# State Space Representation for Verification of Open Systems

IREM AKTUG

Licentiate Thesis
Stockholm, Sweden 2006

TRITA-CSC-A2006:3
ISSN 1653-5723
ISRN KTH/CSC/A-06/03–SE
ISBN 91-7178-341-5

KTH CSC
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie licentiatavhandling 31 Maj 2006 10:00 i E3, Kungl Tekniska högskolan, Osquars Backe 14, Stockholm.

Tryck: Universitetsservice US AB

## Abstract

When designing an open system, there might be no implementation available for certain components at verification time. For such systems, verification has to be based on assumptions on the underspecified components. In this thesis, we present a framework for the verification of open systems through explicit state space representation.

We propose Extended Modal Transition Systems (EMTS) as a suitable structure for representing the state space of open systems when assumptions on components are written in the modal $\mu$-calculus. EMTSs are based on the Modal Transition Systems (MTS) of Larsen. This representation supports state space exploration based verification techniques, and provides an alternative formalism for graphical specification. In interactive verification, it enables proof reuse and facilitates visualization for the user guiding the verification process.

We present a two-phase construction from process algebraic open system descriptions to such state space representations. The first phase deals with component assumptions, and is essentially a maximal model construction for the modal $\mu$-calculus that makes use of a powerset construction for the fixed point cases. In the second phase, the models obtained are combined according to the structure of the open system to form the complete state space. The construction is sound and complete for systems with a single unknown component and sound for those without dynamic process creation.

We suggest a tableau-based proof system for establishing open system properties of the state space representation. The proof system is sound and it is complete for modal $\mu$-calculus formulae with only prime subformulae.

A complete framework based on the state space representation is offered for the automatic verification of open systems. The process begins with specifying the open system by a process algebraic term with assumptions. Then, the state space representation is extracted from this description using the construction described above. Finally, open system properties can be checked on this representation using the proof system.

## Acknowledgements

I owe my deepest gratitude to my supervisor, Dilian Gurov. He always managed to find the time and patience for guiding me in the past 2,5 years. Through our long discussions, and his ingenious comments I have learned how to be a scientist. Without his continuous encouragement and friendly support combined with invaluable expert advice, this thesis would have never been finished.

I thank Mads Dam, Marieke Huisman and Christoph Sprenger for helpful comments on this work.

I also want to thank two great scientists from my former university METU, Halit Oguztuzun and Cem Bozsahin, who have supported me beginning from my undergraduate days and gave me the initial excitement of science. I think of you every step of the way as I try to live up to the title "diligent".

Mika Cohen has been my longest lasting office mate, my first friend in Sweden, my first logician acquaintance and many more things. I can not imagine a Ph.D. life without him.

I thank everyone who has put up with me as I produced the work in this thesis and complained incessantly in the meanwhile: Volkan Bilyar, all my friends at IMIT Adam Strak, Steffen Albrecht, Sezi Yamac, my dear home mate Bagsen Aktas, all the friends I have made at summer schools but especially the one, my friends from Turkey Utku Erdogdu, Baris Sertkaya, Sinan Kalkan, Ruken Cakici, my eternal student Bartan, our new group member Wen Xu, and last but most, Anders Johansson.

I am most grateful to my fairies, Serife Tekin and Idil Aktug, as always. They give me inspiration, warmth; help me swim deep and fly high and write fluent.

I am indebted to my beloved friends Gokcen Bas, Elcil Kaya, Zeren Ergonul, Elif Bato and Gonca Barit from Izmir Science High School. Though we may not see each other every year, I rejoice your being at all times. You are a part of me.

Finally, I dedicate this thesis to my parents.. Siz olmasaniz ben ne olurdum, ne ben olurdum.

# Contents

# Chapter 1

# Introduction

Modern software is designed as a collection of components. Modularity brings flexibility to both the development and use of software. For instance, components are developed by different partners and put together at later stages or some component of the system is replaced after some initial phase of use by a new component which performs the same task in a more efficient manner. Certain components can even join the system after it has been put in operation. This is the case, for example, when applications are loaded on a smart card after the card has been issued (see e.g. [35]).

In such scenarios, each intermediate system which "misses" components can be thought of as an *open system*. An open system is a system with "holes" in it standing for the missing components. Each hole is accompanied by some property which is a condition that the component to fill the hole should satisfy. In contrast, a closed system has all its components fixed. An open system captures an infinite set of closed systems, where each holes is filled with some component that satisfies the corresponding property.

*Verification* is the task of showing that software does what it is intended to do, i.e. showing it behaves according to its *specification*. A common way of specifying desired behaviour is through expressing it as a collection of properties in some temporal logic. Verification of an open system amounts to showing that all the closed systems captured by the open system display these properties. This can only be achieved through a symbolic representation of the open system behaviour.

In this thesis, we propose a framework for the verification of open systems through explicit state space representation. In our approach, we represent the behaviour of the open system as a finite structure which is comprised of states, transitions and an acceptance condition which excludes certain non-terminating behavior. The variety in behaviour induced by the assumptions on the not-yet-available components is reflected through *necessary* and *admissible* transitions, which respectively correspond to common and possible behaviour of the closed systems captured by the open system. When the state space of the open system is captured by such a structure, verification of desired properties of the open system can be performed on this finite structure.

The thesis is organized as follows. We first give motivation for our approach accompanied by a detailed account of our framework and related work. In Chapter 3, we introduce the syntax of open terms with assumptions (OTA), the notion we use to specify open systems. The structure we use to represent the state space of open systems, Extended Modal Transition Systems (EMTS), is presented in Chapter 4 along with a simulation relation which defines the set of closed systems denoted by an EMTS. Chapter 5 aims to illustrate the procedure we have introduced for automatic conversion of OTA to EMTS through examples. In Chapter 6, a proof system is presented for verifying properties of EMTS states expressed in modal $\mu$-calculus. Chapter 7 mentions work that is not directly used in this study but is nevertheless related to our approach. Section 8.1 summarizes the current study and its contributions. Finally, Section 8.2 concludes the thesis with an outline of future work.

The work presented here resulted in two papers:

1. Aktug and D.Gurov, "Towards State Space Exploration Based Verification of Open Systems" to appear in Proceedings of the $4^{th}$ International Workshop on Automated Verification of Infinite-State Systems (AVIS'05), April 2005, Edinburgh, Scotland

2. Aktug and D. Gurov, "State Space Representation for Verification of Open Systems", to appear in Proceedings of the $11^{th}$ International Conference on Algebraic Methodology and Software Technology, (AMAST '06), July 2006, Kuressaare, Estonia

The main text of this thesis is designed as an introductory text that is complementary to these papers, which can be found in Appendix A and B. The main text provides an overview of our work and intends to clarify certain points, e.g. through the use of examples, that were left out in the papers due to lack of space. Paper 1 includes the introduction of the EMTS notion and the proof system that we use to show properties of EMTSs. It also contains the soundness and completeness proofs of this proof system. Paper 2, on the other hand, concentrates on the construction of EMTSs from OTAs while presenting an updated version of the definition of an EMTS and the adaptation of the proof system to this new version. The proofs of the theorems can be found in Appendix C.

## 1.1 Motivation

Modal Transition Systems is an intuitive notion that was designed for graphical specification of system behaviour [29]. Each MTS specifies a set of processes through an interval determined by necessary and admissable transitions. MTSs are equiexpressive with Hennessy-Milner logic, i.e. an HML formula can be characterized as an MTS and vice versa. MTSs provide a natural representation of open systems when assumptions on the behavior of the not-yet-available components are specified in HML.

Such an explicit state space representation supports various phases of the development of open systems:

- In *the modeling phase*, this formalism can be used as an alternative means of graphical specification. Certain kinds of properties are easier to express graphically than in temporal logics.
- In *automatic verification*, it provides a visualization of the system behaviour. This is mostly beneficial if the automatic proof construction fails and an understanding of the open system behaviour becomes necessary for debugging. Furthermore, computing the whole state space enables proof reuse when the same system is to be checked for several properties.
- In *interactive verification*, such a state space representation is all the more vital. While it is possible to use conventional methods like encoding system behaviour into alternating automata [28] for automatic cases, the human factor in interactive verification requires a more intuitive representation.

In a process algebraic setting, the behaviour of an open system can be specified by an *open process term with assumptions* (OTA). An OTA has the shape $\Gamma \rhd E$ and consists of a process term $E$ equipped with a list of behavioural assumptions $\Gamma$ of the shape $X : \Phi$, where $X$ is a process variable free in $E$ and $\Phi$ is a temporal property. Such an open term denotes a set of closed systems, namely those that can be obtained by substituting each free process variable in E with a closed component satisfying the respective assumptions specified in $\Gamma$. A property of an OTA is then a property shared by all the closed systems in its denotation.

MTS are not expressive enough for representing the state space of open systems when assumptions are temporal properties. We extend MTSs so that we can represent the state space of open systems when the component assumptions are written in modal $\mu$-calculus, which adds the expressive power of least and greatest fixed point recursion to HML. Besides the must and may transitions of MTS, our notion, *Extended Modal Transition System* (EMTS) has sets of states (instead of single states) as targets to transitions - an extension which is needed for dealing with disjunctive assumptions. In addition, we add well-foundedness constraints to the structure to handle least fixed point assumptions.

In this thesis, we offer an automatic method for open system verification through explicit state space representation in the form of an EMTS. The process begins by specifying the system as an OTA. Then a two-phase construction, under given restrictions, automatically extracts an EMTS from an OTA. The first phase in the construction corresponds to a maximal model construction for each component assumption. In the second phase, the maximal models are composed according to the structure of the OTA term. The construction is *sound* (resp. *complete*) if the denotation of the OTA is a subset (resp. superset) of the denotation of the resulting

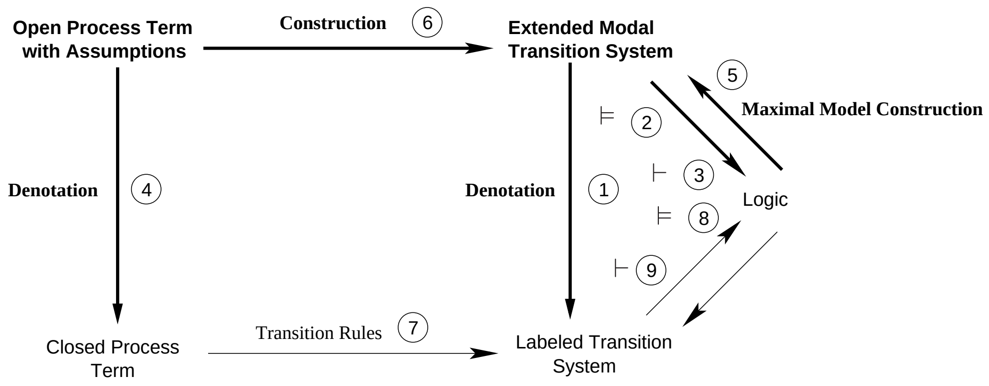

Open Process Term with Assumptions
Construction
6
Extended Modal Transition System
5
Maximal Model Construction
⊨ 2
⊢ 3
⊨ 8
⊢ 9
Logic
Denotation
4
Denotation
1
Transition Rules
7
Closed Process Term
Labeled Transition System

Figure 1.1: Overview of Notions

EMTS. We show soundness of the construction for systems without dynamic process creation, and soundness and completeness for systems without parallel composition. Finally, we give a proof system to prove properties of EMTSs.

The proof system based method of Dam and Gurov [15] is an example of interactive verification of open systems. Reasoning about open systems in such a proof-theoretic manner can essentially be viewed as a symbolic execution of OTA. As the state space is explored guided by the formula to be shown, a symbolic state-transition graph can be generated which is conveniently captured as an EMTS. This graph can be used to visualize the behaviour of the system that (otherwise) remains implicit in the proof tree, thus providing an understanding of the behaviour of the system that aids the current interaction as well as future verification efforts. It also serves proof reuse as mentioned above. We leave possible interactive approaches based on EMTSs to future work.

## 1.2 Overview of Notions and Results

Figure 1.1 shows an overview of the central notions used in the thesis and the relations between them.*OTA* and *EMTS* are the new notions that are proposed by the thesis (See Chapters 3 and 4 respectively). Whereas, the concepts of *closed process term* and *labeled transition system* are already well-developed.

For modeling open systems, we propose open process terms with assumptions on the free variables (OTA). Such an open term *denotes* an infinite set of closed terms, namely all those which can be obtained from the open term by substituting the free variables with closed terms satisfying the respective assumptions. The analogy between closed and open terms through the relationship of closed system to open system can be extended to one between labeled transition systems (LTS) and extended modal transition systems (EMTS). The *denotation* of a state of an

EMTS is the set of labeled transition system states that this state relates to by some *simulation* relation.

The particular *logic* we use in this study is the modal $\mu$-calculus. (see Chapter 3 for a short introduction) The assumptions in an OTA and the properties to be checked for the open system are both expressed in this temporal logic. *Satisfaction* defines when an EMTS state is said to satisfy a temporal logic formula. We use the proof system by Bradfield and Stirling for checking if states of an LTS satisfy a temporal property expressed in modal $\mu$-calculus. We present a *proof system* to check satisfaction of a modal $\mu$-calculus property by a state of an EMTS. (A summary of both proof systems along with an account of major differences can be found in Chapter 6) The soundness and completeness properties shown for prime formulae make our proof system adequate for proving satisfaction for prime properties (Items 2 and 3 in Figure 1.1).

Given a temporal logic formula, the EMTS that characterizes it can be constructed using the maximal model construction presented in Chapter 5. The labeled transition system corresponding to a closed term can be constructed using transition rules. Similarly, *construction* of the state space of an OTA in the form of an EMTS can be done in different ways. Here we present an automatic construction in which the maximal models for assumptions of an OTA are combined according to the structure of the process term. (See Paper 2 and Sections 5.1 and 5.2 for details of the construction and examples)

If the various transformations are correctly defined, the diagram should commute. In particular, given a labeled transition system, the construction of an EMTS from an OTA should preserve the denotation (Items 6 and 1 vs. Items 4 and 7 of Figure 1.1). This is the case for the automatic construction we introduce in this thesis when the open system does contains a single unknown component. (See Section 5.3) Similarly, the correctness of the defined proof system (Item 3) combined with soundness and completeness properties would provide a proof for the satisfaction of a property by a state of the EMTS if and only if for each LTS, the set of all states that are denoted (Item 1) by this state satisfy the property (Item 8).

# Chapter 2

# Background and Related Work

In this chapter, we give a brief account of various methods of verification that were inspirational to our work: *compositional reasoning* and structures capturing properties, namely *model transition systems* and *automata*. Previous research in compositional reasoning has guided us in determining the state space of an open system, while MTSs and automata inspired us in designing a structure to represent it.

In Chapter 7, we summarize approaches that are related to ours but not directly inspirational like *abstraction* and *partial model checking*.

## 2.1 Compositional Reasoning and Maximal Models

Compositional reasoning aims to avoid state space explosion by taking advantage of the natural decomposition of the system in components. The goal is to verify properties of individual components, and infer a property of the system which is formed by a composition of these components and thus avoid to compute the state space of the whole system.

The earliest formalization of this intuition is Pnueli's *assume-guarantee paradigm* [34]. The compositional reasoning extension is the following additional rule to the logic:

$$\frac{\begin{array}{c}\langle\Phi\rangle\, P\, \langle\Psi\rangle \\ \langle true\rangle\, P'\, \langle\Phi\rangle\end{array}}{\langle true\rangle\, P' \mid P\, \langle\Psi\rangle}$$

The first premise expresses that assuming the environment satisfies $\Phi$, component $P$ guarantees the satisfaction of the temporal property $\Psi$. The second premise simply expresses that the rest the rest of the system, $P'$, satisfies $\Phi$. From these it is concluded that their composition $P' \mid P$ satisfies $\Psi$. The rule, then, brings together a proof about component $P$ with one about the rest of the system $P'$ to reach a conclusion about their composition.

The decomposition of the required property $\Psi$ into an adequate assumption $\Phi$ for component $P$ requires knowledge of the system and remains largely to be a task of the user. In order to automate the rest of the tasks, Grumberg and Long suggested a preorder on the finite state models that preserves satisfaction of temporal logic formulae [19]. The finite models in this study are synchronous parallel compositions of Kripke structures under fairness assumptions.

**Definition 2.1** (Structure (Exists) Simulation)**.** Let $P$ and $P'$ be two structures and let $s$ and $s'$ be states in $S_P$ and $S_{P'}$, respectively. A relation $H \subseteq S_P \times S_{P'}$ is a simulation relation from $(P, s)$ to $(P', s')$ iff the following conditions hold:

1. $H(s, s')$.
2. For all $t$ and $t'$, $H(t, t')$ implies:
   (a) $t'$ satisfies all atomic propositions satisfied by $t$
   (b) for every fair path $n = t_0t_1t_2..$ in $P$ there exists a fair path $n' = t'_0t'_1t'_2..$ in $P'$ such that for every $i \geq 0$, $H(t_i, t'_i)$.

$H$ is a simulation from $P$ to $P'$ if and only if for every initial state $s_0 \in S_P$ there is an initial state $s'_0 \in S_{P'}$ such that $H(s_0, s'_0)$. If there is such a simulation relation from $P$ to $P'$, then we say $P'$ *simulates* $P$, denoted $P \preceq P'$.

This simulation relation has two important features. The first is the preservation of temporal formula. So if $P \preceq P'$, then for every $\forall CTL$ formula $\Phi$, $P' \vDash \Phi$ implies $P \vDash \Phi$. The second is that $P$ simulates every system that consists of its composition with some component $P'$, i.e. for any $P$ and $P'$, $P \parallel P' \preceq P$.

The automatization of maximal model construction is one of the keys to the applicability of compositional verification. A tableau construction for ∀CTL formulae was described in Grumberg and Long [19]. The method was later extended to ∀CTL* by Kupferman and Vardi [26]. The maximal model $\mathcal{M}_\Psi$ for a formula $\Psi$ can be thought of as the most generic model to satisfy the formula, so that its behaviors are shared by all other models that satisfy $\Psi$.

$$P \models \Psi \iff P \preceq \mathcal{M}_\Psi$$

Through the use of maximal models, checking $P \parallel P' \vDash \Psi$ is reduced to the following steps: 1. Decompose $\Phi$ to the local property $\Psi$ 2. Construct the maximal model for $\Psi$ $\mathcal{M}_\Psi$ 3. Check by standard model checking algorithms that $P'$ satisfies $\Psi$ 4. Check by standard model checking algorithms that $\mathcal{M}_\Psi \parallel P$ satisfies $\Phi$:

$$\frac{P' \models \Psi \qquad \mathcal{M}_\Psi \parallel P \models \Phi}{P' \parallel P \models \Phi}$$

Maximal model construction is explored in [35] for reasoning about sequential applets which have potentially infinite behavior. The process is proposed for a logic equivalent to modal $\mu$-calculus without diamond modalities and least fixed points. In this study, first the simulation and a corresponding logic, called simulation logic, is introduced. Then, two characterization results are presented. The first is the behavioral characterization of logical satisfaction, which corresponds to our maximal model definition.

The second is the complementary result of logical characterization of simulation that says there exists a characteristic formula $\chi(S)$ for each specification $S$ with respect to the simulation relation $\preceq$:

$$P \preceq S \quad \Longleftrightarrow \quad P \vDash \chi(S)$$

Maximal model construction is applied to all formulae in the simulation logic by transforming them stepwise to simulation normal form for which the mapping is defined directly. These two characterizations form a Galois connection with respect to the preorder of logical formulae ordered by logical consequence and the preorder of specifications ordered by simulation.

## Compositional Reasoning for Open Systems

In our understanding of the term, verification of open systems can be performed by a compositional proof system due to Dam et al [16] for CCS processes and [14] for Erlang programs. It is a Gentzen-style compositional proof system.

In this proof system system, the sequents are of the form $\Gamma \vdash \Delta$ where $\Gamma$ and $\Delta$ are comprised of correctness assertions. These assertions may require a process to satisfy a temporal formula $E : \phi$, require a process to do a certain transition $E \xrightarrow{\alpha} F$ or force a relation between ordinal variables $\kappa < \kappa'$. The ordinal variables are used to relate the rates of progress for fixed point formulae appearing in different places of a sequent.

In this system, compositional reasoning is accomplished through a general rule of subterm cut:

$$\frac{\Gamma \vdash Q : \psi, \Delta \qquad \Gamma, x : \psi \vdash P : \phi, \Delta}{\Gamma \vdash P[Q/x] : \phi, \Delta}$$

The proof progresses guided by the temporal logic formula to be verified and a global discharge condition is employed which recognizes proofs by well-founded induction.

It is possible to formulate open system verification problem in this framework by placing assumptions on components in $\Gamma$ while the structure of the system can be asserted as a process algebra term in $\Delta$. This proof system is more powerful than our current framework: for instance it is possible to verify systems with dynamically changing configuration due to dynamic process spawning. Nevertheless, we feel that our approach also has its advantages. In the above proof system, (the explored part of) the state space is only implicitly present in a proof. Building an explicit representation of the state space allows proof reuse utilizing the (part of) the behavior already explored during proof search. When the verification task is undecidable (as in the present case, unless the temporal logic is appropriately restricted), one has to rely on interactive methods, and then visualizing the state space can be a significant aid in guiding the proof. Finally, separating the task of building the state space from the task of checking its properties (even if in a synchronized fashion as in local model checking) allows user interaction to focus on the first, potentially undecidable task, and thus be freed from the second task which is decidable for any finite representation of the state space.

## 2.2 Structures Capturing Properties

Attempts to characterize formula with finite structures resulted from different concerns. Modal Transition System (MTS) is a graphical specification language in the process algebra framework. MTS was designed as a more intuitive alternative to Hennessy-Milner logic. Whereas, automata have been used more for verification purposes, for instance maximal models used in compositional reasoning have been constructed in the form of automata.

We have been inspired by both MTSs and automata when coming up with a notion that is suitable for representing the state space of open systems where assumptions on components are expressed in the modal $\mu$-calculus. Our structure, EMTS, is based on modal transition systems of Larsen with an acceptance condition borrowed from automata in order to encode prohibited infinite runs of the system.

### Modal Transition Systems

MTSs were designed as a graphical specification language in the process algebra framework by Larsen [29]. Each MTS specifies a set of processes through an interval determined by necessary and admissable transitions. MTSs are equiexpressive with Hennesy-Milner Logic [22].

