister. Semantics of OCL Operation Specifications. *Electr. Notes Theor. Comput. Sci.*, 102:111–132, 2004.

[104] T. A. Henzinger. The theory of hybrid automata. In *11th Annual IEEE Symposium on Logic in Computer Science (LICS 1996)*, pages 278–292. IEEE Computer Society, 1996.

[105] T. A. Henzinger, R. Majumdar, and V. S. Prabhu. Quantifying similarities between timed systems. In P. Pettersson and W. Yi, editors, *Third International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS 2005)*, volume 3829 of *Lecture Notes in Computer Science*, pages 226–241. Springer, 2005.

[106] T. A. Henzinger and J. Sifakis. The embedded systems design challenge. In *14th International Symposium on Formal Methods (FM 2006)*, volume 4085 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2006.

[107] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[108] M. Huth. Refinement is complete for implementations. *Formal Asp. Comput.*, 17(2):113–137, 2005.

[109] H. Hüttel and K. G. Larsen. The use of static constructs in a modal process logic. In A. R. Meyer and M. A. Taitslin, editors, *Logic at Botik '89, Symposium on Logical Foundations of Computer Science*, volume 363 of *Lecture Notes in Computer Science*, pages 163–180. Springer, 1989.

[110] ISO/IEC. *Information Processing Systems – Open Systems Interconnection: LOTOS, A Formal Description Technique Based on the Temporal Ordering of Observational Behavior*, 1989.

[111] P. Jancar and F. Moller. Techniques for decidability and undecidability of bisimilarity. In J. C. M. Baeten and S. Mauw, editors, *10th International Conference on Concurrency Theory (CONCUR 1999)*, volume 1664 of *Lecture Notes in Computer Science*, pages 30–45. Springer, 1999.

[112] S. Janisch. *Behaviour and Refinement of Port-Based Components with Synchronous and Asynchronous Communication*. PhD thesis, Institut für Informatik, Ludwig-Maximilians-Universität München, July 2010.

[113] C. B. Jones. *Development methods for computer programs including a notion of interference*. PhD thesis, Oxford University Computing Laboratory, 1981.

[114] L. Juhl, K. Larsen, and J. Srba. Modal transition systems with weight intervals. *Journal of Logic and Algebraic Programming*, 81(4):408–421, 2012.

[115] P. C. Kanellakis and S. A. Smolka. CCS Expressions, Finite State Processes, and Three Problems of Equivalence. *Inf. Comput.*, 86(1):43–68, 1990.

[116] D. K. Kaynar, N. A. Lynch, R. Segala, and F. W. Vaandrager. Timed I/O Automata: A mathematical framework for modeling and analyzing real-time systems. In *24th IEEE Real-Time Systems Symposium (RTSS 2003)*, pages 166–177. IEEE Computer Society, 2003.

[117] J. Kofron. Extending behavior protocols with data and multisynchronization. Technical Report 2006/10, Dep. of SW Engineering, Charles University in Prague, October 2006.

[118] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[119] A. Kucera and P. Jancar. Equivalence-checking on infinite-state systems: Techniques and results. *Theory and Practice of Logic Programming*, 6(3):227–264, 2006.

[120] K. G. Larsen. *Context-Dependent Bisimulation Between Processes*. PhD thesis, Edinburgh University, 1986.

[121] K. G. Larsen. Modal specifications. In J. Sifakis, editor, *Automatic Verification Methods for Finite State Systems (AVMFSS 1989)*, volume 407 of *Lecture Notes in Computer Science*, pages 232–246. Springer, 1989.

[122] K. G. Larsen, U. Fahrenberg, and C. Thrane. Metrics for weighted transition systems: Axiomatization and complexity. *Theoretical Computer Science*, 412(28):3358–3369, 2011.

[123] K. G. Larsen, U. Nyman, and A. Wasowski. Interface input/output automata. In *14th International Symposium on Formal Methods (FM 2006)*, volume 4085 of *Lecture Notes in Computer Science*, pages 82–97. Springer, 2006.

[124] K. G. Larsen, U. Nyman, and A. Wasowski. Modal I/O Automata for Interface and Product Line Theories. In R. D. Nicola, editor, *16th European Symposium on Programming, Programming Languages and Systems (ESOP 2007)*, volume 4421 of *Lecture Notes in Computer Science*, pages 64–79. Springer, 2007.

[125] K. G. Larsen, U. Nyman, and A. Wasowski. On Modal Refinement and Consistency. In *18th International Conference on Concurrency Theory (CONCUR 2007)*, volume 4703 of *Lecture Notes in Computer Science*, pages 105–119. Springer, 2007.

[126] K. G. Larsen and B. Thomsen. A modal process logic. In *Third Annual Symposium on Logic in Computer Science (LICS 1988)*. IEEE Computer Society, IEEE Computer Society, 1988.

[127] K. G. Larsen and L. Xinxin. Equation solving using modal transition systems. In *Fifth Annual IEEE Symposium on Logic in Computer Science (LICS 1990)*, pages 108–117. IEEE Computer Society, 1990.