**Definition 2.2** (MTS)**.** A modal transition system is a structure $\mathcal{S} = (S, A, \longrightarrow_\Box, \longrightarrow_\Diamond)$ where $\mathcal{S}$ is a set of specifications, $A$ is a set of actions and $\longrightarrow_\Box, \longrightarrow_\Diamond \subseteq S \times A \times S$, satisfying the consistency condition $\longrightarrow_\Box \subseteq \longrightarrow_\Diamond$.

An MTS can be refined stepwise to an *implementation* that performs all the *must* transitions ($\longrightarrow_\Box$) of the MTS but performs only a subset of the *may* transitions ($\longrightarrow_\Diamond$). The stepwise refinement indicates a preorder between MTSs so that as the specification gets refined the set of processes that implement it gets smaller.

**Definition 2.3** (Refinement)**.** A refinement R is a binary relation on $S$ such that whenever $SRT$ and $a \in A$ then the following holds:

1. Whenever $S \overset{a}{\longrightarrow}_\Diamond S'$, then $T \overset{a}{\longrightarrow}_\Diamond T'$ for some $T'$ with $S'RT'$
2. Whenever $T \overset{a}{\longrightarrow}_\Box T'$, then $S \overset{a}{\longrightarrow}_\Box S'$ for some $S'$ with $S'RT'$

$S$ is said to be a refinement of $T$ in case $(S, T)$ is contained in some refinement $R$, which is denoted $S \lhd T$.

A process $p$ implements a structure $\mathcal{S}$ if there is a refinement relation which contains $(p, \mathcal{S})$, that is if $p \lhd \mathcal{S}$. Processes are MTSs where the must and may transitions coincide, $\longrightarrow_{\Diamond} = \longrightarrow_{\Box}$, since all admissable transitions are also required for a process.

MTSs can also be combined using process constructs of process algebra. This enables a component to be replaced by its refinement. If $S$ and $T$ are MTSs, then transitions of $S|T$ and $S+T$ are defined as below:

$$\begin{array}{rcl} S \mid T \xrightarrow{a}_m V & \Longleftrightarrow & (V = S' \mid T \wedge S \xrightarrow{a}_m S') \vee (V = S' \mid T \wedge S \xrightarrow{a}_m S') \\ S + T \xrightarrow{a}_m V & \Longleftrightarrow & (S \xrightarrow{a}_m V) \vee (T \xrightarrow{a}_m V) \end{array}$$

where $m$ ranges over $\Box$ and $\Diamond$.

It is shown that for each MTS, a characteristic formula exists in Hennessy-Milner Logic so that $S$ is a refinement of $T$ if and only if it satisfies $T$'s characteristic formula, and viewed as specifications both $T$ and its characteristic formula are implemented by the same set of processes.

In [6], a concept similar to maximal models is introduced. The class of formulae of the logic for which a maximal model in the form of an MTS can be constructed is the class of *graphically representable* formulae. A logical formula is graphically representable (i.e. by a single MTS) if and only if it is consistent and *prime*. A formula is prime if and only if it implies one of its disjuncts. The rest of formulae is representable by finitely many MTSs.

Finally, these results show that a Galois connection between the logical consequence preorder on consistent prime formulae and the refinement preorder on MTSs has been established.

## Automata Theoretic Approaches

The establishment of the clean connection between Büchi automata and linear temporal logic (LTL) enabled verification-related problems such as satisfiability and model-checking to be reduced to standard automata-theoretic problems. The observation is to associate with each linear temporal logic formula a finite automaton over infinite words that accepts exactly the computations that satisfy the formula. As a result of this correspondence, already known optimal algorithms from automata theory could be imported to verification.

Similar efforts for branching time logics resulted in the emergence of many different structures to capture temporal formula, e.g. alternating tree automata [33] and amorphous automata [5]. These structures run on infinite trees instead of infinite words and are akin to tree automata. A number of different acceptance conditions also emerged, of which the most frequently used are Muller, Rabin, Streett, and parity conditions. When considered for tree automata, Muller, Rabin and Streett acceptance conditions are equivalent in power. (For a comprehensive survey of automata on infinite trees see [37].)

Emerson and Jutla have shown that modal $\mu$-calculus formula and nondeterministic automata on trees is equiexpressive [17]. They show that the parse tree of a formula of modal $\mu$-calculus can be seen as an alternating tree automaton with, for instance, Streett acceptance condition and then they convert this alternating tree automaton to an equivalent nondeterministic tree automaton. This second step is in general not possible since alternating tree automata are a generalization of nondeterministic tree automata, but alternating tree automata obtained from modal $\mu$-calculus formulae have a special property of being "history-free" which makes the conversion possible. In our maximal model construction for modal $\mu$-calculus we were inspired by the construction Kaivola offered for converting the formula from the alternation-depth class $\Pi_2$ fragment of $\mu$-calculus to Büchi Automata [24].

The reason we introduce yet another formalism to capture modal $\mu$-calculus formulae is that we are interested in representing the state space of any component that satisfies this property in a common structure. Although expressively powerful, we think that the aforementioned structures do not provide an intuitive representation of the state space in terms of states and transitions. The combination of complicated transition relations with acceptance conditions, (consider for instance alternating automata with Streett acceptance [26]), make automata an unattractive choice for graphical specification. In our structure we bring together may and must transitions of MTSs with the parity acceptance condition. The choice of parity acceptance for capturing alternation of fixed points in modal $\mu$-calculus formulae is natural as was noted by Emerson and Jutla [17].

# Chapter 3

# Specifying Open Systems

A system, the behaviour of which is parameterized on the behaviour of certain components, is conveniently represented as a pair $\Gamma \rhd E$, where $E$ is an open process-algebraic term, and $\Gamma$ is a list of assertions of the shape $X : \Phi$ where $X$ is a process variable free in $E$ and $\Phi$ is a closed formula in a process logic.

In the present study, we work with the class of Basic Parallel Processes (BPP)[9]. The terms of BPP are generated by:

$$E ::= \mathbf{0} \mid X \mid a.E \mid E + E \mid E \parallel E \mid \mathit{fix}\ X.E$$

where $X$ ranges over a set of process variables $\mathit{ProcVar}$ and $a$ over a finite set of actions $A$. We assume that $\mathit{ProcVar}$ is partitioned into assumption process variables $\mathit{AssProcVar}$ used in assertions, and recursion process variables $\mathit{RecProcVar}$ bound by the *fix* operator. A term $E$ is called *linear* if every assumption process variable occurs in $E$ at most once. The operational semantics of closed process terms (called processes and ranged over by $t$) is standard. In the rest of this text, the symbol "$\parallel$" signifies *merge composition*, while the symbol "$|$" is used as a symbol for parallel composition in general.

$$\frac{\cdot}{a.E \xrightarrow{a} E} \qquad \frac{E_1 \xrightarrow{a} E_1'}{E_1 + E_2 \xrightarrow{a} E_1'} \qquad \frac{E_2 \xrightarrow{a} E_2'}{E_1 + E_2 \xrightarrow{a} E_2'}$$

$$\frac{E_1 \xrightarrow{a} E_1'}{E_1 \parallel E_2 \xrightarrow{a} E_1' \parallel E_2} \qquad \frac{E_2 \xrightarrow{a} E_2'}{E_1 \parallel E_2 \xrightarrow{a} E_1 \parallel E_2'} \qquad \frac{E_1[\mathit{fix}\ X.E_1/X] \xrightarrow{a} E_1'}{\mathit{fix}\ X.E_1 \xrightarrow{a} E_1'}$$

As a process logic for specifying behavioural assumptions of components, as well as for specifying system properties to be verified, we consider the modal $\mu$-calculus [25]. We have selected to work with it because it subsumes most other well-known logics like CTL and LTL. The formulae of modal $\mu$-calculus are generated by:

$$\Phi ::= \mathrm{tt} \mid \mathrm{ff} \mid Z \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\,\Phi \mid \langle a \rangle\,\Phi \mid \nu Z.\Phi \mid \mu Z.\Phi$$

where $Z$ ranges over a set of propositional variables *PropVar*.

Variable X in $\sigma X.\Phi$, where $\sigma \in \{\nu, \mu\}$ is called *guarded* if every occurrence of $X$ in $\Phi$ is in the scope of some modality operator $\langle a \rangle$ or $[a]$. We say that a formula is guarded if every bound variable in the formula is guarded. A formula $\Phi$ is a *normal* formula if $\sigma_1 Z_1$ and $\sigma_2 Z_2$ are two different occurrences of binders in $\Phi$ then $Z_1 \neq Z_2$ and no occurrence of a free variable $Z$ is also used in a binder $\sigma Z$ in $\Phi$. Let $\Phi$ be a normal formula and $\sigma_1 X.\Psi_1$ and $\sigma_2 Z.\Psi_2$ be subformulae of $\Phi$, then $X$ *subsumes* $Z$ if $\sigma_2 Z.\Psi_2$ is a subformula of $\sigma_1 X.\Psi_1$.

**Definition 3.1** ($\mu$-calculus: semantics)**.** The semantics of the $\mu$-calculus is given in terms of the denotation $||\Phi||^{\mathcal{T}}_{\mathrm{V}} \subseteq S_{\mathcal{T}}$ where $\mathrm{V} : PropVar \to S_{\mathcal{T}}$ is a valuation that maps propositional variables to processes of some *labeled transition system* (LTS) $\mathcal{T} = (S_{\mathcal{T}}, A, \longrightarrow_{\mathcal{T}})$ as follows:

$$\begin{array}{l}
||\mathrm{tt}||^{\mathcal{T}}_{\mathrm{V}} = S_{\mathcal{T}} \\
||\mathrm{ff}||^{\mathcal{T}}_{\mathrm{V}} = \emptyset \\
||Z||^{\mathcal{T}}_{\mathrm{V}} = \mathrm{V}(Z) \\
||\Phi_1 \vee \Phi_2||^{\mathcal{T}}_{\mathrm{V}} = ||\Phi_1||^{\mathcal{T}}_{\mathrm{V}} \cup ||\Phi_2||^{\mathcal{T}}_{\mathrm{V}} \\
||\Phi_1 \wedge \Phi_2||^{\mathcal{T}}_{\mathrm{V}} = ||\Phi_1||^{\mathcal{T}}_{\mathrm{V}} \cap ||\Phi_2||^{\mathcal{T}}_{\mathrm{V}} \\
||\langle a \rangle\, \Phi||^{\mathcal{T}}_{\mathrm{V}} = \{t \mid \exists t'.\, t \xrightarrow{a}_{\mathcal{T}} t' \wedge t' \in ||\Phi||^{\mathcal{T}}_{\mathrm{V}}\} \\
||\,[a]\, \Phi||^{\mathcal{T}}_{\mathrm{V}} = \{t \mid \forall t'.\, t \xrightarrow{a}_{\mathcal{T}} t' \wedge t' \in ||\Phi||^{\mathcal{T}}_{\mathrm{V}}\} \\
||\mu Z.\Phi||^{\mathcal{T}}_{\mathrm{V}} = \bigcap\{T \subseteq S_{\mathcal{T}} \mid T \supseteq ||\Phi||^{\mathcal{T}}_{\mathrm{V}[T/Z]}\} \\
||\nu Z.\Phi||^{\mathcal{T}}_{\mathrm{V}} = \bigcup\{T \subseteq S_{\mathcal{T}} \mid T \subseteq ||\Phi||^{\mathcal{T}}_{\mathrm{V}[T/Z]}\}
\end{array}$$

An alternative, but equivalent, interpretation of extremal fixed points is through approximants. We provide a characterization where *Ord* is the set of ordinals, $\alpha, \kappa \in Ord$ are ordinals, and $\lambda \in Ord$ is a limit ordinal. Let $(\sigma Z.\Phi)^{\alpha}$ be the $\alpha$-unfolding of $\sigma Z.\Phi$ with the following interpretation:

$$\begin{array}{ll}
||(\nu Z.\Phi)^0||^{\mathcal{T}}_{\mathrm{V}} = S_{\mathcal{T}} & ||(\mu Z.\Phi)^0||^{\mathcal{T}}_{\mathrm{V}} = \emptyset \\
||(\nu Z.\Phi)^{\alpha+1}||^{\mathcal{T}}_{\mathrm{V}} = ||\Phi||^{\mathcal{T}}_{\mathrm{V}[||(\nu Z.\Phi)^{\alpha}||^{\mathcal{T}}_{\mathrm{V}}/Z]} & ||(\mu Z.\Phi)^{\alpha+1}||^{\mathcal{T}}_{\mathrm{V}} = ||\Phi||^{\mathcal{T}}_{\mathrm{V}[||(\mu Z.\Phi)^{\alpha}||^{\mathcal{T}}_{\mathrm{V}}/Z]} \\
(\nu Z.\Phi)^{\lambda} = \bigcap\{||(\nu Z.\Phi)^{\alpha}||^{\mathcal{T}}_{\mathrm{V}} \mid \alpha < \lambda\} & (\mu Z.\Phi)^{\lambda} = \bigcup\{||(\mu Z.\Phi)^{\alpha}||^{\mathcal{T}}_{\mathrm{V}} \mid \alpha < \lambda\}
\end{array}$$

Approximants are used in connection to Theorems 3.3, 3.4 and 3.2 in the proof of the maximal model construction that can be found in appendix A of paper 2.

**Theorem 3.2** (Unfolding Theorem)**.** $||\sigma Z.\Phi||^{\mathcal{T}}_{\mathrm{V}} = ||\Phi[\sigma Z.\Phi/Z]||^{\mathcal{T}}_{\mathrm{V}}$*, where* $\sigma$ *is either* $\mu$ *or* $\nu$*.*

**Theorem 3.3** (Knaster-Tarski Theorem)**.** $||(\mu Z.\Phi)||^{\mathcal{T}}_{\mathrm{V}} = \bigcup\limits_{\alpha} ||(\mu Z.\Phi)^{\alpha}||^{\mathcal{T}}_{\mathrm{V}}$

**Theorem 3.4.** $||(\mu Z.\Phi)^{\kappa}||^{\mathcal{T}}_{\mathrm{V}} = \bigcup\limits_{\alpha<\kappa} ||\Phi||^{\mathcal{T}}_{\mathrm{V}[||(\mu Z.\Phi)^{\alpha}||^{\mathcal{T}}_{\mathrm{V}}/Z]}$

As usual, we write $t \models^{\mathcal{T}}_{\mathrm{V}} \Phi$ whenever $t \in ||\Phi||^{\mathcal{T}}_{\mathrm{V}}$. In the sequel, we omit the subscript V from $||\Phi||^{\mathcal{T}}_{\mathrm{V}}$ when $\Phi$ is a closed formula. Satisfaction is lifted to sets of states in the natural way, so that a set of states $S \subseteq S_{\mathcal{T}}$ satisfies a property $\Phi$, $S \vDash_{\mathrm{V}} \Phi$, only if for all $s \in S, s \vDash_{\mathrm{V}} \Phi$.

We say that an OTA $\Gamma \rhd E$ is *guarded* when the term $E$ and all modal $\mu$-calculus formula $\Phi$ in $\Gamma$ are guarded. Similarly, we say an OTA is linear when the term it contains is linear.

The behaviours specified by an open term with assumptions is given with respect to a labeled transition system $\mathcal{T}$ that is closed under the transition rules and is closed under substitution of processes for assumption process variables in subterms of the OTA. The states of LTS correspond to processes in our process algebra. The denotation of an OTA is then the set of all processes obtained by substituting each assumption process variable in the term by a process from $\mathcal{T}$ satisfying the respective assumptions.

**Definition 3.5** (OTA Denotation)**.** Let $\Gamma \rhd E$ be an OTA, $\mathcal{T}$ be an LTS, and $\rho_R$ : RecProcVar $\to S_{\mathcal{T}}$ be a recursion environment. The *denotation* of $\Gamma \rhd E$ relative to $\mathcal{T}$ and $\rho_R$ is defined as:

$$[\![\Gamma \rhd E]\!]^{\mathcal{T}}_{\rho_R} \triangleq \{E\rho_R\rho_A \mid \forall (X : \Phi) \in \Gamma.\ \rho_A(X) \models^{\mathcal{T}} \Phi\}$$

where $\rho_A$ : AssProcVar $\to S_{\mathcal{T}}$ ranges over assumption environments.

**Example.** Consider an operating system in the form of a concurrent server that spawns off *Handler* processes each time it receives a request. These processes run system calls for handling the given requests to produce a result (modeled by the action $\overline{out}$). *Handler* is defined as $Handler \overset{def}{=} In \parallel \overline{out}.\mathbf{0}$ where $In \overset{def}{=} in.In$. Although it is possible to communicate with request handlers through the attached channel (modeled by the action *in*), they do not react to further input. A property one would like to prove of such a server is that it stabilizes whenever it stops receiving new requests. Eventual stabilization can be formalized in the modal $\mu$-calculus as $\mathbf{stab} \triangleq \nu X.\mu Y.\,[in]\,X \wedge [\overline{out}]\,Y$. We can reduce this verification task to proving that the open system modeled by the OTA

$$X : \mathbf{stab} \rhd X \parallel Handler$$

which consists of $Handler$ and any stabilizing process $X$, eventually stabilizes.

# Chapter 4

# Extended Modal Transition Systems

We propose Extended Modal Transition Systems (EMTS) as an explicit state space representation for open systems with temporal assumptions. In this chapter, we summarize the main definitions.

The notion of EMTS is based on Larsen's Modal Transition Systems [29]. In addition to may and must transitions for dealing with modalities, EMTSs include sets of states (instead of single states) as targets to transitions to capture disjunctive assumptions, and a set of prohibited infinite runs defined through a coloring function to represent termination assumptions.

**Definition 4.1** (EMTS)**.** An *extended modal transition system* is a structure

$$\mathcal{E} = (S_\mathcal{E}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, c)$$

where (i) $S_\mathcal{E}$ is a set of *abstract states*, (ii) $A$ is a set of *actions*, (iii) $\longrightarrow^{\Diamond}_{\mathcal{E}}$, $\longrightarrow^{\Box}_{\mathcal{E}}$ $\subseteq S_\mathcal{E} \times A \times 2^{S_\mathcal{E}}$ are *may* and *must transition relations*, and (iv) $c : S_\mathcal{E} \rightarrow \mathbb{N}^k$ is a *coloring function* for some $k \in \mathbb{N}$.

May transitions of an EMTS show possible behaviours of the closed systems represented, while must transitions specify behaviour shared by all these closed systems. A *run* (or may–run) of $\mathcal{E}$ is a possibly infinite sequence of transitions $\rho_\mathcal{E} = s_0 \xrightarrow{a_0}_\mathcal{E} s_1 \xrightarrow{a_1}_\mathcal{E} s_2 \xrightarrow{a_2}_\mathcal{E} \ldots$ where for every $i \geq 0$, $s_i \overset{a_i}{\longrightarrow}^{\Diamond}_{\mathcal{E}} S$ for some $S$ such that $s_{i+1} \in S$. Must–runs are defined similarly. We distinguish between two kinds of *a-derivatives* of a state $s$: $\partial^{\Diamond}_a(s) \triangleq \{S \mid s \overset{a}{\longrightarrow}^{\Diamond}_{\mathcal{E}} S\}$ and $\partial^{\Box}_a(s) \triangleq \{S \mid s \overset{a}{\longrightarrow}^{\Box}_{\mathcal{E}} S\}$.

The coloring function $c$ specifies a set $W_\mathcal{E}$ of prohibited infinite runs by means of a *parity acceptance condition*(cf. [32, 17]). The function $c$ is extended to infinite runs so that $c(\rho_\mathcal{E}) = (c(s_0)(1) \cdot c(s_1)(1) \ldots, \ldots, c(s_0)(k) \cdot c(s_1)(k) \ldots)$ is a $k$-tuple of infinite words where $c(s)(j)$ denotes the $j^{th}$ component of $c(s)$. Let $\mathit{inf}(c(\rho_\mathcal{E})(i))$ denote the set of infinitely occurring colors in the $i^{th}$ word of this tuple. Then the run $\rho_{\mathcal{E}}$ is prohibited, $\rho_{\mathcal{E}} \in W_{\mathcal{E}}$, if and only if $\mathit{max}\,(\mathit{inf}\,(c(\rho_{\mathcal{E}})(i)))$ is odd for some $1 \leq i \leq k$, i.e. the greatest number that occurs infinitely often in one of these $k$ infinite words is odd.

Our coloring scheme is different from the typical one in the sense that it allows colors to be tuples of natural numbers as opposed to single ones. However, we can still obtain a set of state-set pairs, which would prohibit the same set of infinite runs by means of a Streett acceptance condition. Given the EMTS $\mathcal{E}$, the coloring function $c$ can be used to specify a set of state-set pairs $\Omega$ so that $(L_{ij}, U_{ij}) \in \Omega$ if and only if:

- $L_{ij} = \{s \in S_{\mathcal{E}} \mid c(s)(j) = 2 * i + 1\}$ where $1 \leq 2 * i + 1 \leq \mathit{max}_j$ and
- $U_{ij} = \{s \in S_{\mathcal{E}} \mid c(s)(j) = 2 * i' \wedge i' \geq i\}$

and $\mathit{max}_j$ is the largest number that occurs in the $j^{th}$ entry of the states of $S_{\mathcal{E}}$. In this way, a run is *not* prohibited only if the odd color in the $j^{th}$ entry of an infinitely often visited state is canceled out by infinitely often visiting a state which has a larger, even color in the same entry.

Next, we define a simulation relation between the states of an EMTS as a form of mixed fair simulation (cf. e.g. [19, 8]).

**Definition 4.2** (Simulation)**.** $R \subseteq S_{\mathcal{E}} \times S_{\mathcal{E}}$ is a *simulation relation* between the states of $\mathcal{E}$ if whenever $s_1 R s_2$ and $a \in A$:

1. if $s_1 \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} S_1$, then there is a $S_2$ such that $s_2 \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} S_2$ and for each $s_1' \in S_1$, there exists a $s_2' \in S_2$ such that $s_1' R s_2'$;
2. if $s_2 \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S_2$, then there is a $S_1$ such that $s_1 \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S_1$ and for each $s_1' \in S_1$, there exists a $s_2' \in S_2$ such that $s_1' R s_2'$;
3. if the run $\rho_{s_2} = s_2 \xrightarrow{a_1}_{\mathcal{E}} s_2^1 \xrightarrow{a_2}_{\mathcal{E}} s_2^2 \xrightarrow{a_3}_{\mathcal{E}} \ldots$ is in $W_{\mathcal{E}}$ then every infinite run $\rho_{s_1} = s_1 \xrightarrow{a_1}_{\mathcal{E}} s_1^1 \xrightarrow{a_2}_{\mathcal{E}} s_1^2 \xrightarrow{a_3}_{\mathcal{E}} \ldots$ such that $s_1^i R s_2^i$ for all $i \geq 1$ is also in $W_{\mathcal{E}}$.

We say that abstract state $s_2$ *simulates* abstract state $s_1$, denoted $s_1 \preceq s_2$, if there is a simulation relation $R$ such that $s_1 R s_2$. Simulation can be generalized to two different EMTSs $\mathcal{E}_1$ and $\mathcal{E}_2$ in the natural way.

Labeled transition systems can be viewed as a special kind of EMTS, where: $\longrightarrow^{\Box}_{\mathcal{E}} = \longrightarrow^{\Diamond}_{\mathcal{E}}$, the target sets of the transition relation are singleton sets of states, and the set of prohibited runs $W$ is empty.

We give the meaning of an abstract state relative to a given LTS, as the set of concrete LTS states simulated by the abstract state.

**Definition 4.3** (Denotation)**.** Let $\mathcal{E}$ be an EMTS, and let $\mathcal{T}$ be an LTS. The *denotation* of abstract state $s \in S_{\mathcal{E}}$ is the set $[\![s]\!]_{\mathcal{T}} \triangleq \{t \in S_{\mathcal{T}} \mid t \preceq s\}$. This notion is lifted to sets of abstract states $S' \subseteq S_{\mathcal{E}}$ in the natural way: $[\![S']\!]_{\mathcal{T}} \triangleq \bigcup\{[\![s]\!]_{\mathcal{T}} \mid s \in S'\}$.

In the rest of this thesis, we assume that EMTSs obey the following *consistency* restrictions: $\longrightarrow^{\Box}_{\mathcal{E}} \subseteq \longrightarrow^{\Diamond}_{\mathcal{E}}$, $s \overset{a}{\longrightarrow}^{\Box}_{\mathcal{E}} S$ implies $S$ is non-empty, and $W$ does not contain runs corresponding to infinite must–runs of the EMTS.

In Chapter 6, we present a proof system for proving properties of abstract states. For this purpose, we define when an abstract state $s$ satisfies a modal $\mu$-calculus formula $\Phi$. The global nature of the set $W$ in EMTSs makes it cumbersome to define the denotation of a fixed point formula compositionally as a set of abstract states. We therefore give an indirect definition of satisfaction, by means of the denotation $[\![s]\!]_{\mathcal{T}}$ of a state $s$.

**Definition 4.4** (Satisfaction)**.** Let $\mathcal{E}$ be an EMTS, $s \in S_{\mathcal{E}}$ be an abstract state of $\mathcal{E}$ and $\Phi$ be a modal $\mu$-calculus property. Then $s$ satisfies $\Phi$ under valuation $\mathcal{V} : \mathrm{PropVar} \to 2^{S_{\mathcal{E}}}$, denoted $s \models^{\mathcal{E}}_{\mathcal{V}} \Phi$, if and only if for any LTS $\mathcal{T}$ $[\![s]\!]_{\mathcal{T}} \models^{\mathcal{T}}_{\mathrm{V}} \Phi$ where valuation $\mathrm{V} : \mathrm{PropVar} \to 2^{S_{\mathcal{T}}}$ is induced by $\mathcal{V}$ as $\mathrm{V}(Z) \overset{\Delta}{=} \bigcup\{[\![s]\!]_{\mathcal{T}} \mid s \in \mathcal{V}(Z)\}$.

**Example.** The state space of the open system introduced in the previous section is captured by the EMTS in Figure 4.1. In Figures 4.1 and 4.2 start states of the EMTSs are marked by a green arrow and blue, red, green circles correspond to the state colors 0, 1 and 2, respectively. For any labeled transition system $\mathcal{T}$, the processes simulated by the state $s_1$ are those denoted by the open term $X : \mathbf{stab} \rhd X \parallel \mathit{Handler}$. The EMTS consists of six abstract states, each state denoting the set of processes which it simulates. For instance, states $s_5$ and $s_6$ in the example denote all processes which can engage in arbitrary interleavings of *in* and $\overline{out}$ actions, but so that *in* has to be enabled throughout while $\overline{out}$ has not. Infinite runs stabilizing on $\overline{out}$ actions are prohibited by the coloring of $s_3$ and $s_6$.

Consider the processes a) *fix* $A.in.A$, b)*fix* $A.(in.A + \overline{out}.(\mathit{fix}\ B.in.B))$ and c) *fix* $A.(in.A{+}\overline{out}.A)$, for which corresponding EMTSs are shown in Figure 4.2. In order to show that processes are denoted by the open system $X : \mathbf{stab} \rhd X \parallel \mathit{Handler}$, simulation relations between the start states of the EMTSs of processes and the start state of the EMTS of the open system should be established. With the help of these two figures, it is possible to see that the second process is in the denotation of this open system while the first and third processes are not:

1. The relation $R_1 = \{(t_1, s_1), (t_1, s_2)\}$ is not a simulation relation because of the second item of Definition 4.2. It is not possible to build a simulation relation that contains the pair $(t_1, s_1)$, since $t_1$ does not have any successors to be paired with $s_4$. Since the transition from $s_1$ to $s_4$ is a must transition for the action $\overline{out}$, in order to be simulated by $s_1$, $t_1$ should have an $\overline{out}$ successor.

2. The relation $R_2 = \{(t_2, s_1), (t_2, s_2), (t_2', s_4), (t_2', s_5)\}$ is a simulation relation for the second process and the open system. Furthermore, this is the only possible simulation relation that contains the pair $(t_2, s_1)$.

3. The relation $R_3 = \{(t_3, s_1), (t_3, s_2), (t_3, s_4), (t_3, s_5)\}$ is the obvious candidate for a simulation relation for the third process. $R_3$ is not a simulation relation

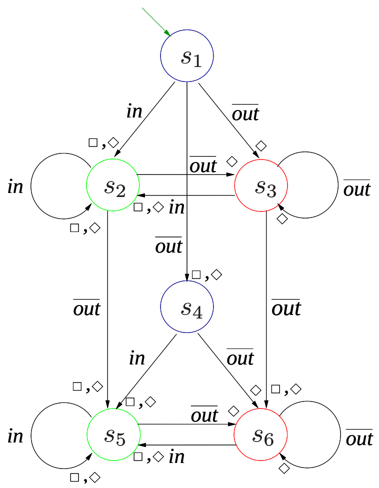


Figure 4.1: EMTS for $X : \boldsymbol{stab} \rhd X \parallel Handler$

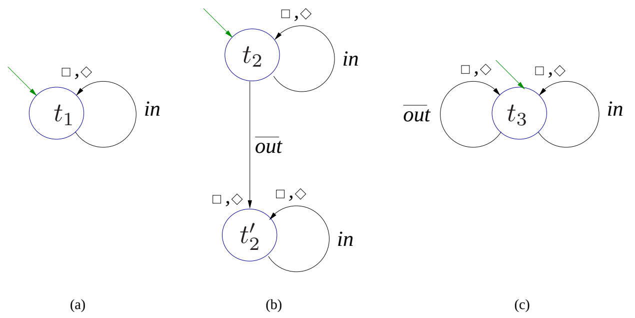


Figure 4.2: EMTSs for processes a) $\mathit{fix}\ A.in.A$, b)$\mathit{fix}\ A.(in.A + \overline{out}.(\mathit{fix}\ B.in.B))$ and c) $\mathit{fix}\ A.(in.A + \overline{out}.A)$

since the pair $(t_3, s_5)$ does not satisfy the third item of Definition 4.2. Because the color of state $s_3$ and the colors of both its $\overline{out}$-successors, $s_3$ and $s_6$, are odd, processes simulated by this state are not permitted to stabilize on $\overline{out}$. But $t_3$ can perform such a stabilizing run, hence $t_3$ is not simulated by $s_3$.

# Chapter 5

# From Specification to State Space Representation

We propose a two-phase construction $\varepsilon$ that translates an open term $\Gamma \rhd E$ to an EMTS, denoted $\varepsilon(\Gamma \rhd E)$. In the first phase, an EMTS is constructed for each underspecified component. This part is essentially a maximal model construction for the modal $\mu$-calculus. The second phase consists of combining the EMTSs produced in the first step according to the structure of the term $E$.

We will illustrate the construction with the use of examples. In the examples below, the set of actions is $A = \{a, b\}$. Blue, red and green circles around state names correspond to integers 0, 1 and 2 respectively and are used to indicate the color of the state. The number of circles around a state indicates the length of the color tuple for this state. The outermost circle around a state corresponds to the leftmost entry of its color, while the innermost circle corresponds to the rightmost. For example, a green outer circle in combination with a red inner circle means that the state has color (2,1). Color tuples are contracted into equivalent but shorter tuples when possible.

## 5.1 Maximal Model Construction

We define the function $\varepsilon$ which maps modal $\mu$-calculus formulae to triples of the shape $(\mathcal{E},\ S,\ \lambda)$, where $\mathcal{E} = (S_{\mathcal{E}}, A, \longrightarrow_{\mathcal{E}}^{\Diamond}, \longrightarrow_{\mathcal{E}}^{\Box}, c)$ is an *EMTS*, $S \subseteq S_{\mathcal{E}}$ is a set of *start states* of $\mathcal{E}$, and $\lambda : S_{\mathcal{E}} \to 2^{PropVar}$ is a *labeling function.* The function definition is inductive on the structure of $\Phi$ and can be found in Figure B.2 of Paper 2.

The EMTS for formula tt consists of the single state $s_{\mathrm{tt}}$ with may transitions to itself for every action (See Figure 5.1(a)), while the EMTS for ff is the empty EMTS. The EMTS for a propositional variable consists of a single state with may transitions to $s_{\mathrm{tt}}$ for each action. Essentially, the particular valuation used for propositional variables does not play a role in the final EMTS, since the properties

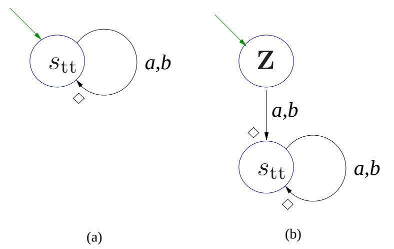


Figure 5.1: (a) $\varepsilon(\text{tt})$, (b) $\varepsilon(Z)$

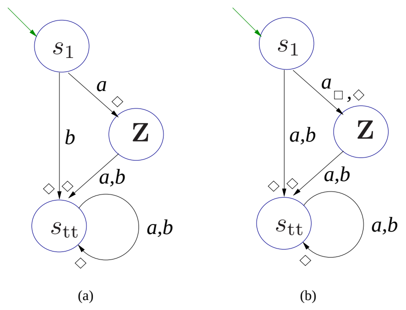


Figure 5.2: (a) $\varepsilon([a]\,Z)$, (b) $\varepsilon(\langle a\rangle\,Z)$

used as assumptions of an OTA are closed. Nevertheless, the meaning of open formulae that arises in intermediate steps are given by the by the valuation which assigns the whole set of processes $S_{\mathcal{T}}$ to each propositional variable. This is achieved by constructing the EMTS for the propositional variable $Z$ as a single start state, which has may transitions to $s_{\text{tt}}$ for each action (See Figure 5.1(b)).

For the modal cases, a new state $s_{new}$ is set as the start state. The EMTS for $\varepsilon([a]\,\Phi)$ has a single may transition for $a$, which is to the set of initial states of $\varepsilon(\Phi)$ (See Figure 5.2(a)). This is to ensure all simulated processes satisfy $\Phi$ after engaging in an $a$. Additionally, there is a may transition to $s_{\text{tt}}$ for all other actions. The EMTS for $\varepsilon(\langle a\rangle\,\Phi)$ includes a must transition for $a$ from this start state to the start states of $\varepsilon(\Phi)$, along with may transitions for all actions to $s_{\text{tt}}$ forcing the simulated processes to have an $a$ transition to some process satisfying $\Phi$ and allowing any other transitions besides (See Figure 5.2(b)).

The states of the EMTS for the conjunction of two formulae is the cross product of the states of the EMTSs constructed for each conjunct, excluding pairs with incompatible capabilities (See Figure 5.3(a)). The color of a state of $\varepsilon(\Phi_1 \wedge \Phi_2)$ is the concatenation of the colors of the paired states. In the case of disjunction, the set of start states of $\varepsilon(\Phi_1 \vee \Phi_2)$ is the union of the start states of $\varepsilon(\Phi_1)$ and $\varepsilon(\Phi_2)$ which reflects the union of their denotation (See Figure 5.3(b)). The color of a state is given by padding with 0's from either the left or right.

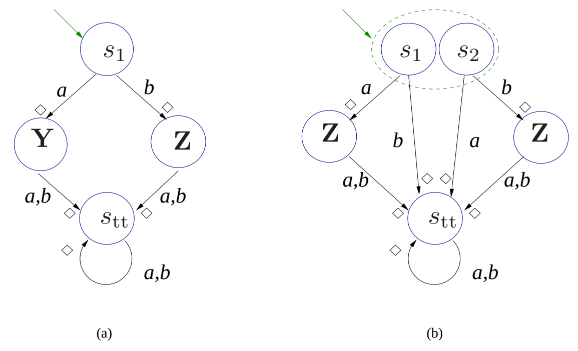


Figure 5.3: (a) $\varepsilon([a]\,Y \wedge [b]\,Z)$, (b) $\varepsilon([a]\,Z \vee [b]\,Z)$

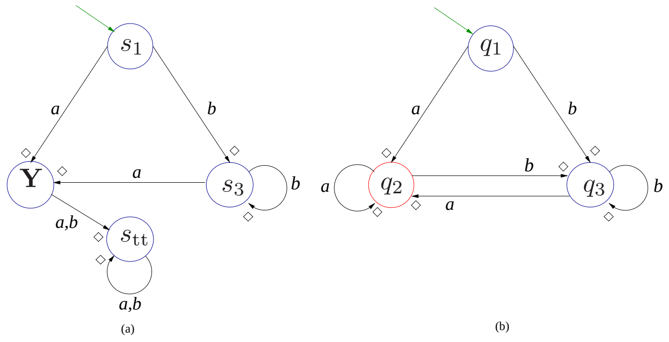


Figure 5.4: (a) $\varepsilon(\nu Z.\,[a]\,Y \wedge [b]\,Z)$, (b) $\varepsilon(\mu Y.\nu Z.\,[a]\,Y \wedge [b]\,Z)$

The construction for fixed point formulae is a powerset construction which is similar to the constructions given in [13] and [24] for the purpose of constructing Büchi Automata for linear time and the alternation-depth class $\Pi_2$ fragments of the $\mu$-calculus, respectively. The states of $\varepsilon(\sigma Z.\Phi)$ consist of sets of states of $\varepsilon(\Phi)$ and its start states are singletons containing some start state of $\varepsilon(\Phi)$. For a transition of state $q = \{s_1, \ldots, s_n\}$ of $\varepsilon(\sigma Z.\Phi)$, each state $s_i$ has a transition in $\varepsilon(\Phi)$. A member state of the target of this transition, then, contains a derivative for each $s_i$. A member of the target state additionally contains an initial state of $\varepsilon(\Phi)$ if one of the derivatives included is labeled by $Z$.

Each component of the color of state $q$ is determined by comparing the corresponding entries of the member states $s_i$. When for at least one of these states $s_i$, this entry is odd, the greatest of the corresponding odd entries is selected as the entry of $q$, otherwise the maximum entry is selected for the same purpose. The color of $q$ is further updated if it contains a state $s_i$ labeled by $Z$. When $Z$ identifies a greatest fixed point formula, each entry of the constructed tuple is defined to be the least even upper bound of the integers used in this entry of $\varepsilon(\Phi)$. Whereas,

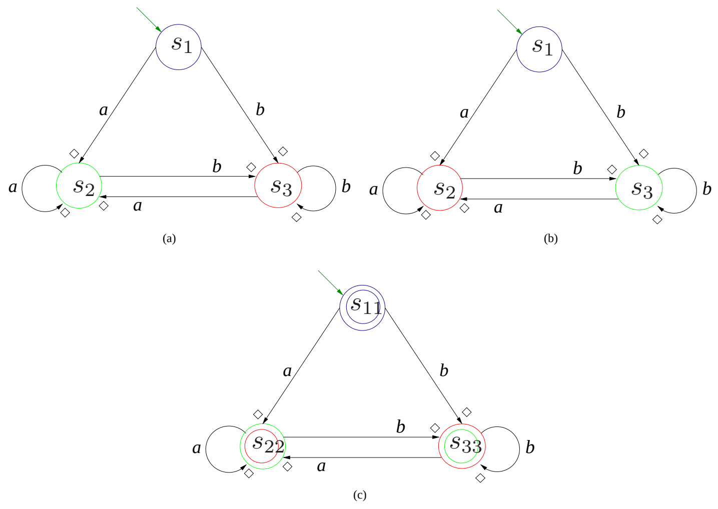


Figure 5.5: (a) $\varepsilon(\nu Y.\mu Z.\,[a]\,Y \wedge [b]\,Z)$, (b) $\varepsilon(\nu Z.\mu Y.\,[a]\,Y \wedge [b]\,Z)$, (c) $\varepsilon((\nu Y.\mu Z.\,[a]\,Y \wedge [b]\,Z) \wedge (\nu Z.\mu Y.\,[a]\,Y \wedge [b]\,Z))$

when $Z$ identifies a least fixed point formula, the least odd upper bound of the integers is the entry for the color of $q$. Figures 5.3(a) and 5.4(a,b) illustrate how the alternation of fixed points is handled. In this example, the innermost fixed point is a greatest fixed point which means that the color of the state labeled by the variable identifying this fixed point (**Z**) is not changed going from Figure 5.3(a) to Figure 5.4(a). On the other hand, the outer fixed point is a least fixed point therefore the least odd upper bound of the colors of Figure 5.4(a) is computed and the result (1) is used to color the state labeled with the variable that identifies this fixed point (**Y**) in Figure 5.4(b).

In Figure 5.5, an example which requires colors of states to be tuples with multiple entries is given.

This part of the construction potentially causes an exponential blow-up in the number of states. Ideally, an algorithm of this step would start with the set of start state singletons and grow the EMTS by computing the target of one transition at each step. Then, the average number of states would be much less since most of the subset-states are not reachable from the start states. In Figure 5.6, we can see how the state space grows from the state state singletons and Figure 5.7 shows the EMTS constructed.

## 5.2 Construction for Terms

We extend the function $\varepsilon$ to the domain of OTAs so that $\varepsilon(\Gamma \rhd E) = (\mathcal{E}, S, \lambda)$, where $\mathcal{E} = (S_{\mathcal{E}}, A, \longrightarrow^{\diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, c)$ is an EMTS, $S \subseteq S_{\mathcal{E}}$ is the set of *start states* of $\mathcal{E}$, and

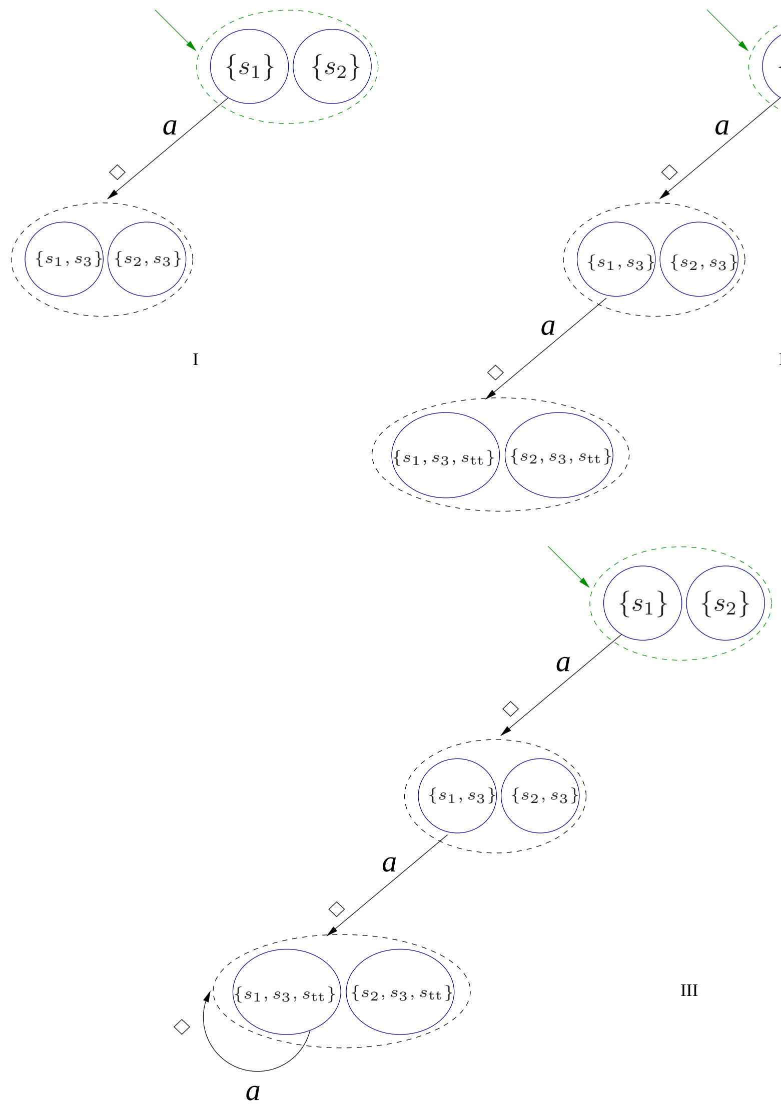


Figure 5.6: Three Steps of Constructing EMTS for $\mu Z. [a]\, Z \vee [b]\, Z$ from $\varepsilon([a]\, Z \vee [b]\, Z)$

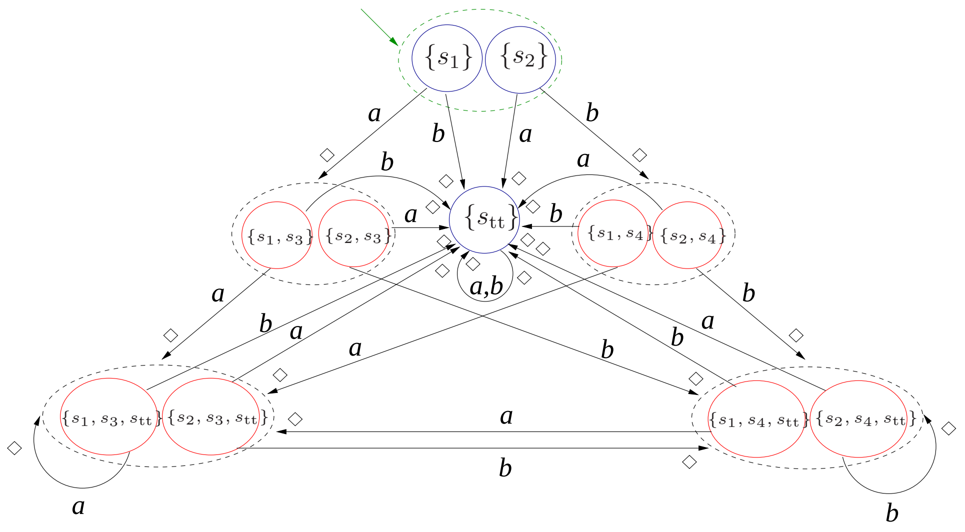


Figure 5.7: $\varepsilon(\mu Z. [a]\, Z \vee [b]\, Z)$

$\lambda : S_{\mathcal{E}} \to 2^{RecProcVar}$ is a *labeling function*.