[128] E. A. Lee and A. L. Sangiovanni-Vincentelli. Comparing models of computation. In *International Conference on Computer-Aided Design (ICCAD 1996)*, pages 234–241. ACM and IEEE Computer Society, 1996.

[129] E. A. Lee and A. L. Sangiovanni-Vincentelli. Component-based design for the future. In *Design, Automation and Test in Europe (DATE 2011)*, page 1029. IEEE, 2011.

[130] D. J. Lehmann, A. Pnueli, and J. Stavi. Impartiality, justice and fairness: The ethics of concurrent termination. In S. Even and O. Kariv, editors, *8th Colloquium on Automata, Languages and Programming (ICALP 1981)*, volume 115 of *Lecture Notes in Computer Science*, pages 264–277, 1981.

[131] B. Liskov and J. M. Wing. A behavioral notion of subtyping. *ACM Trans. Program. Lang. Syst.*, 16(6):1811–1841, 1994.

[132] Z. Liu, J. He, and X. Li. rCOS: Refinement of Component and Object Systems. In *Third International Symposium on Formal Methods for Components and Objects (FMCO 2004)*, volume 3657 of *Lecture Notes in Computer Science*, pages 183–221. Springer, 2004.

[133] N. Lynch and M. R. Tuttle. An introduction to Input/Output automata. *CWI-quarterly*, 2(3), 1989.

[134] N. A. Lynch, R. Segala, and F. W. Vaandrager. Hybrid I/O automata. *Inf. Comput.*, 185(1):105–157, 2003.

[135] N. A. Lynch and M. R. Tuttle. Hierarchical correctness proofs for distributed algorithms. In *6th Annual Symp. Principles of Distributed Computing (PODC 1987)*, pages 137–151. ACM Press, 1987.

[136] J. Magee, N. Dulay, S. Eisenbach, and J. Kramer. Specifying distributed software architectures. In W. Schäfer and P. Botella, editors, *5th European Software Engineering Conference (ESEC 1995)*, volume 989 of *Lecture Notes in Computer Science*, pages 137–153. Springer, 1995.

[137] J. Magee and J. Kramer. *Concurrency - state models and Java programs (2. ed.)*. Wiley, 2006.

[138] P. Mayer. *MDD4SOA - Model-Driven Development for Service-Oriented Architectures*. PhD thesis, Institut für Informatik, Ludwig-Maximilians-Universität München, October 2010.

[139] P. Mayer, A. Schroeder, and S. Bauer. A Strict-Observational Interface Theory for Analysing Service Orchestrations. *Electronic Notes in Theoretical Computer Science*, 264(1):125–139, 2010.

[140] B. Meyer. Applying "design by contract". *IEEE Computer*, 25(10):40–51, 1992.

[141] R. Milner. *A Calculus of Communicating Systems*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1982.

[142] R. Milner. *Communication and Concurrency*. Prentice Hall (International Series in Computer Science), 1989.

[143] M. L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1967.

[144] MIO Workbench. `http://www.miowb.net/`.

[145] J. Misra and K. M. Chandy. Proofs of networks of processes. *IEEE Trans. Software Eng.*, 7(4):417–426, 1981.

[146] S. Mouelhi, S. Chouali, and H. Mountassir. Refinement of interface automata strengthened by action semantics. *Electr. Notes Theor. Comput. Sci.*, 253(1):111–126, 2009.

[147] J. R. Munkres. *Topology*. Prentice Hall, second edition, 2000.

[148] Object Management Group (OMG). Object Constraint Language (OCL) Specification, Version 2.2. Technical report, OMG, 2010. `http://www.omg.org/spec/OCL/2.2/PDF`, last accessed on August 25, 2012.

[149] Object Management Group (OMG). UML Superstructure Specification 2.4.1. Technical report, OMG, August 2011. `http://www.omg.org/spec/UML/2.4.1/Superstructure/PDF`, last accessed on August 25, 2012.

[150] C. S. Pasareanu, M. B. Dwyer, and M. Huth. Assume-guarantee model checking of software: A comparative case study. In D. Dams, R. Gerth, S. Leue, and M. Massink, editors, *Theoretical and Practical Aspects of SPIN Model Checking, 5th and 6th International SPIN Workshops*, volume 1680 of *Lecture Notes in Computer Science*, pages 168–183. Springer, 1999.

[151] F. Plasil and S. Visnovsky. Behavior protocols for software components. *IEEE Trans. Software Eng.*, 28(11):1056–1076, 2002.

[152] K. Pohl, G. Böckle, and F. van der Linden. *Software Product Line Engineering—Foundations, Principles, and Techniques*. Springer, 2005.

[153] A. Puri. Dynamical properties of timed automata. *Discrete Event Dynamic Systems*, 10(1-2):87–113, 2000.