The function $\varepsilon$ is defined inductively on the structure of $E$ as shown in Figure B.3. The EMTS corresponding to the nil process **0** consists of an abstract state without outgoing transitions, indicating that no transition is allowed for processes simulated by this state. If a process variable $X$ in the term $E$ stands for an underspecified component of the system, that is if $X$ is an assumption process variable, then the EMTS for $X$ is a maximal model for the conjunction of the properties specified for this component in the assumption list $\Gamma$.

The EMTS for a recursion process variable $X$ is a single state without outgoing transitions, since the capabilities of the processes simulated are determined by the binding *fix* -expression. The function $\lambda$ labels the state $X$. Given the EMTS for the term of the *fix* -expression where $X$ is free, the transitions of the start states are transferred to the states labeled by $X$.

The EMTS for a subterm prefixed by an action $a$ is given by a start state with a must $a$-transition to the set of start states of the EMTS for the subterm. The EMTS for the sum operator consists of an EMTS where the start states are the cross product of the start states of the EMTSs for the subterms. It is assumed for this case that there are no incoming transitions to the start states of the EMTSs being combined. This is an invariant of the construction, except the case for tt which can be trivially converted to an equivalent EMTS to satisfy the property.

Finally, the states of the EMTS for a parallel composition of two components consists of a state from each component. Each state has transitions such that one of the components make a transition while the other stays in the same state. Each state is further marked by 1 or 2 to keep track of which component has performed the last transition; this is necessary to enable a run of the composition if the interleaved runs are enabled.

## 5.3 Correctness Results

The aim of the above construction is to capture by means of an EMTS exactly those behaviors denoted by the given OTA. The construction is *sound* (resp. *complete*) if the denotation of the OTA is a subset (resp. superset) of the denotation of the resulting EMTS. Our first theorem establishes the soundness and completeness of the maximal model construction.

**Theorem 5.1.** *Let $\mathcal{T}$ be a transition-closed LTS, $\Phi$ be a closed and guarded modal $\mu$-calculus formula and $\varepsilon(\Phi) = (\mathcal{E}, S, \lambda)$. Then $[\![S]\!]_{\mathcal{T}} = ||\Phi||^{\mathcal{T}}$.*

**Proof** The proof is done by induction on the structure of the logical formula and can be found in Appendix C.

Our next result shows that the construction is sound and complete when assumptions exist on only one of the components that are running in parallel and the rest of the system is fully determined.

**Theorem 5.2.** *Let $\mathcal{T}$ be a transition-closed LTS, $\Gamma \rhd E \parallel t$ be a guarded linear OTA where $E$ does not contain parallel composition, and $t$ is closed, and let $\varepsilon(\Gamma \rhd E \parallel t) = (\mathcal{E}, S, \lambda)$. Then $[\![S]\!]_{\mathcal{T}}$ is equal to the set $[\![\Gamma \rhd E \parallel t]\!]_{\rho_0}$ up to bisimulation, where $\rho_0$ maps each recursion process variable $X$ to $\mathbf{0}$.*

Theorems 5.1 and 5.2 are proved by induction on the structure of the logical formula and the process term, respectively, and can be found in Appendix C.

In the general case, when multiple underspecified components run in parallel, we only have soundness: our construction is sound for systems without dynamic process creation. For systems with dynamic process creation, the construction does not terminate.

**Theorem 5.3.** *Let $\mathcal{T}$ be a transition-closed LTS, $\Gamma \rhd E$ be a guarded linear OTA where every recursion process variable in the scope of parallel composition is bound by a* fix *operator in the same scope, and let $\varepsilon(\Gamma \rhd E) = (\mathcal{E}, S, \lambda)$. Then the set $[\![S]\!]_{\mathcal{T}}$ includes $[\![\Gamma \rhd E]\!]_{\rho_0}$ up to bisimulation.*

The proof of the theorem is as the proof of Theorem 5.2, but includes a more general case for parallel composition and can be found in Appendix C.

**Example.** Take a system made up of two components that run in parallel with the only available actions being $a$ and $b$. The assumption on the first component is called **NeverDoesa** and means that this component can never perform action $a$ and dually the assumption on the second component, **NeverDoesb**, is that action $b$ is always disabled. The state space of the system constructed through $\varepsilon$ is given

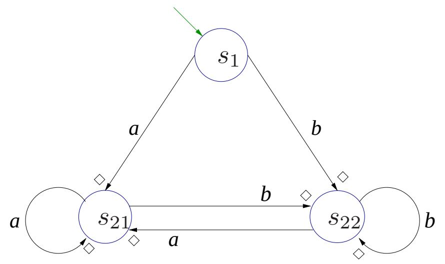


Figure 5.8: $\varepsilon(X : \mathbf{NeverDoesa}, Y : \mathbf{NeverDoesb} \rhd X \| Y)$

in Figure 5.8 after some simplifications. Unfortunately, the start state $s_1$ of this EMTS simulates any process and is clearly a proper superset of the intended set of processes. This state space also captures systems where an $a$ transition becomes available although it was initially disabled while trying to capture the fact that the first component may start at an arbitrary instant. This over-approximation makes it impossible to prove some simple properties of the open system through the constructed state space. One such property is $\langle a \rangle \, \mathrm{ff} \vee \langle b \rangle \, [a] \, \mathrm{tt}$ which states that either it is impossible to perform an $a$ initially or for each initial $b$-transition there exists a follower $a$-transition. Proving such a property of an EMTS requires the presence of a must transition.

Our last result reflects the fact that verification of open systems in the presence of parallel composition is undecidable for the modal $\mu$-calculus in general. Completeness results can, however, be obtained for various fragments of the $\mu$-calculus, such as ACTL, ACTL* and the simulation logic of [35]. In our approach, the tasks of constructing a finite representation of the state space in the form of an EMTS and the task of verifying properties of this representation are separated. This allows different logics to be employed for expressing assumptions on components and for specifying system properties, giving rise to more refined completeness results.

# Chapter 6

# Proof System

In this section we present the proof system we use for showing that a state of an EMTS satisfies a modal $\mu$-calculus property. Our proof system $\Sigma_{\mathcal{E}}$ is a specialization of a proof system $\Sigma_{\mathcal{T}}$ by Bradfield and Stirling for showing properties of sets of LTS states. It is sound and complete for prime formulae.

In both systems, a proof tree is constructed using the corresponding proof rules. The construction starts with the goal and progresses in a goal-directed fashion, checking at each step if a terminal node was reached. A successful tableau (or proof) is a finite proof tree having successful terminals as leaves. Below, we contrast the major components of the two proof systems for a better understanding: sequents, proof rules and conditions for being a successful/unsuccessful terminal, in particular discharge conditions for repeat nodes.

**Sequents** Sequents of $\Sigma_{\mathcal{T}}$ (left) include a set of LTS states $S$ while sequents of $\Sigma_{\mathcal{E}}$ (right) include a single state $s$ of the EMTS. $\Phi$ and $\Psi$ are modal $\mu$-calculus properties. The similarity of the sequents is natural since the abstract state s corresponds to a set of concrete states $[\![s]\!]_{\mathcal{T}}$, its denotation with respect to the labeled transition system, $\mathcal{T}$.

$$S \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \qquad\qquad s \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$$

**Rules** The rules of the two proof systems are shown in Figure 6.1. Common rules of the two proof systems reduce the goal in a similar manner. The rule of disjunction in our proof system is not as powerful as the one in Stirling's. When we are to show an abstract state s satisfies property $\Phi_1 \vee \Phi_2$, we have to choose one of $\Phi_1$ and $\Phi_2$ for s to satisfy since s can not be split. In our proof system, $\Sigma_{\mathcal{E}}$, we can show that the state s satisfies $\Phi_1 \vee \Phi_2$, only if $[\![s]\!]_{\mathcal{T}}$ satisfies one of these properties $\Psi_1$ and $\Psi_2$ in every $\mathcal{T}$. This results in our proof system to be *prime-complete* instead of complete.

| Name | $\Sigma_{\mathcal{T}}$ Rule | $\Sigma_{\mathcal{E}}$ Rule |
|---|---|---|
| $\wedge$ | $\dfrac{S \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \wedge \Phi_2}{S \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \quad S \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_2}$ | $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \wedge \Phi_2}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \quad s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_2}$ |
| $\vee$ | $\dfrac{S \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \vee \Phi_2}{S_1 \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \quad S_2 \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_2}\ S = S_1 \cup S_2$ | $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \vee \Phi_2}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1} \quad \dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \vee \Phi_2}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_2}$ |
| $\Box$ | $\dfrac{S \vdash^{\mathcal{T}}_{\mathrm{V}} [a]\,\Phi}{\partial_a(S) \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}$ | $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} [a]\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \ \ldots\ s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} \{s_1, ..., s_n\} = \cup\, \partial^{\Diamond}_a(s)$ |
| $\Diamond$ | $\dfrac{S \vdash^{\mathcal{T}}_{\mathrm{V}} \langle a \rangle\,\Phi}{f_a(S) \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ f_a : s \mapsto s' \in \partial_a(s)$ | $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \langle a \rangle\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \ \ldots\ s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi}\ \{s_1, \ldots, s_n\} \in \partial^{\Box}_a(s)$ |
| $\sigma Z$ | $\dfrac{S \vdash^{\mathcal{T}}_{\mathrm{V}} \sigma Z.\Phi}{S \vdash^{\mathcal{T}}_{\mathrm{V}} Z}$ | $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \sigma Z.\Phi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z}$ |
| $Z$ | $\dfrac{S \vdash^{\mathcal{T}}_{\mathrm{V}} Z}{S \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ Z$ identifies $\sigma Z.\Phi$ | $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} Z$ identifies $\sigma Z.\Phi$ |
| Thin | $\dfrac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}{\mathrm{R} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ \mathrm{S} \subset \mathrm{R}$ | |
| Cut | $\dfrac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}{\mathrm{S}_1 \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \quad \mathrm{S}_2 \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ \mathrm{S} = \mathrm{S}_1 \cup \mathrm{S}_2$ | |

In the rules above, $\sigma$ ranges over $\mu$ and $\nu$.

Figure 6.1: Proof Rules for $\Sigma_{\mathcal{T}}$ and $\Sigma_{\mathcal{E}}$

Proof trees (possibly) branch in $\Sigma_{\mathcal{E}}$ for $\Box$ and $\Diamond$-rules since each goal contains a single abstract state and not a set of states. The choice in the $\Diamond$-rule of $\Sigma_{\mathcal{T}}$ result in a goal with a single state, while the choice between $\Box$-successors in $\Sigma_{\mathcal{E}}$ results in a set of states. The Cut rule does not exist in the original proof system of Stirling. We extended $\Sigma_{\mathcal{T}}$ with the Cut rule in order to be able to reflect branchings of a proof tree of $\Sigma_{\mathcal{E}}$ in a proof tree of $\Sigma_{\mathcal{T}}$, when we translate proof trees for showing soundness and completeness of our proof system. Finally, we do not have a Thin rule. In order to have such a rule in $\Sigma_{\mathcal{E}}$, we would have to define when a state "includes" another, but for now we can only test two states for identity.

**Terminals** The conditions for being a terminal is also similar for $\Sigma_{\mathcal{T}}$ and $\Sigma_{\mathcal{E}}$ and can be found in Section A.3 of Paper 1 and Section B.5 of Paper 2, respectively. Here we will only look at the interesting case of discharge conditions for repeat nodes which somewhat differ from one another.

A node $n$ in a $\Sigma_{\mathcal{T}}$ proof tree labelled by a sequent $S \vdash_{\mathrm{V}}^{\mathcal{T}} \Psi$ is denoted $n : S \vdash_{\mathrm{V}}^{\mathcal{T}} \Psi$. If $n :\ S \vdash_{\mathrm{V}}^{\mathcal{T}} Z$ is a node where $Z$ identifies a fixed point formula $\sigma Z.\Phi$, and there is a ancestor node $n' :\ S' \vdash_{\mathrm{V}}^{\mathcal{T}} Z$ above $n$ with at least one application of a rule other than Thin and Cut in between, $S' \supseteq S$ and for any other fixed point variable $Y$ on this path, $Z$ subsumes $Y$, then node $n$ is called a $\sigma$-terminal. So no further rules are applied to it. The most recent node making $n$ a $\sigma$-terminal is called $n$'s companion. The conditions for being a $\sigma$-terminal and definition of companion node is similar in our proof system, where $\sigma$-terminals and their companions mention the same state. The proof systems differ in the way they determine whether a $\sigma$-terminal is successful or not.

The $\sigma$-terminal of $\Sigma_{\mathcal{T}}$, $R \vdash_{\mathrm{V}}^{\mathcal{T}} Z$, is a successful terminal if $Z$ identifies a greatest fixed point formula. If $Z$ identifies a least fixed point formula, for the terminal to be successful no infinite chain of composable trails $T_0 \circ T_1 \circ T_2 \ldots$ of companion node $n :\ S \vdash_{\mathrm{V}}^{\mathcal{T}} \Psi$ should exist. The notion of *trail* basically captures a path from a state in $S$ to a state in $R$ where each state in the path is a *dependent* of the previous one. These two concepts are defined below:

**Definition 6.1** (Dependent)**.** If node $n' :\ S' \vdash_{\mathrm{V}}^{\mathcal{T}} \Phi'$ is an immediate successor of node $n :\ S \vdash_{\mathrm{V}}^{\mathcal{T}} \Phi$, then state $s' \in S'$ at $n'$ is a *dependant* of state $s \in S$ at $n$ if:

- s $= s'$ and the rule applied to n is $\wedge$, $\vee$, $\sigma Z$, $Z$, or Thin, or
- $s \xrightarrow{a}_{\mathcal{T}} s'$ and the rule is $\Box_a$, or
- $s' = f_a(s)$ and the rule is $\Diamond_a$ applied with choice function $f_a$.

**Definition 6.2** ($\mathcal{T}$-Trail)**.** Assume that node $n_k{:}S_k \vdash_{\mathrm{V}}^{\mathcal{T}} Z$ is a $\mu$-terminal and node $n_0{:}S_0 \vdash_{\mathrm{V}}^{\mathcal{T}} Z$ is its companion. A trail $T$ of the companion node $n_0$ is a sequence of state–node pairs $(s_0, n_0), \ldots, (s_k, n_k)$ from state $s_0 \in S_0$ at $n_0$ to $s_k \in S_k$ at $n_k$, such that for all $0 \leq i < k$, one of the following holds:

1. $s_{i+1} \in S_{i+1}$ at $n_{i+1}$ is a dependent of $s_i \in S_i$ at $n_i$, or
2. $n_i$ is the immediate predecessor of a $\sigma$-terminal node $n' \neq n_k$ whose companion is $n_j$ for some $j : 0 \leq j \leq i$, and $n_{i+1} = n_j$ and $s_{i+1} \in S_{i+1}$ at $n'$ is a dependant of $s_i \in S_i$ at $n_i$.

Two trails $T_1$ and $T_2$ of the same companion node are *composable*, if the last pair of $T_1$ and the first pair of $T_2$ mention the same state; in this case their composition is denoted by $T_1 \circ T_2$.

The $\sigma$-terminal of $\Sigma_{\mathcal{E}}$, $r \vdash_{\mathcal{V}}^{\mathcal{E}} Z$, is a successful terminal if $Z$ identifies a greatest fixed point formula. If $Z$ identifies a least fixed point formula, for the terminal to be successful, for every *unique* trail $T_u$ of the companion node $n_0 : r \vdash_{\mathcal{V}}^{\mathcal{E}} Z$ there should exist $1 \leq j \leq k$ such that $\mathit{max}\,(c(\alpha(T_u))(j))$ is odd. This ensures, for an infinite run $w_{n_0} = \alpha(T_1) \circ \alpha(T_2) \circ \alpha(T_3) \ldots$ where for all $i \geq 1$, $T_i$ is a trail of $n_0$, that there exists some $1 \leq j' \leq k$ such that $\mathit{max}\,(\mathit{inf}\,(c(w_{n_0})(j')))$ is odd.

**Definition 6.3** (Unique $\mathcal{E}$-Trail)**.** A unique trail $T_u$ of the companion node $n_0$ is a sequence of state–node pairs $(r, n_0), \ldots, (r, n_k)$ such that for all $0 \leq i < k$, one of the following holds:

1. $n_{i+1} : r_{i+1} \vdash_{\mathcal{V}}^{\mathcal{E}} \Psi_{i+1}$ is an immediate successor of $n_i : r_i \vdash_{\mathcal{V}}^{\mathcal{E}} \Psi_i$, or
2. $n_i$ is the immediate predecessor of a $\sigma$-terminal node $n' : r' \vdash_{\mathcal{V}}^{\mathcal{E}} Z'$ where $n' \neq n_k$ whose companion is $n_j : r' \vdash_{\mathcal{V}}^{\mathcal{E}} Z'$ for only one $j : 0 \leq j \leq i$, $n_{i+1} = n_j$, and $r_{i+1} = r'$.

In order to convert a trail to a corresponding run, we use the function $\alpha$, which returns the empty string when the trail contains only one pair, and is defined for longer trails as follows:

$$\alpha((r_1, n_1) \cdot (r_2, n_2) \cdot T) \triangleq \begin{cases} (r_1 \xrightarrow{a}_{\mathcal{E}} r_2) \cdot \alpha((r_2, n_2) \cdot T) & \Box_a \text{ or } \Diamond_a\text{-rule is applied to } n_1 \\ \alpha((r_2, n_2) \cdot T) & \text{otherwise.} \end{cases}$$

Stirling's discharge condition aims to check that infinite behaviour banned by the least fixed point formula is not performed by the processes in the set mentioned in the companion node. Our discharge condition does the same thing in an indirect manner.

It is possible to give an algorithm for showing that an abstract state satisfies a formula. The application of the rules in our proof system are deterministic, except for the rules of disjunction and diamond, and in these cases the number of possible applications are finite. For the discharge condition, color sequences of all unique trails of a companion node should be checked to have a dominating odd entry. It is possible to construct all unique trails of a companion node (Note that this is not always possible, for the set of trails is possibly infinite). For a unique trail it is possible to "come back" to a state-node pair only once hence there are only finitely many unique trails for a terminal.

**Theorem 6.4** (Soundness)**.** *$s \vdash_{\mathcal{V}}^{\mathcal{E}} \Phi$ implies $s \models_{\mathrm{V}}^{\mathcal{E}} \Phi$.*

**Proof** The proof is performed by converting a proof tree in $\Sigma_{\mathcal{E}}$ to a proof tree in $\Sigma_{\mathcal{T}}$. This is done by translating each rule application in $\Sigma_{\mathcal{E}}$ to a sequence of rule applications in $\Sigma_{\mathcal{T}}$. The transformation $\pi$ used in this process can be found in Definition A.9 and is presented for the $\Diamond$-rule below.

After a corresponding proof tree in Stirling's proof system is created, it remains to show that if the terminals of the former tree are successful, the terminals of the latter tree will also be successful. The main effort goes in this case to show terminals which mention a least fixed point formula successful. This is achieved by ruling out certain infinite trails of concrete states that are simulated by the abstract state that appears in the terminal.

**Transformation:** $\pi$

Our Rule

$$\frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \langle a\rangle \Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \quad \dots \quad s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi}\ \{s_1,\dots,s_n\} \in \partial^{\Box}_a(s)$$

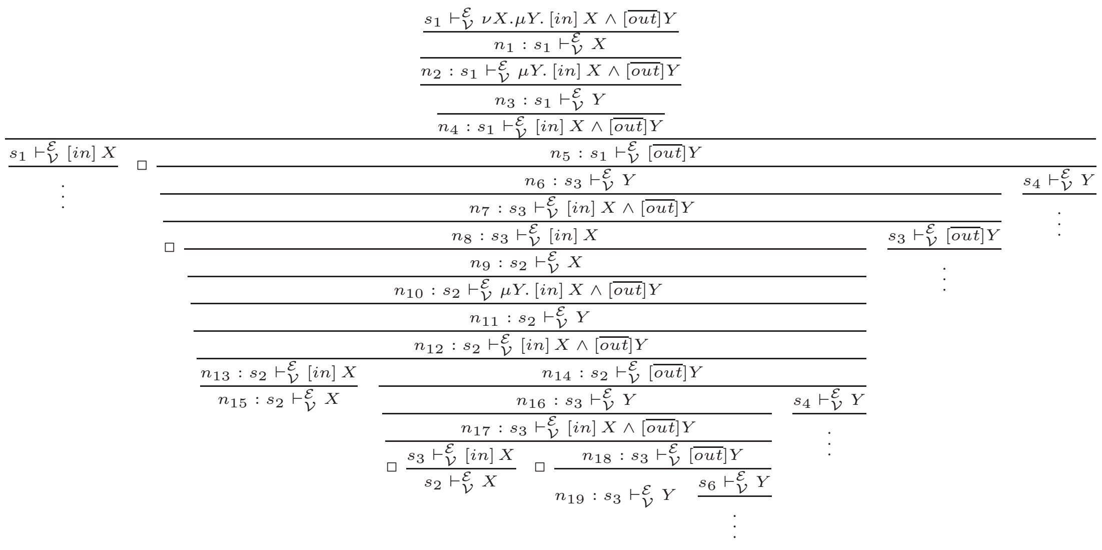
Corresponding Tree

$$\cfrac{\cfrac{\cfrac{\llbracket s\rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \langle a\rangle \Phi}{f_a(\llbracket s\rrbracket_{\mathcal{T}}) \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\Diamond_a}{\llbracket\{s_1,\dots,s_n\}\rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Thin*}}{\cfrac{\llbracket s_1\rrbracket_{\mathcal{T}} \cup\dots\cup \llbracket s_{n-1}\rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}{\cfrac{\vdots}{\llbracket s_1\rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \quad \llbracket s_2\rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Cut}}\,\text{Cut} \quad \llbracket s_n\rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Cut}$$

**Theorem 6.5** (Completeness)**.** *Let $\mathcal{E}$ be a finite–state EMTS, $s \in S_{\mathcal{E}}$, and let $\Phi$ have prime subformulae only. Then $s \models^{\mathcal{E}}_{\mathrm{V}} \Phi$ implies $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$.*

**Proof** The proof is carried by translating a proof tree in $\Sigma_{\mathcal{T}}$ to one in $\Sigma_{\mathcal{E}}$ when the former tree obeys certain conditions. The transformation is carried through the inverse translation $\pi^{-1}$.

**Example.** For the open system $X : \textbf{stab} \rhd X \parallel \mathit{Handler}$, a corresponding EMTS was given in Figure 4.1. Eventual stabilization of all processes denoted by the abstract state $s_1$ in this EMTS can be shown using $\Sigma_{\mathcal{E}}$. Here we present a part of this proof tree that is representative:



Node $n_{15}$ is discharged with companion node $n_9$ without appealing to colorings since $X$ identifies a greatest fixed point formula. To discharge node $n_{19}$ with

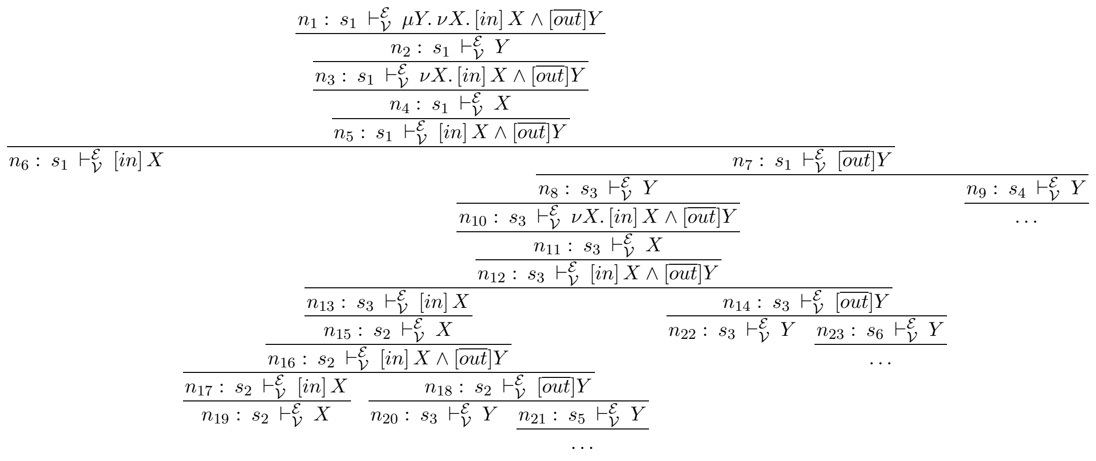

$n_1 : s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \mu Y.\, \nu X.\, [in]\, X \wedge [\overline{out}]Y$
$n_2 : s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} Y$
$n_3 : s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \nu X.\, [in]\, X \wedge [\overline{out}]Y$
$n_4 : s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} X$
$n_5 : s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\, X \wedge [\overline{out}]Y$
$n_6 : s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\, X$
$n_7 : s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} [\overline{out}]Y$
$n_8 : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} Y$
$n_9 : s_4 \vdash^{\mathcal{E}}_{\mathcal{V}} Y$
$\dots$
$n_{10} : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} \nu X.\, [in]\, X \wedge [\overline{out}]Y$
$n_{11} : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} X$
$n_{12} : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\, X \wedge [\overline{out}]Y$
$n_{13} : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\, X$
$n_{14} : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} [\overline{out}]Y$
$n_{15} : s_2 \vdash^{\mathcal{E}}_{\mathcal{V}} X$
$n_{22} : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} Y$
$n_{23} : s_6 \vdash^{\mathcal{E}}_{\mathcal{V}} Y$
$\dots$
$n_{16} : s_2 \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\, X \wedge [\overline{out}]Y$
$n_{17} : s_2 \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\, X$
$n_{18} : s_2 \vdash^{\mathcal{E}}_{\mathcal{V}} [\overline{out}]Y$
$n_{19} : s_2 \vdash^{\mathcal{E}}_{\mathcal{V}} X$
$n_{20} : s_3 \vdash^{\mathcal{E}}_{\mathcal{V}} Y$
$n_{21} : s_5 \vdash^{\mathcal{E}}_{\mathcal{V}} Y$
$\dots$

Figure 6.2: Unsuccessful Proof Tree Example

$n_{16}$, however, we need to make sure that all infinite runs of $\mathcal{E}$ corresponding to infinite sequences of trails of $n_{16}$ are in $W$. Node $n_{16}$ has only one unique trail $T_u = ((s_3, n_{16}), (s_3, n_{17}), (s_3, n_{18}), (s_3, n_{19}))$. The maximum color occurring in the corresponding run $c(s_3 \xrightarrow{\overline{out}} s_3)$ is 1. So we have that $max\,(c(\alpha(T_u))(1))$ is odd. Therefore we can conclude that the infinite run $s_3 \xrightarrow{\overline{out}} s_3 \xrightarrow{\overline{out}} \dots$, which this trail gives rise to, is prohibited. Hence the terminal is successful.

**Example.** As a further example, we show how an attempt to show the same system satisfies the property $stab2 = \mu Y.\, \nu X.\, [in]\, X \wedge [\overline{out}]Y$ fails. This property requires that the overall number of $\overline{out}$ actions is finite, though these may be interleaved with arbitrarily many *in* actions. This is not necessarily the case in this open system, for example if the process we plug in the process *fix* $A.(in.\overline{out}.A)$ for the underspecified component. This process satisfies the *stab* property and so is eligible to join the system. But the resulting system clearly violates the *stab*2 property defined above. Each *in* action is matched with at least one $\overline{out}$ action, thus infinitely many of the former action will result in infinitely many of the latter action. We expect to find no successful proof tree$_1$ for the goal $s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \mu Y.\, \nu X.\, [in]\, X \wedge [\overline{out}]Y$. Since there are no diamond modalities or disjunctions in the formula *stab*2, there is only one possible proof tree. We present a part of this tree in Figure 6.2, which includes the unsuccessful terminal node $n_{20}$.

In order for the terminal $n_{20}$ to be successful for every *unique* trail $T_u$ of the companion node $n_8$, there should exist $1 \leq j \leq k$ such that $max\,(c(\alpha(T_u))(j))$ is odd, since $Y$ identifies a least fixed point formula. The unique trails of $n_{20}$ and the corresponding partial runs are given below:

1. $T_{u1} = ((s_3, n_8), (s_3, n_{10}), (s_3, n_{11}), (s_3, n_{12}), (s_3, n_{13}), (s_2, n_{15}), (s_2, n_{16}), (s_2, n_{18}), (s_3, n_{20}))$
   $\alpha(T_{u1}) = s_3 \xrightarrow{in} s_2 \xrightarrow{\overline{out}} s_3$
2. $T_{u2} = ((s_3, n_8), (s_3, n_{10}), (s_3, n_{11}), (s_3, n_{12}), (s_3, n_{13}), (s_2, n_{15}), (s_2, n_{16}), (s_2, n_{17}), (s_2, n_{15}), (s_2, n_{16}), (s_2, n_{18}), (s_3, n_{20}))$
   $\alpha(T_{u2}) = s_3 \xrightarrow{in} s_2 \xrightarrow{in} s_2 \xrightarrow{\overline{out}} s_3$
3. $T_{u3} = ((s_3, n_8), (s_3, n_{10}), (s_3, n_{11}), (s_3, n_{12}), (s_3, n_{14}), (s_3, n_8), (s_3, n_{10}), (s_3, n_{11}), (s_3, n_{12}), (s_3, n_{13}), (s_2, n_{15}), (s_2, n_{16}), (s_2, n_{18}), (s_3, n_{20}))$
   $\alpha(T_{u3}) = s_3 \xrightarrow{\overline{out}} s_3 \xrightarrow{in} s_2 \xrightarrow{\overline{out}} s_3$
4. $T_{u4} = ((s_3, n_8), (s_3, n_{10}), (s_3, n_{11}), (s_3, n_{12}), (s_3, n_{14}), (s_3, n_8), (s_3, n_{10}), (s_3, n_{11}), (s_3, n_{12}), (s_3, n_{13}), (s_2, n_{15}), (s_2, n_{16}), (s_2, n_{17}), (s_2, n_{15}), (s_2, n_{16}), (s_2, n_{18}), (s_3, n_{20}))$
   $\alpha(T_{u4}) = s_3 \xrightarrow{\overline{out}} s_3 \xrightarrow{in} s_2 \xrightarrow{in} s_2 \xrightarrow{\overline{out}} s_3$

These trails show that the state $s_2$ occurs infinitely often in some infinite runs, which means that the color 2 will be occurring infinitely often. For all unique trails $T_{ui}$, where $1 \geq i \geq 4$, of the companion node $n_8$ with the terminal $n_{20}$, $max\,(c(\alpha(T_{ui}))(1))$ is 2, which is even. So the condition is not met and $n_{20}$ is not a successful terminal.

Terminal $n_{22}$ has the same companion node $n_8$. Similarly, all but one of the unique trails of $n_8$ with the terminal $n_{22}$ mention $s_2$. (The unique trail $T_{u'} = ((s_3, n_8), (s_3, n_{10}), (s_3, n_{11}), (s_3, n_{12}), (s_3, n_{14}), (s_3, n_{22}))$ and the corresponding run $\alpha(T_{u'})$=$s_3 \xrightarrow{\overline{out}} s_3$ does not mention $s_2$ so the condition is met for this trail with $j$=1.) Since unique trails that do not meet the condition exist, this terminal is also unsuccessful.

Chapter 7

# More Related Work

In this chapter, we present work that is not directly inspirational but is still closely related to our work. We mention some other extensions to MTSs that are in certain ways similar to EMTSs but were created for use in abstraction-refinement. Other methods that can be employed in open system verification like robust satisfaction and partial model checking are also summarized.

## 7.1 MTS extensions for Abstraction

*Abstraction* is a technique used to deal with the state space explosion problem in model checking [11]. In this method, knowledge about system and specification-to-be-met is used to extract a simplified model of the system. This model is *conservative* for the logic used to express the properties to be checked, so that if the abstract model satisfies a property it is guaranteed to hold in the concrete model. As a result, desired properties of the system can be checked on the simplified model. Abstraction aims to hide away details of the model and concentrate on the aspects necessary to verify a property in order to reduce the state space while still retaining necessary information to perform verification.

One crucial problem of the abstraction method is to come up with an appropriate abstract model of the original system. This process is not straightforward considering that the abstract model should both be simple enough to deliver efficiency and still be adequate for verification purposes. In *abstraction refinement*, this problem is solved by the iterative refinement of the abstract model with the feedback from the verification process. For instance, in counter-example guided abstraction refinement [10], verification begins with a relatively simple initial abstraction of the system. At each iteration, the abstract model is checked against the desired property. If it satisfies the property, the iterative process terminates with a positive answer since the abstract model is conservative. If the verification of the abstract model ends unsuccessfully, a counter-example for this model is produced. If this counter-example is also a counter-example of the original model, one can deduce that the formula does not hold for the original system, hence the iterative process terminates with a negative answer. If the counter-example turns out to be "spurious", the current abstract model is refined using the information extracted from the counter-example so that it can not perform this spurious run that does not exist in the original model.

MTSs have been used by Godefroid et al [18] as a representation for abstract systems where the aim was to check properties written in model $\mu$-calculus. Since the refinement preorder on MTSs preserves any such property, if an MTS satisfies a property, any implementation of the MTS also satisfies the property. Besides expressing over-approximations like conventional state-transition models [11], MTSs can be used as under-approximations thanks to the "lower bound" set on behaviour by must transitions. This enables both safety and liveness properties to be deduced. Additionally, MTSs allow three-valued analysis. In three-valued analysis, the result of checking if a state of the MTS satisfies a formula results in *true*, *false* or *indefinite*. Abstract models are designed to be conservative for *false* as well as *true* and refinement is performed only if the verification of the abstract model is *indefinite*.

Two structures inspired by the notion of MTS were designed for representing state space abstractions in abstraction refinement frameworks. Kripke modal transition system (KMTS), was introduced by Huth et al. [23]. KMTSs include two labeling functions $L_{must}$, $L_{may}$ that label states with atomic propositions. These specify an interval of propositions to be satisfied by each state besides the interval of transitions specified by may and must transitions. General Kripke modal transition systems (GKMTS) have *hyper* must-transitions, i.e. sets of states as targets to transitions, and are otherwise similar to KMTSs. The simulation relation, called *generalized mixed simulation*, between two states of an GKMTS is defined with an intention to preserve CTL properties. This relation is similar to our simulation relation as defined in Definition 4.2 without well-foundedness requirements as set by the third part of the definition. GKMTs were introduced for performing abstraction-refinement for CTL by Grumberg and Shoham [20].

## 7.2 Other Methods for the Verification of Open Systems

The term open system has been used in literature for referring to systems whose behaviour depends on its interaction with the environment, and is not fully determined by its internal state [21, 27, 3]. In this sense of the term, a property of an open system $M$ is a property that is satisfied by the composition of the system with any environment $M'$. For such an open system $M$ to satisfy property $\Phi$, then, the composed system $M \mid M'$ should satisfy the property. In [27], this notion of satisfaction is called *robust satisfaction* and is considered for systems where $M$ is given as a finite state (possibly nondeterministic) Moore machine that communicates with the environment via input and output variables. The problem is considered for the logics CTL, CTL* and $\mu$-calculus and is solved by reduction to the emptiness problem of alternating tree automata. Given a system as the process algebra term $E_M$, the problem of robust satisfaction can be stated in our framework as showing that the open system

$$X : \mathrm{tt} \rhd E_M \mid X$$

satisfies $\Phi$ where no unbound process variable occurs in $E_M$. It is important to note that, in our current framework transitions of the system $M$ can not be disabled by any other component $X$ that runs in parallel since we use CSP-like parallel composition without communication. In order to model this, one should use a synchronous notion of parallel composition.

Another method that takes advantage of the compositional nature of the system to deal with state space explosion is *partial model checking* [4]. In this method, computing the state space of a concurrent system is avoided by removing some component while transforming the specification accordingly. Given a process $t$ and a property $\Phi$, Andersen defines procedures to compute the *quotient property* $\Phi/t$, making use of knowledge from compositional reasoning. For any process $t'$ that $t$ is composed with, the quotient property $\Phi/t$ satisfies the following:

$$t'|t \models \Phi \qquad \Longleftrightarrow \qquad t' \models \Phi/t$$

The "only if" direction of this equivalence allows us to model check $t'$ against the quotient property $\Phi/t$ instead of model checking $t'|t$ against the original property $\Phi$. The task of model checking the system is then simplified, assuming the size of the quotient property is not much larger than the size of the original property. In order to keep the size of the quotient property reasonably small, it is simplified at each step with the help of heuristics.

Martinelli observes in [30] that "security protocols can be conveniently described by open systems." He gives two examples. In the first example, an attacker tries to listen to the conversation between the two agents $A$ and $B$. This can be modelled by the open system consisting of the process algebra terms $E_A$, $E_B$ which represent the agents and $X$, a placeholder for the attacker with unpredictable behaviour:

$$X : \mathrm{tt} \rhd E_A \mid E_B \mid X$$

The second example again mentions two parties willing to communicate, but this time one of them can not be trusted. Let $A$ and $B$ be these two agents and suppose $B$ has unknown behaviour and may try to exploit the protocol to gain advantage. If the agent $A$ is specified by the term $E_A$, in this open system $X$ is a placeholder for the possibly malicious agent $B$:

$$X : \mathrm{tt} \rhd E_A \mid X$$

In these two examples, it is possible to show that the communication is secure in any scenario through showing properties of the open systems above provided the agents are specified in a suitable process algebra and a suitable logic is used for expressing properties. However, in [30] the verification is performed in a slightly different manner. Only systems with one unknown component is considered. Suppose the known participants of the system are given as the term $E_S$ and the desirable property of the system is $\Phi$, then partial model checking techniques mentioned above are used to find a property $\Psi$ on the unknown component $X$ such that

$$X : \mathrm{tt} \rhd E_S \mid X \text{ has property } \Phi \qquad \Longleftrightarrow \qquad X \text{ has property } \Psi$$

Then, the remaining task is to find if $\Psi$ is satisfiable, that is to find if there exists a malicious partner which may have the property $\Psi$. In the first system above, for instance, this would mean that a potential attacker exists that can obtain the secret message. Although his verification method is different, Martinelli's work is important for us since it provides us with a possible application area.

Chapter 8

# Conclusion

## 8.1 Summary and Contribution

In this thesis, a finite structure is introduced that captures the state space of open systems when component assumptions are written in modal $\mu$-calculus. We provide a method that extracts such a structure from open system descriptions in the form of process algebra terms with assumptions, and show it sound for terms without dynamic process creation and complete for systems with a single underspecified component. We also adapt an existing proof system for the task of proving behavioural properties of open systems based on the given state space representation. Our proof system is sound and prime-complete.

A complete framework based on the state space representation is offered for verification of open systems. The process begins with specifying the open system by a process algebraic term with assumptions. Then, the state space representation is extracted from this description. Finally, open system properties can be checked on this representation. The main feature of the approach is to have the main focus of the verifying process on understanding the behaviour of the system rather than proving properties of it. System behaviour is captured in the form of a space space representation, so that the more tedious part of showing properties through this representation is automatable. This creates an advantage over other methods of open system verification in certain cases, especially when a visualization of the behaviour is important, for instance in debugging.

Below are the contributions of this thesis:

- A finite structure (EMTS) was introduced that is suitable for the representation of the state space of open systems since it supports:
  - visualization of the state space, i.e. the system behavior,
  - graphical specification,
  - state space exploration for interactive techniques,
  - verification,

    – proof reuse.

- A maximal model construction for the modal $\mu$-calculus was offered that builds an EMTS capturing the formula by composing EMTSs of subformulae.
- An automatic construction that extracts the state space of an open system from process algebraic system descriptions is offered. The construction is exact provided that the open system has a single unknown component and over-approximating when the open system does not include dynamic process creation,
- A sound and prime-complete proof system that can be used to prove properties using the state space representation.

**My contribution**

The initial idea of state space representation of open systems as a means of open system verification, analogical to the verification of closed systems, is due to my advisor Dilian Gurov. The novel notions presented here for achieving this purpose are joint work and have been developed in our common discussions. Finally, the workout of the construction of Chapter 5 and the workout of the proofs are due to me. The papers published as a result of the work have also been written jointly.

## 8.2 Future Work

*Characterization* The correctness results on the automatic construction of Chapter 5 can be made more precise in a number of ways. First of all, Theorem 5.3 states that the construction over-approximates if the open system includes parallel composition but there is no indication how much. We believe that any additions of must-transitions or the subtraction of may-transitions in an effort to make the constructed structure more precise would result in an under-approximation. Thus, our construction comes as close to capturing the open term as possible with an EMTS. A more practically interesting problem is to determine which temporal properties can still be shown using the constructed EMTS, even when it over-approximates. The question is, more formally put: for which temporal property $\Phi$, it is the case that the open system $\Gamma \rhd E$ satisfies $\Phi$ if the start states of the constructed EMTS satisfy $\Phi$ as well?

$$\forall \mathcal{T}.\varepsilon(\Gamma \rhd E) \text{ satisfies } \Phi \implies [\![\Gamma \rhd E]\!]_{\mathcal{T}} \text{ satisfies } \Phi$$

Another set of correctness results could be established considering different logics for expressing assumptions and open system properties.

*Parallel composition* In order to be able to handle more systems in our framework, it is essential to consider different types of parallel composition in specification and eventually in construction. Currently we consider only systems where components act in parallel but independently of each other, which leaves out many interesting open systems.

*Interactive exploration* Since the problem of verifying open systems in the presence of parallel composition when assumptions are expressed in modal $\mu$-calculus is undecidable (consider for instance dynamic process spawning [15]), user interaction is necessary to perform the verification task in general. Therefore, user interaction should be integrated to our framework in order to deal with a larger class of open systems. Interactive verification is to be performed as state space exploration. This corresponds to the symbolic execution of OTA in a stepwise manner. Some extensions to the notion of EMTS like sub and super states, and to the notion of OTA with transition assertions that keep historical information about component behaviour seems to be necessary for this purpose. Another major problem in this context is setting the colors of states to model terminating behavior, especially when fixed point alternation is present in some component assumption.

*Algorithms* Automatic construction of Chapter 5 will be formulated as an algorithm along with the proof system of Chapter 6. These will complete the automatization of the framework. An algorithm for the minimization of the constructed state space is also considered for visualization issues.

*Tool development* The analysis results and algorithms suggested above would make it possible to develop a tool for the verification of open systems. This tool is to contain the following features:

- *Specification*: The specification of the open system is to be given either in the form of an OTA or can be directly provided as an EMTS. The OTA may include assumptions expressed in different fragments of modal $\mu$-calculus.

- *Automatic State Space Extraction* The construction presented in Section 5 can be used for automatic extraction of the state space when the specification of the system is given as an OTA.

- *Interactive State Space Exploration* As an alternative, interactive exploration may be employed for forming the (partial) state space when automatic extraction is not possible. This feature is to appear similar to the "Simulation" command of Concurrency Workbench [12].

- *Visualize State Space* EMTSs are to be visualized by the aid of a graph visualizer. Minimization may be necessary for a comprehensible visualization.

- *Verification* Verification of modal $\mu$-calculus properties of the state space will be possible by an implementation of an algorithmic version of the proof system of Section 6.

*Case Studies* A tool based on our theory can be used for the evaluation of the approach through case studies. The possible applications are limited by the lack of modelling of data in our framework. Nevertheless, I think interesting examples that illustrate how helpful the tool is for working with open systems are not rare. We also plan more realistic case studies like applications in security, similar to those suggested in [30]. These require different logics and process algebra to be adapted in the framework.

.

# Appendix A

# Paper I

## Verification of Open Systems Based on Explicit State Space Representation

*Irem Aktug, Dilian Gurov*
KTH Communication and Information Technology
Stockholm, Sweden
{irem,dilian}@imit.kth.se

**Abstract**

When verifying behavioural properties of a system, one has sometimes to deal with components for which there is no implementation available yet, but only a specification of its behaviour. Such a *component specification* denotes a potentially infinite set of possible implementations of the component, which induces a potentially infinite set of possible system behaviours. A behavioural property of such an *open system* is then a property shared by *all* possible system behaviours. To support verification of open system properties based on state space exploration, these behaviours have to be suitably represented in a finite structure.