[154] S. Quinton and S. Graf. Contract-based verification of hierarchical systems of components. In *Sixth IEEE International Conference on Software Engineering and Formal Methods (SEFM 2008)*, pages 377–381. IEEE Computer Society, 2008.

[155] M. O. Rabin. Probabilistic automata. *Information and Control*, 6(3):230–245, 1963.

[156] J.-B. Raclet. Residual for component specifications. *Electr. Notes Theor. Comput. Sci.*, 215:93–110, 2008.

[157] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay, and R. Passerone. Modal interfaces: unifying interface automata and modal specifications. In S. Chakraborty and N. Halbwachs, editors, *9th ACM & IEEE International conference on Embedded software (EMSOFT 2009)*, pages 87–96. ACM, 2009.

[158] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay, and R. Passerone. A modal interface theory for component-based design. *Fundam. Inform.*, 108(1-2):119–149, 2011.

[159] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, and R. Passerone. Why Are Modalities Good for Interface Theories? In *9th International Conference on Application of Concurrency to System Design (ACSD 2009)*, pages 119–127, Los Alamitos, CA, USA, 2009. IEEE Computer Society.

[160] M. Roggenbach. CSP-CASL - A new integration of process algebra and algebraic specification. *Theor. Comput. Sci.*, 354(1):42–71, 2006.

[161] A. Sampaio, J. Woodcock, and A. Cavalcanti. Refinement in Circus. In L.-H. Eriksson and P. A. Lindsay, editors, *International Symposium of Formal Methods Europe (FME 2002)*, volume 2391 of *Lecture Notes in Computer Science*, pages 451–470. Springer, 2002.

[162] D. Sannella and M. Wirsing. A kernel language for algebraic specification and implementation. In M. Karpinski, editor, *Second Workshop on Abstract Data Type (ADT 1983)*, volume 158 of *Lecture Notes in Computer Science*, pages 413–427. Springer, 1983.

[163] S. Shoham and O. Grumberg. Monotonic Abstraction-Refinement for CTL. In K. Jensen and A. Podelski, editors, *10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2004)*, volume 2988 of *Lecture Notes in Computer Science*, pages 546–560. Springer, 2004.

[164] J. Sifakis. Embedded systems design - scientific challenges and work directions. In J. Esparza and R. Majumdar, editors, *16th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2010)*, volume 6015 of *Lecture Notes in Computer Science*, page 1. Springer, 2010.

[165] J. Sifakis. A vision for computer science – the system perspective. *Central European Journal of Computer Science*, 1(1):108–116, 2011.

[166] G. Smith and J. Derrick. Specification, Refinement and Verification of Concurrent Systems-An Integration of Object-Z and CSP. *Formal Methods in System Design*, 18(3):249–284, 2001.

[167] J. M. Spivey. *Z Notation - a reference manual (2. ed.)*. Prentice Hall International Series in Computer Science. Prentice Hall, 1992.

[168] C. Stirling. Local model checking games. In I. Lee and S. A. Smolka, editors, *6th International Conference on Concurrency Theory (CONCUR 1995)*, volume 962 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 1995.

[169] C. Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, 2nd edition, 2002.

[170] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5:285–309, 1955.

[171] The Eclipse Foundation. Xtext 2.1, 2012. `http://www.eclipse.org/Xtext/`.

[172] C. Thrane. *Quantitative Models and Analysis For Reactive Systems*. PhD thesis, Aalborg University, 2011.

[173] C. Thrane, U. Fahrenberg, and K. G. Larsen. Quantitative simulations of weighted transition systems. *Journal of Logic and Algebraic Programming*, 79(7):689–703, 2010.

[174] UPPAAL. `http://www.uppaal.org/`.

[175] R. J. van Glabbeek and W. P. Weijland. Branching time and abstraction in bisimulation semantics. *J. ACM*, 43(3):555–600, 1996.

[176] M. Wirsing. Algebraic specification. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 675–788. Elsevier and MIT Press, 1990.

[177] J. Woodcock and A. Cavalcanti. The semantics of Circus. In D. Bert, J. P. Bowen, M. C. Henson, and K. Robinson, editors, *Second International Conference on Formal Specification and Development in Z and B (ZB 2002)*, volume 2272 of *Lecture Notes in Computer Science*, pages 184–203. Springer, 2002.

[178] M. D. Wulf, L. Doyen, and J.-F. Raskin. Almost asap semantics: from timed models to timed implementations. *Formal Asp. Comput.*, 17(3):319–341, 2005.

[179] D. N. Xu, G. Gößler, and A. Girault. Probabilistic contracts for component-based design. In *8th International Symposium on Automated Technology for Verification and Analysis (ATVA 2010)*, volume 6252 of *Lecture Notes in Computer Science*, pages 325–340. Springer, 2010.

[180] U. Zwick and M. Paterson. The complexity of mean payoff games. In D.-Z. Du and M. Li, editors, *First Annual International Conference on Computing and Combinatorics (COCOON 1995)*, volume 959 of *Lecture Notes in Computer Science*, pages 1–10. Springer-Verlag, 1995.