In this paper we propose a representation of the state space of open systems for which the behaviour of the yet unavailable components is specified in the modal $\mu$-calculus, and a proof system for verifying open system properties written in the same logic. Following earlier work on modular verification for various fragments of the logic, we base the state space representation on the notion of modal transition systems, suitably extended to deal with disjunctive assumptions and with least fixed point assumptions. The latter are captured through a set of prohibited infinite runs. The representation, called *extended modal transition system*, has been tailored to allow (a) convenient construction from an open system description, and (b) verification of open system properties. For the latter task, we adapt an existing proof system for infinite state systems due to Stirling and Bradfield. Soundness and (relative) completeness of our proof system are established

through a translation between the two systems. The problem of constructing a finite state space representation from a given open system description will be addressed elsewhere.

## A.1 Introduction

When verifying the behaviour of a system based on state space exploration, it is usually assumed that the system's implementation is available at verification time, and thus an accurate representation of the system's behavior can be extracted. Such a *closed world* assumption, however, cannot always be made, since sometimes the implementation of some of the components is not available at verification time. This is the case, for example, when certain components join the system after the latter has been put in operation, as for example when applications are loaded on a smart card after the card has been issued (see e.g. [35]). Such systems are usually referred to as *open*. Still, even if the implementation of certain components is not yet available, an open system can be verified on the basis of the components' specifications, i.e. the properties which they are assumed to satisfy. Furthermore, proving system properties relative to given assumptions on some of its components is a standard element of compositional reasoning. Such reasoning can for example be employed for verifying systems with dynamically changing configuration due to dynamic process spawning [14]. Finally, replacing a component implementation with a specification can be used as a means of abstraction in order to control state space explosion.

State space exploration based verification of open systems requires the state space of the open system to be represented in a suitable fashion. A component specification denotes a set of possible behaviours of the component. Hence, the state space of an open system corresponds to a (potentially infinite) set of possible behaviours, as induced by the component specifications. A behavioural property of an open system is then a property which is satisfied by all such behaviours. These behaviours have to be conveniently represented in a structure, which is at the same time suitable for verification. We propose *extended modal transition systems* (EMTS) for these purposes, when component specifications are written in the modal $\mu$-calculus. These are based on the notion of modal transition systems, introduced by Larsen [29, 6] as a graphical specification language. A similar structure, called a Kripke modal transition system, was first introduced by Huth [23], and later refined by Shoham and Grumberg [20] as a means for representing state space abstractions in an abstraction refinement framework for CTL. As in the latter approach, EMTSs have sets of states (instead of single states) as targets to transitions – an extension which is needed for dealing with disjunctive assumptions. In addition, we add well-foundedness constraints to the structure to handle least fixed point assumptions. These serve a purpose dual to the one of fairness constraints (see e.g. [11]).

The notion of EMTS is planned to be used in a larger context to be developed for the verification of open systems using state space exploration. The process begins with specifying the system as a process algebra term with assumptions on variables representing the underspecified components. This model is to be converted to an EMTS in such a way that, ideally, all possible behaviours of the model are captured by the EMTS. Finally, to check whether the system has a certain property, the property is checked for this EMTS.

*Related Work.* Besides the beforementioned work on modal transition systems and its variants, our work is closely related to the approach of using maximal model constructions for modular verification, as pioneered by Grumberg and Long [19] in the context of ACTL, extended to ACTL* by Kupferman and Vardi [26], and applied by Sprenger et al to the fragment of the modal $\mu$-calculus without least fixed points and diamond modalities [35]. These approaches provide an algorithmic solution to the problem of verifying open systems for the particular logics, all of which are proper fragments of the modal $\mu$-calculus, when the same logic is used for specifying the properties of both the yet unavailable components and the open system. For the modal $\mu$-calculus, however, no such algorithmic solution is possible. In [15], a proof system is presented to verify properties of open CCS terms parameterized on variables with modal mu-calculus assumptions. The proof is guided by the formula to be verified and the state space is implicit in the proof tree. In our approach, we separate the tasks of constructing a finite representation of the state space of an open system (to be addressed elsewhere) from the task of verifying properties of the representation (addressed here) allowing different fragments of the modal $\mu$-calculus to be explored for these two tasks. Having this seperation has additional benefits like state space visualisation and possibility of proof reuse.

*Organization.* In this paper, we first give a motivating example for EMTSs, followed by the definition of this notion. We also define a simulation relation in Section A.2 which specifies the set of states of a given labelled transition system (LTS) denoted by an EMTS state. Section A.3 gives background information about modal $\mu$-calculus along with a proof system due to Bradfield and Stirling [7, 36] for checking properties of labelled transition systems in this logic. In Section A.4 we introduce our proof system for verifying modal $\mu$-calculus properties of EMTS states, which is an adaptation of Bradfield and Stirling's. Soundness and relative completeness of our proof system is shown by providing a translation of proofs in our proof system to proofs in the other system. Finally, Section A.6 concludes the paper with an outline of future work.

## A.2 Extended Modal Transition Systems

*Motivation.* Consider a concurrent server that spawns off *Handle* processes for handling requests to produce a result, modelled by the action $\overline{out}$. Although *Handle* processes can receive further input, modelled by the action *in*, they do not process these. We would like to prove that such a system eventually stabilizes – i.e., always comes to a state where only input actions are enabled – provided that the server does not receive any new requests. Eventual stabilization can be formalized in the modal $\mu$-calculus as $\nu X.\mu Y.\,[in]\,X \wedge [\overline{out}]\,Y$. This system can be specified

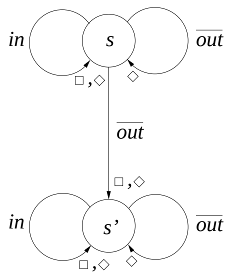


Figure A.1: EMTS representation of an open system.

as the open system, $x \mid \mathit{Handle}$, which consists of component *Handle* and any eventually stabilizing component $x$. In order to verify the above open system, we propose an abstract representation of its behaviour as the EMTS shown in Figure A.1. The EMTS consists of two "abstract" states, each state denoting the set of "concrete" states (processes) which it *simulates*, and two transition relations called *may* and *must*, depicted by $\longrightarrow^{\Diamond}$ and $\longrightarrow^{\Box}$ arrows, respectively. The must transitions are a subset of the may transitions, defining an interval of possible behaviours. Thus, state $s'$ in the example denotes all processes which can engage in arbitrary interleavings of *in* and $\overline{out}$ actions, but so that *in* has to be enabled throughout while $\overline{out}$ has not. In addition, the EMTS specifies explicitly a set $W$ of infinite runs which are deemed impossible. In this example, $W$ consists of the infinite runs stabilizing on $\overline{out}$ actions. A proof of eventual stabilization of the system using this representation can be found in Section A.4.

*EMTS.* We introduce the central notion of the present paper, aimed at representing the state space of an open system for the purposes of verifying temporal properties. The notion of EMTS is based on Modal Transition Systems (MTS) of Larsen [29], and borrows ideas from some generalizations [19, 20]. The formal definition of EMTS has been chosen to support in a natural way the construction of an EMTS from an open system specification: it includes may and must transitions to deal with modal assumptions, sets of states (instead of single states) as targets to transitions to deal with disjunctive assumptions, and well-foundedness constraints to deal with least fixed point assumptions.

**Definition A.1** (EMTS)**.** An *extended modal transition system* is a structure

$$\mathcal{E} = (S_{\mathcal{E}}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, W)$$

where (i) $S_{\mathcal{E}}$ is a set of abstract states, (ii) $A$ is a set of actions, (iii) $\longrightarrow^{\Diamond}_{\mathcal{E}}$, $\longrightarrow^{\Box}_{\mathcal{E}} \subseteq S_{\mathcal{E}} \times A \times 2^{S_{\mathcal{E}}}$ are *may* and *must* transition relations, and (iv) $W$ is a set of infinite $\mathcal{E}$-runs.

A *run*, or may–run, of $\mathcal{E}$ is a possibly infinite sequence of transitions $\rho_{\mathcal{E}} = s_0 \xrightarrow{a_0}_{\mathcal{E}} s_1 \xrightarrow{a_1}_{\mathcal{E}} s_2 \xrightarrow{a_2}_{\mathcal{E}} \ldots$ where for every $i \geq 0$, $s_i \overset{a_i}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}} S_e$ for some $S_e$ such that $s_{i+1} \in S_e$. Must–runs are defined similarly. We distinguish between two kinds of *a-derivatives*: $\partial_a^{\diamond}(s) \triangleq \{S \mid s \xrightarrow{a}{}^{\diamond}_{\mathcal{E}} S\}$, and $\partial_a^{\Box}(s) \triangleq \{S \mid s \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S\}$. For purposes of generality, the definition of $W$ is extentional; there are various possibilities for giving $W$ a finite characterization.

We define a simulation relation between the states of an EMTS, a form of fair simulation (cf. e.g. [19, 8]).

**Definition A.2** (Simulation)**.** $R \subseteq S_{\mathcal{E}} \times S_{\mathcal{E}}$ is a *simulation relation* between the states of an $\mathcal{E}$ if:

1. whenever $sRt$ and $a \in A$:

   a) if $s_1 \xrightarrow{a}{}^{\diamond}_{\mathcal{E}} S_1$, then there is a $S_2$ such that $s_2 \xrightarrow{a}{}^{\diamond}_{\mathcal{E}} S_2$ and for each $s_1' \in S_1$, there exists a $s_2' \in S_2$ such that $s_1' R s_2'$;

   b) if $s_2 \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S_2$, then there is a $S_1$ such that $s_1 \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S_1$ and for each $s_1' \in S_1$, there exists a $s_2' \in S_2$ such that $s_1' R s_2'$.

2. If $\rho_t = t \xrightarrow{a_1}_{\mathcal{E}} t_1 \xrightarrow{a_2}_{\mathcal{E}} t_2 \xrightarrow{a_3}_{\mathcal{E}} \ldots$ is in $W$, then every infinite run $\rho_s = s \xrightarrow{a_1}_{\mathcal{E}} s_1 \xrightarrow{a_2}_{\mathcal{E}} s_2 \xrightarrow{a_3}_{\mathcal{E}} \ldots$ such that $s_i R t_i$ for all $i \geq 1$ is in $W$.

The choice for (ii) is made to guarantee soundness of the proof system presented in Section A.4. We say that abstract state $t$ *simulates* abstract state $s$, denoted $s \preceq t$, if there is a simulation relation $R$ such that $sRt$. Simulation can be generalized to two different EMTSs $\mathcal{E}_1$ and $\mathcal{E}_2$ in the natural way.

The standard notion of *labelled transition system* (LTS) can be viewed as a special kind of EMTS, where: $\longrightarrow^{\Box}_{\mathcal{E}} = \longrightarrow^{\diamond}_{\mathcal{E}}$, the target sets of the transition relation are singleton sets of states, and set $W$ is empty. We give the meaning of an abstract state relative to a given LTS, as the set of concrete LTS states simulated by the abstract state.

**Definition A.3** (Denotation)**.** The *denotation* of an abstract state $s_{\mathcal{E}}$ w.r.t. a given LTS $\mathcal{T}$ is the set $[\![s_{\mathcal{E}}]\!]_{\mathcal{T}} \triangleq \{t \in S_{\mathcal{T}} \mid t \preceq s_{\mathcal{E}}\}$.

The notion is lifted to sets of abstract states $S' \subseteq S_{\mathcal{E}}$ in the natural way: $[\![S']\!]_{\mathcal{T}} \triangleq \bigcup\{[\![s]\!]_{\mathcal{T}} \mid s \in S'\}$. In the rest of the paper, we shall assume that EMTSs obey the following *consistency* restrictions: $\longrightarrow^{\Box}_{\mathcal{E}} \subseteq \longrightarrow^{\diamond}_{\mathcal{E}}$, $s \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S$ implies $S$ is non-empty, and $W$ does not contain runs corresponding to infinite must–runs of the EMTS.

## A.3 Modal $\mu$-Calculus

In this section we give a summary of the modal $\mu$-calculus [25], a process logic which we consider for specifying assumptions about the behavior of components, as well as for specifying the system properties to be verified. We present a known proof system, which we refer to for showing soundness and completeness of the proof system which we propose in the next section.

**Definition A.4** ($\mu$-calculus syntax)**.** A formula $\Phi$ of the modal $\mu$-calculus is generated by the following grammar, where $a$ ranges over a set of actions, and $Z$ over a set of propositional variables.

$$\Phi ::= \mathrm{tt} \mid \mathrm{ff} \mid Z \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\,\Phi \mid \langle a \rangle\,\Phi \mid \nu Z.\Phi \mid \mu Z.\Phi$$

A formula $\Phi$ is a *normal* formula if $\sigma_1 Z_1$ and $\sigma_2 Z_2$ are two different occurrences of binders in $\Phi$ then $Z_1 \neq Z_2$ and no occurrence of a free variable $Z$ is also used in a binder $\sigma Z$ in $\Phi$. Let $\Phi$ be a normal formula and $\sigma_1 X.\Psi_1$ and $\sigma_2 Z.\Psi_2$ be subformulas of $\Phi$, then $X$ *subsumes* $Z$ if $\sigma_2 Z.\Psi_2$ is a subformula of $\sigma_1 X.\Psi_1$.

The semantics of the $\mu$-calculus is standard (see e.g. [36]), defining a satisfaction relation $\models^{\mathcal{T}}_{\mathrm{V}}$ relative to a LTS $\mathcal{T} = (S_{\mathcal{T}}, A, \longrightarrow_{\mathcal{T}})$ and a valuation V mapping propositional variables $Z$ to a sets of states $\mathrm{V}(Z) \subseteq S_{\mathcal{T}}$. Satisfaction is then lifted to sets of states in the natural way.

The proof system $\Sigma_{\mathcal{T}}$ given below has been proposed by Bradfield and Stirling [7, 36] for showing that a set of states in a given LTS $\mathcal{T}$ (or, as in the original text, a set of closed CCS process terms) satisfies a property given as a $\mu$-calculus formula. A proof tree is constructed starting from the goal using the rules in a goal-directed fashion, checking at each step if a terminal node was reached. A proof tree is a proof if all the terminal nodes are successful. In the rules below, $\sigma$ ranges over $\mu$ and $\nu$. We give a single-action version of Bradfield and Stirling's $K$ notation. The original set of rules has been augmented with an admissable Cut rule for later purposes. The rules are presented in a goal-directed fashion.

The proof system is sound and complete (cf. [7, 36]). In the next section, we present a similar proof system for states of extended modal transition systems. We rely on the properties of the above proof system to show that our system enjoys these properties as well.

$$\wedge\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \wedge \Psi}{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \quad \mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi} \qquad\qquad \vee\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \vee \Psi}{\mathrm{S}_1 \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \quad \mathrm{S}_2 \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi}\ \mathrm{S} = \mathrm{S}_1 \cup \mathrm{S}_2$$

$$\sigma Z\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \sigma Z.\Phi}{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} Z} \qquad\qquad \Box_a\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} [a]\,\Phi}{\partial_a(\mathrm{S}) \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}$$

$$Z\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} Z}{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ Z \text{ identifies } \sigma Z.\Phi \qquad\qquad \Diamond_a\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \langle a \rangle\,\Phi}{f_a(\mathrm{S}) \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ f_a : \mathrm{s} \mapsto \mathrm{s}' \in \partial_a(\mathrm{s})$$

$$\text{Thin}\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}{\mathrm{R} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ \mathrm{S} \subset \mathrm{R} \qquad\qquad \text{Cut}\ \frac{\mathrm{S} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}{\mathrm{S}_1 \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \quad \mathrm{S}_2 \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\ \mathrm{S} = \mathrm{S}_1 \cup \mathrm{S}_2$$

A successful tableau (or proof) is a finite proof tree having successful terminals as leaves. A node $n$ in a proof tree labelled by a sequent $S \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi$ is denoted $n : S \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi$. If $n : S \vdash^{\mathcal{T}}_{\mathrm{V}} Z$ is a node where $Z$ identifies a fixed point formula $\sigma Z.\Phi$, and there is a ancestor node $n' : S' \vdash^{\mathcal{T}}_{\mathrm{V}} Z$ above $n$ with at least one application of a rule other than Thin and Cut in between, $S' \supseteq S$ and for any other fixed point variable $Y$ on this path, $Z$ subsumes $Y$, then node $n$ is called a $\sigma$-terminal. So no further rules are applied to it[1]. The most recent node making $n$ a $\sigma$-terminal is called $n$'s companion. The conditions for a leaf sequent $R \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi$ to be a successful (resp. unsuccessful) terminal are as follows

**Successful Terminals**

1. $\Psi = \mathrm{tt}$, or else $\Psi = Z$, $Z$ is free in the initial formula, and $R \subseteq \mathbf{V}(Z)$
2. $R = \emptyset$
3. $\Psi = Z$ where $Z$ identifies a fixed point formula $\sigma Z.\Phi$, and the leaf sequent is a $\sigma$-terminal with companion node $n : S \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi$, then
   a) If $\sigma = \nu$, then the terminal is successful.
   b) If $\sigma = \mu$, then the terminal is successful if there is no infinite chain of composable trails $T_0 \circ T_1 \circ T_2 \ldots$ of $n$ (see Definition A.6 below).

**Unsuccessful Terminals**

1. $\Psi = \mathrm{ff}$, or $\Psi = Z$ and $Z$ is free in the initial formula with R $\not\subseteq \mathbf{V}(Z)$
2. $\Psi = \langle a \rangle \, \Phi$ and for some $r \in R$, $\partial_a(\{r\}) = \emptyset$
3. $\Psi = Z$ where $Z$ identifies a minimal fixed point formula $\mu Z.\Phi$, there is an ancestor node $S \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$ with at least one application of a rule other than Thin and Cut in between, $S \subset R$ and for any other fixed point variable $Y$ on this path, $Z$ subsumes $Y$.

We now explain the notion of trail used in the above definition.

**Definition A.5** (Dependent)**.** If node $n' : S' \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi'$ is an immediate successor of node $n : S \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi$, then state $s' \in S'$ at $n'$ is a *dependant* of state $s \in S$ at $n$ if:

- s = $s'$ and the rule applied to n is $\wedge$, $\vee$, $\sigma Z$, $Z$, or Thin, or
- $s \xrightarrow{a}_{\mathcal{T}} s'$ and the rule is $\Box_a$, or
- $s' = f_a(s)$ and the rule is $\Diamond_a$ applied with choice function $f_a$.

[1]In order to show completeness for our proof system, we relax this condition without harming the soundness and completeness results.

**Definition A.6** ($\mathcal{T}$-Trail)**.** Assume that node $n_k{:}S_k \vdash^{\mathcal{T}}_{\mathrm{V}} Z$ is a $\mu$-terminal and node $n_0{:}S_0 \vdash^{\mathcal{T}}_{\mathrm{V}} Z$ is its companion. A trail $T$ of the companion node $n_0$ is a sequence of state–node pairs $(s_0, n_0), \ldots, (s_k, n_k)$ from state $s_0 \in S_0$ at $n_0$ to $s_k \in S_k$ at $n_k$, such that for all $0 \leq i < k$, one of the following holds:

1. $s_{i+1} \in S_{i+1}$ at $n_{i+1}$ is a dependent of $s_i \in S_i$ at $n_i$, or

2. $n_i$ is the immediate predecessor of a $\sigma$-terminal node $n' \neq n_k$ whose companion is $n_j$ for some $j : 0 \leq j \leq i$, and $n_{i+1} = n_j$ and $s_{i+1} \in S_{i+1}$ at $n'$ is a dependant of $s_i \in S_i$ at $n_i$.

Two trails $T_1$ and $T_2$ of the same companion node are *composable*, if the last pair of $T_1$ and the first pair of $T_2$ mention the same state; in this case their composition is denoted by $T_1 \circ T_2$.

Trails mention state–node pairs. Later in this paper, we will work with actual sequences of transitions (runs). We therefore define a mapping $\alpha$ to extract the corresponding run from a given trail.

**Definition A.7** ($\alpha$: Trail to Run Conversion)**.** Let $T = (s_0, n_0), \ldots, (s_k, n_k)$ be a $\mathcal{T}$-trail of proof tree $\Sigma$. The corresponding run $\alpha(T)$ is inductively defined as follows:

$$\alpha(s, n) \triangleq \varepsilon$$

$$\alpha((s_1, n_1) \cdot (s_2, n_2) \cdot T) \triangleq \begin{cases} (s_1 \xrightarrow{a}_{\mathcal{T}} s_2) \cdot \alpha((s_2, n_2) \cdot T) & \begin{array}{c}\Box_a \text{ or } \Diamond_a\text{-rule} \\ \text{is applied to } n_1\end{array} \\ \alpha((s_2, n_2) \cdot T) & \text{otherwise.} \end{cases}$$

## A.4 A Proof System for EMTSs

Satisfaction of a temporal property by an EMTS state cannot be defined directly. The reason for this is the presence of the set $W$ in EMTSs, which makes it impossible to define the denotation of a fixed point formula compositionally (that is, inductively on the structure of the formula). We therefore give an indirect definition of satisfaction by means of the denotation $[\![s]\!]_{\mathcal{T}}$ of a state $s$.

**Definition A.8.** Let $\mathcal{E}$ be an EMTS, and let $s \in S_{\mathcal{E}}$. We define satisfaction $s \models^{\mathcal{E}}_{\mathrm{V}} \Phi$ to hold if and only if $[\![s]\!]_{\mathcal{T}} \models^{\mathcal{T}}_{\mathrm{V}} \Phi$ holds for any LTS $\mathcal{T}$ and valuation V mapping propositional variables $Z$ to sets of states $\mathrm{V}(Z) \subseteq S_{\mathcal{E}}$, where valuation V is defined via V by $\mathrm{V}(Z) \triangleq \bigcup\{[\![s]\!]_{\mathcal{T}} \mid s \in \mathrm{V}(Z)\}$.

We present a proof system $\Sigma_{\mathcal{E}}$ for verifying whether a state $s$ of an EMTS $\mathcal{E}$ satisfies a modal $\mu$-calculus formula $\Phi$. Since the transitions of the EMTS result in sets of states, the latter are immediately split into separate goals.

$$\frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \wedge \Psi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \quad s \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi} \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \vee \Psi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \vee \Psi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi}$$

$$\frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \sigma Z.\Phi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z} \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} [a]\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \ \ldots \ s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi}\,\{s_1,\ldots,s_n\} = \cup\, \partial^{\Diamond}_a(s)$$

$$\frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} Z \text{ identifies } \sigma Z.\Phi \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \langle a\rangle\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \ \ldots \ s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi}\,\{s_1,\ldots,s_n\} \in \partial^{\Box}_a(s)$$

The conditions for a leaf node of a proof tree in $\Sigma_{\mathcal{E}}$ to be a successful terminal are similar to the ones specified for $\Sigma_{\mathcal{T}}$ (see Section A.3), and are even somewhat simpler since single states are considered instead of sets of states.

Considering node $r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$:

**Successful Terminals**

1. $\Psi = \mathrm{tt}$, or else $\Psi = Z$, $Z$ is free in the initial formula, and $r \in \mathrm{V}(Z)$
2. $\Psi = [a]\,\Phi$ and $\cup\partial^{\Diamond}_a(r) = \emptyset$
3. $\Psi = Z$ where $Z$ identifies a fixed point formula $\sigma Z.\Phi$, and the sequent is a $\sigma$-terminal with companion node $n:\ r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$, then
   a) If $\sigma = \nu$, then the terminal is successful.
   b) If $\sigma = \mu$, then the terminal is successful if every infinite run $w_{n_0}$ of the EMTS that corresponds to an infinite sequence of $\mathcal{E}$-trails of the companion node $n_0$ is in $W$. That is, if $w_{n_0} = \alpha(E_1) \circ \alpha(E_2) \circ \alpha(E_3) \ldots$ is such that the first state of trail $E_i$ is $r$ for all $i \geq 1$, then $w_{n_0} \in W$ (the notion of $\mathcal{E}$-trail is explained below).

**Unsuccessful Terminals**

1. $\Psi = \mathrm{ff}$, or else $\Psi = Z$, $Z$ is free in the initial formula, and $r \notin \mathrm{V}(Z)$
2. $\Psi = \langle a\rangle\,\Phi$ and $\cup\partial^{\Box}_a(r) = \emptyset$
3. $\Psi = Z$ where $Z$ identifies the minimal fixed point formula $\mu Z.\Phi$, and the sequent is a $\sigma$-terminal with companion node $n:\ r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$, then the terminal is unsuccessful if some infinite run $w_{n_0}$ of the EMTS that corresponds to an infinite sequence of $\mathcal{E}$-trails of the companion node $n_0$ is not in $W$.

The notion of $\mathcal{E}$-trails of an EMTS is defined analogously to $\mathcal{T}$-trails of an LTS. Since the nodes of the proof trees mention single states, the notion of dependant becomes superfluous. What is more, $\mathcal{E}$-trails of a node always begin and end with

$$\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \nu X.\mu Y.\,[in]\,X \wedge [\overline{out}]Y}{\dfrac{n_1 : s \vdash^{\mathcal{E}}_{\mathcal{V}} X}{\dfrac{n_2 : s \vdash^{\mathcal{E}}_{\mathcal{V}} \mu Y.\,[in]\,X \wedge [\overline{out}]Y}{\dfrac{n_3 : s \vdash^{\mathcal{E}}_{\mathcal{V}} Y}{\dfrac{n_4 : s \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\,X \wedge [\overline{out}]Y}{\Box\,\dfrac{n_5 : s \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\,X}{n_6 : s \vdash^{\mathcal{E}}_{\mathcal{V}} X} \quad \Box\,\dfrac{n_7 : s \vdash^{\mathcal{E}}_{\mathcal{V}} [\overline{out}]Y}{n_8 : s \vdash^{\mathcal{E}}_{\mathcal{V}} Y \quad \dfrac{n_9 : s' \vdash^{\mathcal{E}}_{\mathcal{V}} Y}{\dfrac{n_{10} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\,X \wedge [\overline{out}]Y}{\Box\,\dfrac{n_{11} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\,X}{\dfrac{n_{12} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} X}{\dfrac{n_{13} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} \mu Y.\,[in]\,X \wedge [\overline{out}]Y}{\dfrac{n_{14} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} Y}{\dfrac{n_{15} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\,X \wedge [\overline{out}]Y}{\Box\,\dfrac{n_{16} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} [in]\,X}{n_{17} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} X} \quad \Box\,\dfrac{n_{18} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} [\overline{out}]Y}{n_{19} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} Y}}}}}} \quad \Box\,\dfrac{n_{20} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} [\overline{out}]Y}{n_{21} : s' \vdash^{\mathcal{E}}_{\mathcal{V}} Y}}}}}}}}}$$

Figure A.2: Proof Tree Example

the same state, and are thus always composable. The mapping $\alpha$ from Definition A.7 which converts an $\mathcal{T}$-trail to an $\mathcal{T}$-run can be applied exactly the same way to $\mathcal{E}$-trails and proofs trees in $\Sigma_{\mathcal{E}}$.

Consider the example from the Introduction, where the set $W$ consists of all infinite runs which have the suffix $s \xrightarrow{\overline{out}} s \xrightarrow{\overline{out}} \ldots$ or $s' \xrightarrow{\overline{out}} s' \xrightarrow{\overline{out}} \ldots$. Using the above proof system, eventual stabilization of all processes denoted by the abstract state $s$ is established with the successful tableau in Figure A.2.

Node $n_6$ is discharged with companion node $n_1$, and similarly $n_{17}$ is discharged with $n_{12}$ without appealing to $W$ since $X$ identifies a greatest fixed point formula. To discharge node $n_8$ with $n_3$, however, we need to make sure that all infinite runs of $\mathcal{E}$ corresponding to infinite sequences of $\mathcal{E}$-trails of $n_3$ are in $W$. Node $n_3$ has only one $\mathcal{E}$-trail, giving rise to a single infinite run $s \xrightarrow{\overline{out}} s \xrightarrow{\overline{out}} \ldots$. Since this run is in $W$, the terminal is successful. Node $n_{19}$ with companion $n_{14}$, and node $n_{21}$ with companion $n_9$ are discharged similarly.

## A.5 Soundness and Completeness

To prove soundness of the proof system $\Sigma_{\mathcal{E}}$ presented above, we show that every successful tableau for sequent $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ in $\Sigma_{\mathcal{E}}$ can be transformed into a successful tableau for sequent $[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathcal{V}} \Phi$ in $\Sigma_{\mathcal{T}}$ for any given $\mathcal{T}$. Soundness of $\Sigma_{\mathcal{E}}$ follows then from the soundness of $\Sigma_{\mathcal{T}}$ by Definition A.8. Full proofs of results in this section has been left out due to lack of space and can be found in [1].

We achieve this transformation by defining a translation from rule instances in $\Sigma_{\mathcal{E}}$ to proof trees in $\Sigma_{\mathcal{T}}$ and we show that its extension to proof trees in $\Sigma_{\mathcal{E}}$

| | |
|---|---|
| $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \wedge \Phi_2}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \quad s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_2}$ | $\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \wedge \Phi_2}{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \quad [\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_2}\wedge$ |
| $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \vee \Phi_2}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1}$ | $\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \vee \Phi_2}{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1}\vee$ |
| $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_1 \vee \Phi_2}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_2}$ | $\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_1 \vee \Phi_2}{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_2}\vee$ |
| $s \vdash^{\mathcal{E}}_{\mathcal{V}} [a]\,\Phi$ and $\cup\,\partial^{\Diamond}_a(s) = \emptyset$ | $\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} [a]\,\Phi}{\emptyset \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\Box_a$ |
| $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} [a]\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \quad \ldots \quad s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi}\ \{s_1,\ldots,s_n\} = \cup\,\partial^{\Diamond}_a(s)$ | $\dfrac{\dfrac{\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} [a]\,\Phi}{\partial_a([\![s]\!]_{\mathcal{T}}) \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\Box_a}{[\![\cup\,\partial^{\Diamond}_a(s)]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Thin*}}{\dfrac{[\![s_1]\!]_{\mathcal{T}} \cup \ldots \cup [\![s_{n-1}]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}{\dfrac{\vdots}{[\![s_1]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \quad [\![s_2]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Cut}}\,\text{Cut} \qquad [\![s_n]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Cut}$ |
| $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \langle a\rangle\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \quad \ldots \quad s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi}\ \{s_1,\ldots,s_n\} \in \partial^{\Box}_a(s)$ | $\dfrac{\dfrac{\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \langle a\rangle\,\Phi}{f_a([\![s]\!]_{\mathcal{T}}) \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\Diamond_a}{[\![\{s_1,\ldots,s_n\}]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Thin*}}{\dfrac{[\![s_1]\!]_{\mathcal{T}} \cup \ldots \cup [\![s_{n-1}]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}{\dfrac{\vdots}{[\![s_1]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi \quad [\![s_2]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Cut}}\,\text{Cut} \qquad [\![s_n]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\text{Cut}$ |
| $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi}\,Z$ identifies $\sigma Z.\Phi$ | $\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} Z}{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi}\,\sigma Z$ |
| $\dfrac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \sigma Z.\Phi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z}$ | $\dfrac{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \sigma Z.\Phi}{[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} Z}\,Z$ |

Figure A.3: Translation $\pi_{\mathcal{T}}$

translates successful tableaux in $\Sigma_{\mathcal{E}}$ to successful tableaux in $\Sigma_{\mathcal{T}}$.

**Definition A.9** (Translation)**.** Translation $\pi_{\mathcal{T}}$, mapping rule instances in $\Sigma_{\mathcal{E}}$ to proof trees in $\Sigma_{\mathcal{T}}$ for a given $\mathcal{T}$, is defined through Figure A.3. In the definition, $f_a$ is a choice function.

## Soundness

**Lemma A.10** (Correctness)**.** *For each rule instance in $\Sigma_{\mathcal{E}}$, translation $\pi_{\mathcal{T}}$ assigns a correct proof tree in $\Sigma_{\mathcal{T}}$, so that each premise (resp. conclusion), $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$, of the rule is matched by a leaf (resp. root), $[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi$, and all unmatched leaves of the constructed proof tree are successful terminals.*

Translation $\pi_{\mathcal{T}}$ extends to proof trees in $\Sigma_{\mathcal{E}}$ in the obvious way as defined by the matching sequents.

**Corollary A.11.** *For proof tree $A_{\mathcal{E}}$ with root sequent $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ in $\Sigma_{\mathcal{E}}$, $A_{\mathcal{T}} = \pi_{\mathcal{T}}(A_{\mathcal{E}})$ is a proof tree with root sequent $[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi$ in $\Sigma_{\mathcal{T}}$, such that each leaf $s_i \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_i$ of $A_{\mathcal{E}}$ is matched by a leaf $[\![s_i]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_i$ in $A_{\mathcal{T}}$, and all unmatched leaves in $A_{\mathcal{T}}$ are successful terminals.*

**Definition A.12** ($\beta$:Trail Translation)**.** Given a proof tree $A_{\mathcal{E}}$ with nodes labeled $m_0 \dots m_l$, the function $\beta$ converts a trail, $T = (t_0, n_0), \dots, (t_k, n_k)$, of the corresponding LTS tree $\pi_{\mathcal{T}}(A_{\mathcal{E}})$ to the $\mathcal{E}$-trail, $E = (s_0, n_0), \dots, (s_{k'}, n_{k'})$ by replacing each node with the matching one and deleting the remaining ones:

$$\begin{aligned} \beta(\varepsilon) &\triangleq \varepsilon \\ \beta((t_1, n_1) \cdot T) &\triangleq \begin{cases} ((s_{1'}, n_{1'}) \cdot \beta(T)) & \text{if } n_{1'} \text{ matches } n_1 \text{ and } s_{1'} \text{ is in } n_{1'} \\ \beta(T) & \text{if no node matches } n_1 \text{ in } A_{\mathcal{E}} \end{cases} \end{aligned}$$

Notice that whenever a pair $(t_i,\, n_i)$ is replaced by $(s_{i'}, n_{i'})$, $t_i \in [\![s_{i'}]\!]_{\mathcal{T}}$, since node $n_i$ contains the sequent $[\![s_{i'}]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi'$.

**Lemma A.13.** *$s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ implies $[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi$ for any LTS $\mathcal{T}$. That is, if there is a successful tableaux for $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ in $\Sigma_{\mathcal{E}}$, then for any $\mathcal{T}$ there is a successful tableaux for $[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi$ in $\Sigma_{\mathcal{T}}$.*

*Proof.* Assume $A_{\mathcal{E}}$ is a successful tableau for sequent $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$, and assume $\mathcal{T}$ is an LTS. To establish the result it suffices, due to Corollary A.11, to show that each leaf of tableaux $A_{\mathcal{T}} = \pi_{\mathcal{T}}(A_{\mathcal{E}})$ matching a (successful) terminal of $A_{\mathcal{E}}$ is a successful terminal. Consider the successful terminal $m : r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$ in $A_{\mathcal{E}}$ and the matching node of $A_{\mathcal{T}}$, $n : [\![r]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi$. The proof that the latter is also a successful terminal is trivial when $\Psi = \mathrm{tt}$, when $\Psi = Z$ and $Z$ is free in the initial formula, when $Z$ identifies the formula $\nu Z.\Psi'$, or when $\Psi = [a]\,\Psi'$ and $\cup\partial^{\diamond}_{a}(r) = \emptyset$.

The only interesting case occurs when $\Psi = Z$ and $Z$ identifies $\mu Z.\Psi'$. In such a case, the condition of subset inclusion is trivially satisfied, so it remains to show that no infinite sequences of composable trails of the companion node $n'$ exist in $A_{\mathcal{T}}$, i.e. there is no $\kappa_{n'} = T_1 \circ T_2 \circ \dots$, such that for all $i \geq 1$, $T_i$ begins with a pair $(t_i, n')$, where $t_i \in [\![r]\!]_{\mathcal{T}}$.

Assume such an infinite sequence, $\kappa_{n'} = T_1 \circ T_2 \dots$, exists. Since the trails are composable, we also know that for all $i \geq 1$, $T_i$ ends with the pair $(t_{i'}, n)$ for some $t_{i'} \in [\![r]\!]_{\mathcal{T}}$. Then the corresponding sequence of the EMTS for terminal node $m$ and companion node $m'$, $w_{m'} = \beta(T_1) \circ \beta(T_2) \dots$ is an infinite sequence of $\mathcal{E}$-trails, and for all $i \geq 1$, $\beta(T_i)$ starts and ends with pairs $(r,m')$ and $(r,\, m)$ respectively. Because $m : r \vdash^{\mathcal{E}}_{\mathcal{V}} Z$ is a successful terminal, the corresponding infinite run $\alpha(w_{m'})$ is guaranteed to be in $W$. Then, by the definitions of simulation and denotation, the run $\alpha(\kappa_{n'})$ of $\mathcal{T}$ cannot be infinite, and therefore neither can $\kappa_{n'}$ be infinite. We thus reached a contradiction. □

**Theorem A.14** (Soundness)**.** *$s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ implies $s \models^{\mathcal{E}}_{\mathrm{V}} \Phi$.*

*Proof.* Follows directly from Lemma A.13, the soundness of Bradfield and Stirling's proof system (cf. [7, 36]), and Definition A.8. □

## Completeness

We base our completeness argument on the existence of a *universal* LTS $\mathcal{U}$, having the property that every LTS $\mathcal{T}$ is isomorphic to a transition–closed sub–structure of $\mathcal{U}$. Because of the shape of our disjunction proof rule, completeness can only be shown for formulas $\Phi$, all subformulas of which are *prime* (cf. [6]). A formula $\Psi$ is prime if whenever it logically implies a disjunction $\Psi_1 \vee \Psi_2$ then it also implies one of the disjuncts.

By the completeness proof of Bradfield and Stirling's proof system ([7], [36]), whenever $[\![s]\!]_{\mathcal{U}} \models^{\mathcal{U}}_{\mathbf{V}} \Phi$ we know that there is a space of canonical proofs of $[\![s]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi$, which differ in the choice functions that are made use of in the $\diamond$-rule applications. For the case when $\mathcal{E}$ is finite–state and all subformulas of $\Phi$ are prime, we show how to construct a proof $A_{\mathcal{U}}$ of $[\![s]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi$ that captures several of these canonical proofs using the branching provided by Cut-rule. This proof is constructed mutually with another proof $A^*_{\mathcal{U}}$ for the same goal that is built using "macro"s from $\pi_{\mathcal{U}}$ instead of single rules. Both proofs are built in the weaker version of Bradfield and Stirling's proof system, where repeat nodes are not necessarily terminals. In the final step of the Completeness proof, $A^*_{\mathcal{U}}$ is translated backward into a proof of $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ using the reverse function $\pi^{-1}_{\mathcal{U}}$.

The constructions of $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$ guide each other. At each step, the rule application in $A_{\mathcal{U}}$ determines the corresponding subtree (or "macro") application taken from the range of translation $\pi_{\mathcal{U}}$ in $A^*_{\mathcal{U}}$ except when the rule is Thin. In turn, $A^*_{\mathcal{U}}$ determines the number of Cut-rules that are to be applied after each $\Box$ and $\diamond$ rule in $A_{\mathcal{U}}$. As a result, each rule application in $A_{\mathcal{U}}$ is matched by the same rule application in $A^*_{\mathcal{U}}$ except for Thin. In order to describe this, we define in the construction process a $\gamma$ function which matches nodes of $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$. If the rule/macro to be used is $\diamond$, $A^*_{\mathcal{U}}$ additionally constraints the choice function used in $A_{\mathcal{U}}$ so that the extension of the former to the set mentioned in the matching node of $A_{\mathcal{U}}$ gives the latter. Finally, when a repeat node of $A_{\mathcal{U}}$ is to be taken as a terminal is also determined by whether the matching node is a terminal in $A^*_{\mathcal{U}}$.

$A_{\mathcal{U}}$ is similar to the canonical proofs from the completeness proof of $\Sigma_{\mathcal{T}}$
[7, 36] in the judicious application of the Thin rule, and in that the validity of the sequent is preserved at each application. Besides being applied to the goal as the first rule, Thin is applied in the rest of the tree only if the goal is of the form $S \ \vdash^{\mathcal{U}}_{\mathbf{V}} \ \sigma Z.\Psi$. The application of Thin should reduce the goal $S \ \vdash^{\mathcal{T}}_{\mathrm{V}} \ \Phi$ to the subgoal $||\Phi||^{\mathcal{T}}_{\mathbf{V}} \ \vdash^{\mathcal{T}}_{\mathrm{V}} \ \Phi$, where $||\Phi||^{\mathcal{T}}_{\mathbf{V}}$ is defined as the set of all states in $S_{\mathcal{T}}$ that satisfy $\Phi$ under valuation $\mathbf{V}$. $A_{\mathcal{U}}$ may include applications of a special version of the Cut-rule, while no Cut-rule applications occur in canonical proofs. Cut may be applied after $\Box$ and $\diamond$ rule in $A_{\mathcal{U}}$, in such a way that the set mentioned in each of the subgoals produced will be identical to the set mentioned in the original node. (Note that this application merely duplicates the current subgoal hence giving the possibility to combine several proof trees for the same subgoal in a single proof tree.)

**Theorem A.15** (Completeness)**.** *Let $\mathcal{E}$ be a finite–state EMTS, $s \in S_{\mathcal{E}}$, and let $\Phi$ have prime subformulas only. Then $s \models^{\mathcal{E}}_{\mathrm{V}} \Phi$ implies $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$.*

*Proof.* See [1]. □

## A.6 Conclusion

We propose extended modal transition systems as a means for representing the set of possible behaviours of an open system in the presence of not yet available components for which we are given behavioural specifications (assumptions) written in the modal $\mu$-calculus. The extentions to the standard notion of modal transition systems are needed to deal with disjunctive assumptions and with liveness assumptions. The latter are captured through a set of well-founded (that is, prohibited infinite) runs, a notion dual to the familiar notion of fairness constraints. We present a proof system for local model checking of EMTS properties written in the modal $\mu$-calculus. We define a translation of proofs in our proof system to proofs in a known sound and complete proof system for infinite state systems due to Bradfield and Stirling, and use this translation to show these properties for our proof system.

This paper presents work in progress towards state space representation and verification of open systems. Current work focuses on developing a formalism to model open systems, and on the problem of constructing a finite EMTS from such a specification, exploring both interactive techniques when dealing with the full logic and algorithmic techniques for certain fragments. The notion of an EMTS will be modified to include a suitable finite representation for the set of well-founded runs $W$, which in turn will make it possible to give an algorithm for verifying that an EMTS state satisfies a temporal property. The theory will be extended to deal with assumptions on data variables, in addition to assumptions on process variables. Finally, tool development will be followed by practical applications and verification case studies.

*Acknowledgement.* The authors are indebted to Christoph Sprenger for his valuable comments on an earlier draft.

# Appendix B

# Paper II

## State Space Representation for Verification of Open Systems

*Irem Aktug, Dilian Gurov*
KTH Computer Science and Communication
Stockholm, Sweden
{irem,dilian}@nada.kth.se

**Abstract**

When designing an open system, there might be no implementation available for certain components at verification time. For such systems, verification has to be based on assumptions on the underspecified components. When component assumptions are expressed in Hennessy-Milner logic (HML), the state space of open systems can be naturally represented with modal transition systems (MTS), a graphical specification language equiexpressive with HML. Having an explicit state space representation supports state space exploration based verification techniques. Besides, it enables proof reuse and facilitates visualization for the user guiding the verification process in interactive verification. As an intuitive representation of system behavior, it aids debugging when proof generation fails in automatic verification.

HML is not expressive enough to capture temporal assumptions. For this purpose, we extend MTSs to represent the state space of open systems where component assumptions are specified in modal $\mu$-calculus. We present a two-phase construction from process algebraic open system descriptions to such state space representations. The first phase deals with component assumptions, and is essentially a maximal model construction for the modal $\mu$-calculus. In the second phase, the models obtained are combined according to the structure of the open system to form the complete state space. The construction is sound and complete for systems with a single unknown component and sound for those without dynamic process creation. For establishing open system properties based on the representation, we present a proof system which is sound and complete for prime formulae.

## B.1 Introduction

In an *open system*, certain components can join the system after it has been put in operation. For example, applications can be loaded on a smart card after the card has been issued (see e.g. [SGH04]). Since the implementations of certain components are not yet available, the verification of the system has to be based on behavioural assumptions on such components. Security protocols can be verified in this manner, for instance by treating an unpredictable attacker as an unknown component of the system [31].

Modal transition systems (MTS) were introduced by Larsen as a graphical specification language [29]. Certain kinds of properties are easier to express graphically than in temporal logics. Each MTS specifies a set of processes as an interval determined by necessary and admissable transitions. MTSs are equiexpressive with Hennessy-Milner logic (HML), i.e. an HML formula can be characterized by an MTS and vice versa. As a result, MTSs provide a natural representation of the state space of open systems when assumptions on the behavior of the not-yet-available components are specified in HML. When the assumptions are temporal properties, however, MTSs are not expressive enough for this purpose. In [2], we extend MTSs to represent the state space of open systems when the component assumptions are written in the modal $\mu$-calculus [25]. This logic adds the expressive power of least and greatest fixed point recursion to HML. Besides the *must* (necessary) and *may* (admissable) transitions of MTS, our notion, *extended modal transition system* (EMTS) has sets of states (instead of single states) as targets to transitions - an extension which is needed for dealing with disjunctive assumptions, and well-foundedness constraints to handle least fixed point assumptions.

Having a way to capture the state space of an open system explicitly can be useful in various phases of the development of open systems. In *the modeling phase*, this formalism can be used as an alternative means of graphical specification of open system behavior. In *interactive verification*, an explicit state space representation facilitates visualization of the system behaviour, assisting the user in guiding the proof. This visualization facility is beneficial in *automatic verification* when the automatic proof construction fails and an understanding of the open system behaviour becomes necessary for debugging. Furthermore, computing the whole state space enables proof reuse when the same system is to be checked for several properties.

In this paper, we address the problem of constructing an explicit state space representation from an open system description and verifying open system properties based on this representation. In a process algebraic setting, the behaviour of an open system can be specified by an *open process term with assumptions* (OTA). An OTA consists of a process term equipped with a list of behavioral assumptions on the free variables of the term. We offer a two-phase construction that, under given restrictions, automatically extracts an EMTS from an OTA. The first phase in the construction corresponds to a maximal model construction for each component assumption. For the fixed point cases, a powerset construction is used that is similar to the one used in the Büchi automata constructions of [13] and [24]. In the second phase, the maximal models are composed according to the structure of the open system. The construction is *sound* (resp. *complete*) if the set of systems denoted by the OTA is a subset (resp. superset) of the denotation of the resulting EMTS. We show soundness of the construction for systems without dynamic process creation, and soundness and completeness for systems with a single unknown component. Finally, we present a proof system for showing open system properties based on EMTSs. The proof system is sound and complete for *prime* formulae, a prime formula being one that logically implies one of the disjuncts whenever it logically implies a disjunction. The relative simplicity of the proof system and its use is an indication of the adequateness of EMTSs for open system state space representation.

**Related Work.** In this strand of research, our work follows earlier work on using maximal model constructions for modular verification for various fragments of the $\mu$-calculus: for ACTL by Grumberg and Long [19], ACTL* by Kupferman and Vardi [26], and the fragment without least fixed points and diamond modalities by Sprenger et al [35]. In automata based approaches (see for instance [17, 24, 28]), various structures like alternating tree automata, Büchi and Rabin automata have been employed for capturing temporal properties. Although expressively powerful, we argue that these structures do not provide an intuitive representation of the state space for branching-time logics.

Proof system based methods have previously been suggested for the interactive verification of open systems [14, 15] where modal $\mu$-calculus is used to express the temporal assumptions on components as well as the desired property of the system. These interactive methods explore the state space implicitly as much as it is necessary for the particular verification task. In contrast to these methods, we separate the tasks of constructing a finite representation of the state space of an open system from the task of verifying its properties. This separation provides a state visualization facility to the user guiding the interactive proof, and offers greater possibilities for proof reuse.

**Organization.** The paper is organized as follows. In section B.2, we make the syntax of OTAs precise by a brief account of the logic used in behavioral assumptions and the process algebra used to define the process term. Section B.3 is a summary of important definitions related to the notion of EMTS. We present the translation from OTA to EMTS in Section B.4, and provide correctness results. In Section B.5, we give a proof system for showing open system properties of EMTSs. The last section presents conclusions and identifies directions for future work.

## B.2 Specifying Open Systems Behaviour

A system, the behaviour of which is parameterized on the behaviour of certain components, is conveniently represented as a pair $\Gamma \rhd E$, where $E$ is an open process-algebraic term, and $\Gamma$ is a list of assertions of the shape $X : \Phi$ where $X$ is a process variable free in $E$ and $\Phi$ is a closed formula in a process logic.

In the present study, we work with the class of Basic Parallel Processes (BPP)[9]. The terms of BPP are generated by:

$$E ::= \mathbf{0} \mid X \mid a.E \mid E + E \mid E \parallel E \mid \mathit{fix}\ X.E$$

where $X$ ranges over a set of process variables *ProcVar* and $a$ over a finite set of actions $A$. We assume that *ProcVar* is partitioned into assumption process variables *AssProcVar* used in assertions, and recursion process variables *RecProcVar* bound by *fix* . A term $E$ is called *linear* if every assumption process variable occurs in $E$ at most once. The operational semantics of closed process terms (called processes and ranged over by $t$) is standard, where the operator $\parallel$ signifies *merge composition.*

As a process logic for specifying behavioural assumptions of components, as well as for specifying system properties to be verified, we consider the modal $\mu$-calculus [25]. Its formulas are generated by:

$$\Phi ::= \mathrm{tt} \mid \mathrm{ff} \mid Z \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid [a]\,\Phi \mid \langle a \rangle\,\Phi \mid \nu Z.\Phi \mid \mu Z.\Phi$$

where $Z$ ranges over a set of propositional variables *PropVar*. The semantics of the $\mu$-calculus is standard and given in terms of the denotation $||\Phi||_{\mathrm{V}}^{\mathcal{T}} \subseteq S_{\mathcal{T}}$ where $\mathrm{V} : PropVar \to S_{\mathcal{T}}$ is a valuation that maps propositional variables to processes of some *labeled transition system* (LTS), which are ranged over by $\mathcal{T}$. As usual, we write $t \models_{\mathrm{V}}^{\mathcal{T}} \Phi$ whenever $t \in ||\Phi||_{\mathrm{V}}^{\mathcal{T}}$. In the sequel, we omit the subscript V when $\Phi$ is a closed formula.

We say that an OTA $\Gamma \rhd E$ is *guarded* when the term $E$ and all modal $\mu$-calculus formula $\Phi$ in $\Gamma$ are guarded. Similarly, we say an OTA is linear when the term it contains is linear.

The behaviours specified by an open term with assumptions is given with respect to a labeled transition system $\mathcal{T}$ that is closed under the transition rules and is closed under substitution of processes for assumption process variables in subterms of the OTA. The states of LTS correspond to processes in our process algebra. The denotation of an OTA is then the set of all processes obtained by substituting each assumption process variable in the term by a process from $\mathcal{T}$ satisfying the respective assumptions.

**Definition B.1** (OTA Denotation)**.** Let $\Gamma \rhd E$ be an OTA, $\mathcal{T}$ be an LTS, and $\rho_R$ : RecProcVar $\to S_{\mathcal{T}}$ be a recursion environment. The *denotation* of $\Gamma \rhd E$ relative to $\mathcal{T}$ and $\rho_R$ is defined as:

$$[\![\Gamma \rhd E]\!]_{\rho_R} \triangleq \{E\rho_R\rho_A \mid \forall (X : \Phi) \in \Gamma.\ \rho_A(X) \models^{\mathcal{T}} \Phi\}$$

where $\rho_A$ : AssProcVar $\to S_{\mathcal{T}}$ ranges over assumption environments.

**Example.** Consider an operating system in the form of a concurrent server that spawns off *Handler* processes each time it receives a request. These processes run system calls for handling the given requests to produce a result (modeled by the action $\overline{out}$). *Handler* is defined as $Handler \overset{def}{=} In \parallel \overline{out}.\mathbf{0}$ where $In \overset{def}{=} in.In$. Although it is possible to communicate with request handlers through the attached channel (modeled by the action *in*), they do not react to further input. A property one would like to prove of such a server is that it stabilizes whenever it stops receiving new requests. Eventual stabilization can be formalized in the modal $\mu$-calculus as $\mathbf{stab} \triangleq \nu X.\mu Y. [in]\, X \wedge \left[\overline{out}\right] Y$. We can reduce this verification task to proving that the open system modeled by the OTA

$$X : \mathbf{stab} \rhd X \parallel Handler$$

which consists of $Handler$ and any stabilizing process $X$, eventually stabilizes.

## B.3 Extended Modal Transition Systems

In [2], we proposed Extended Modal Transition Systems (EMTS) as an explicit state space representation for open systems with temporal assumptions, with an extensional representation for the well-foundedness constraints. In this section, we summarize the main definitions, and propose a concrete representation of well-foundedness constraints. The notion of EMTS is based on Larsen's Modal Transition Systems [29]. Kripke Modal Transition Systems (KMTS) have been first introduced by Huth et. al. [23], and later refined by Grumberg and Shoham [20] for representing state space abstractions in an abstraction refinement framework. EMTS is similar to KMTS with the addition of fairness constraints.

In addition to may and must transitions for dealing with modalities, EMTSs include sets of states (instead of single states) as targets to transitions to capture disjunctive assumptions, and a set of prohibited infinite runs defined through a coloring function to represent termination assumptions.

**Definition B.2** (EMTS)**.** An *extended modal transition system* is a structure

$$\mathcal{E} = (S_{\mathcal{E}}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, c)$$

where (i) $S_{\mathcal{E}}$ is a set of *abstract states*, (ii) $A$ is a set of *actions*, (iii) $\longrightarrow^{\Diamond}_{\mathcal{E}}$, $\longrightarrow^{\Box}_{\mathcal{E}}$ $\subseteq S_{\mathcal{E}} \times A \times 2^{S_{\mathcal{E}}}$ are *may* and *must transition relations*, and (iv) $c : S_{\mathcal{E}} \rightarrow \mathbb{N}^k$ is a *coloring function* for some $k \in \mathbb{N}$.

May transitions of an EMTS show possible behaviours of the closed systems represented, while must transitions specify behaviour shared by all these closed systems. A *run* (or may–run) of $\mathcal{E}$ is a possibly infinite sequence of transitions $\rho_{\mathcal{E}} = s_0 \xrightarrow{a_0}_{\mathcal{E}} s_1 \xrightarrow{a_1}_{\mathcal{E}} s_2 \xrightarrow{a_2}_{\mathcal{E}} \ldots$ where for every $i \geq 0$, $s_i \overset{a_i}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}} S$ for some $S$ such that $s_{i+1} \in S$. Must–runs are defined similarly. We distinguish between two kinds of *a-derivatives* of a state $s$: $\partial^{\Diamond}_a(s) \triangleq \{S \mid s \overset{a}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}} S\}$ and $\partial^{\Box}_a(s) \triangleq \{S \mid s \overset{a}{\longrightarrow}{}^{\Box}_{\mathcal{E}} S\}$.

The coloring function $c$ specifies a set $W_{\mathcal{E}}$ of prohibited infinite runs by means of a *parity acceptance condition* (cf. [32, 17]). The function $c$ is extended to infinite runs so that $c(\rho_{\mathcal{E}}) = (c(s_0)(1) \cdot c(s_1)(1) \ldots, \ldots, c(s_0)(k) \cdot c(s_1)(k) \ldots)$ is a $k$-tuple of infinite words where $c(s)(j)$ denotes the $j^{th}$ component of $c(s)$. Let $\mathit{inf}(c(\rho_{\mathcal{E}})(i))$ denote the set of infinitely occurring colors in the $i^{th}$ word of this tuple. Then the run $\rho_{\mathcal{E}}$ is prohibited, $\rho_{\mathcal{E}} \in W_{\mathcal{E}}$, if and only if $\mathit{max}\,(\mathit{inf}\,(c(\rho_{\mathcal{E}})(i)))$ is odd for some $1 \leq i \leq k$, i.e. the greatest number that occurs infinitely often in one of these $k$ infinite words is odd.

Next, we define a simulation relation between the states of an EMTS as a form of mixed fair simulation (cf. e.g. [19, 8]).

**Definition B.3** (Simulation)**.** $R \subseteq S_{\mathcal{E}} \times S_{\mathcal{E}}$ is a *simulation relation* between the states of $\mathcal{E}$ if whenever $s_1 R s_2$ and $a \in A$:

1. if $s_1 \overset{a}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}} S_1$, then there is a $S_2$ such that $s_2 \overset{a}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}} S_2$ and for each $s_1' \in S_1$, there exists a $s_2' \in S_2$ such that $s_1' R s_2'$;

2. if $s_2 \overset{a}{\longrightarrow}{}^{\Box}_{\mathcal{E}} S_2$, then there is a $S_1$ such that $s_1 \overset{a}{\longrightarrow}{}^{\Box}_{\mathcal{E}} S_1$ and for each $s_1' \in S_1$, there exists a $s_2' \in S_2$ such that $s_1' R s_2'$;

3. if the run $\rho_{s_2} = s_2 \xrightarrow{a_1}_{\mathcal{E}} s_2^1 \xrightarrow{a_2}_{\mathcal{E}} s_2^2 \xrightarrow{a_3}_{\mathcal{E}} \ldots$ is in $W_{\mathcal{E}}$ then every infinite run $\rho_{s_1} = s_1 \xrightarrow{a_1}_{\mathcal{E}} s_1^1 \xrightarrow{a_2}_{\mathcal{E}} s_1^2 \xrightarrow{a_3}_{\mathcal{E}} \ldots$ such that $s_1^i R s_2^i$ for all $i \geq 1$ is also in $W_{\mathcal{E}}$.

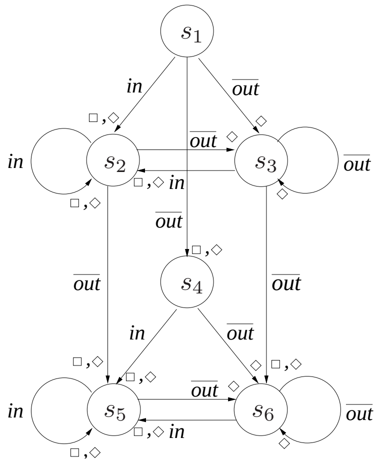
We say that abstract state $s_2$ *simulates* abstract state $s_1$, denoted $s_1 \preceq s_2$, if there is a simulation relation $R$ such that $s_1 R s_2$. Simulation can be generalized to two different EMTSs $\mathcal{E}_1$ and $\mathcal{E}_2$ in the natural way.

Labeled transition systems can be viewed as a special kind of EMTS, where: $\longrightarrow^{\Box}_{\mathcal{E}} = \longrightarrow^{\Diamond}_{\mathcal{E}}$, the target sets of the transition relation are singleton sets of states, and the set of prohibited runs $W$ is empty. We give the meaning of an abstract state relative to a given LTS, as the set of concrete LTS states simulated by the abstract state.

**Definition B.4** (Denotation)**.** Let $\mathcal{E}$ be an EMTS, and let $\mathcal{T}$ be an LTS. The *denotation* of abstract state $s \in S_{\mathcal{E}}$ is the set $[\![s]\!]_{\mathcal{T}} \triangleq \{t \in S_{\mathcal{T}} \mid t \preceq s\}$. This notion is lifted to sets of abstract states $S' \subseteq S_{\mathcal{E}}$ in the natural way: $[\![S']\!]_{\mathcal{T}} \triangleq \bigcup\{[\![s]\!]_{\mathcal{T}} \mid s \in S'\}$.

In the rest of the paper, we shall assume that EMTSs obey the following *consistency* restrictions: $\longrightarrow^{\Box}_{\mathcal{E}} \subseteq \longrightarrow^{\Diamond}_{\mathcal{E}}$, $s \overset{a}{\longrightarrow}{}^{\Box}_{\mathcal{E}} S$ implies $S$ is non-empty, and $W$ does not contain runs corresponding to infinite must–runs of the EMTS.

In section B.5, we present a proof system for proving properties of abstract states. For this purpose, we define when an abstract state $s$ satisfies a modal $\mu$-calculus formula $\Phi$. The global nature of the set $W$ in EMTSs makes it cumbersome to define the denotation of a fixed point formula compositionally as a set of abstract states. We therefore give an indirect definition of satisfaction, by means of the denotation $[\![s]\!]_{\mathcal{T}}$ of a state $s$.



$$c(s_1) = c(s_4) = 0$$
$$c(s_3) = c(s_6) = 1$$
$$c(s_2) = c(s_5) = 2$$

Figure B.1: EMTS for $X : \textbf{\textit{stab}} \rhd X \parallel \mathit{Handler}$

**Definition B.5** (Satisfaction)**.** Let $\mathcal{E}$ be an EMTS, $s \in S_{\mathcal{E}}$ be an abstract state of $\mathcal{E}$ and $\Phi$ be a modal $\mu$-calculus property. Then $s$ satisfies $\Phi$ under valuation $\mathcal{V} : \mathrm{PropVar} \to 2^{S_{\mathcal{E}}}$, denoted $s \models^{\mathcal{E}}_{\mathcal{V}} \Phi$, if and only if for any LTS $\mathcal{T}$ $[\![s]\!]_{\mathcal{T}} \models^{\mathcal{T}}_{\mathrm{V}} \Phi$ where valuation $\mathrm{V} : \mathrm{PropVar} \to 2^{S_{\mathcal{T}}}$ is induced by $\mathcal{V}$ as $\mathrm{V}(Z) \triangleq \bigcup\{[\![s]\!]_{\mathcal{T}} \mid s \in \mathcal{V}(Z)\}$.

**Example.** The state space of the open system introduced in the previous section is captured by the EMTS in Figure B.1. For any labeled transition system $\mathcal{T}$, the processes simulated by the state $s_1$ are those denoted by the open term $X : \textbf{stab} \rhd X \parallel \mathit{Handler}$. The EMTS consists of six abstract states, each state denoting the set of processes which it simulates. For instance, states $s_5$ and $s_6$ in the example denote all processes which can engage in arbitrary interleavings of *in* and $\overline{out}$ actions, but so that *in* has to be enabled throughout while $\overline{out}$ has not. Infinite runs stabilizing on $\overline{out}$ actions are prohibited by the coloring of $s_3$ and $s_6$.

## B.4 From OTA to EMTS

In this section, we address the problem of providing an explicit state space representation for a given open term $\Gamma \rhd E$, by means of an EMTS $\mathcal{E}$. While it is tempting to define $\longrightarrow^{\Diamond}_{\mathcal{E}}$ and $\longrightarrow^{\square}_{\mathcal{E}}$ through transition rules, the global nature of the well-foundedness constraints suggests that a direct construction would be more convenient for automatic construction. We propose a two-phase construction $\varepsilon$ that translates an open term $\Gamma \rhd E$ to an EMTS, denoted $\varepsilon(\Gamma \rhd E)$. In the first phase, an EMTS is constructed for each underspecified component. This part is essentially a maximal model construction as developed by Grumberg and Long for ACTL [19], extended to ACTL* by Kupferman and Vardi [26], and applied by Sprenger et al to the fragment of the modal $\mu$-calculus without least fixed points and diamond modalities [35]. For the construction of the fixed point cases, we adapt a powerset construction used earlier to convert fragments of the modal $\mu$-calculus to Büchi automata which was introduced by Dam [13] for linear time $\mu$-calculus and extended by Kaivola [24] to the $\Pi_2$ fragment. The second phase consists of combining the EMTSs produced in the first step according to the structure of the term $E$. We then show the correctness of the construction by relating the set of states simulated by the constructed EMTS to the denotation of the given OTA.

## Maximal Model Construction

We define the function $\varepsilon$ which maps modal $\mu$-calculus formulas to triples of the shape $(\mathcal{E},\, S,\, \lambda)$, where $\mathcal{E} = (S_{\mathcal{E}}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, c)$ is an *EMTS*, $S \subseteq S_{\mathcal{E}}$ is a set of *start states* of $\mathcal{E}$, and $\lambda : S_{\mathcal{E}} \to 2^{PropVar}$ is a *labeling function*.

The function is defined inductively on the structure of $\Phi$ as shown in Figure B.2. The meaning of open formulae that arises in intermediate steps are given by the by the valuation which assigns the whole set of processes $S_{\mathcal{T}}$ to each propositional variable. Essentially, the particular valuation used does not play a role in the final EMTS, since the properties used as assumptions of an OTA are closed.

In the definition, let $\varepsilon(\Phi_1)$ be $((S_{\mathcal{E}_1},\, A,\, \longrightarrow^{\Diamond}_{\mathcal{E}_1},\, \longrightarrow^{\Box}_{\mathcal{E}_1},\, c_1),\, S_1,\, \lambda_1)$ and $\varepsilon(\Phi_2)$ be $((S_{\mathcal{E}_2},\, A,\, \longrightarrow^{\Diamond}_{\mathcal{E}_2},\, \longrightarrow^{\Box}_{\mathcal{E}_2},\, c_2),\, S_2,\, \lambda_2)$ where $S_{\mathcal{E}_1}$ and $S_{\mathcal{E}_2}$ are disjoint sets. The new state $s_{new}$ is not in $S_{\mathcal{E}_1}$ and $a$ and $a'$ are actions in $A$. The coloring functions $c_1 : S_{\mathcal{E}_1} \to \mathbb{N}^{k_1}$ and $c_2 : S_{\mathcal{E}_2} \to \mathbb{N}^{k_2}$ color the states of $\mathcal{E}_1$ and $\mathcal{E}_2$ with integer tuples of length $k_1$ and $k_2$ respectively.

For a set $S$, $S\mid_{\Box}$ denotes the largest transition-closed set contained in $S$ such that there is no element $s \in S\mid_{\Box}$ with the empty set as the target to a must transition, that is, there is no $s$ such that for some $a \in A$, $s \overset{a}{\longrightarrow}^{\Box}_{\mathcal{E}} \emptyset$ and each state $s$ is reachable from some start state.

In what follows, we explain the various cases of the construction. The EMTS for formula tt consists of the single state $s_{\text{tt}}$ with may transitions to itself for every action, while the EMTS for ff is the empty EMTS. The EMTS for a propositional variable consists of a single state with may transitions to $s_{\text{tt}}$ for each action.

The states of the EMTS for the conjunction of two formulas is the cross product of the states of the EMTSs constructed for each conjunct, excluding pairs with incompatible capabilities. The color of a state of $\varepsilon(\Phi_1 \wedge \Phi_2)$ is the concatenation of the colors of the paired states. In the case of disjunction, the set of start states of $\varepsilon(\Phi_1 \vee \Phi_2)$ is the union of the start states of $\varepsilon(\Phi_1)$ and $\varepsilon(\Phi_2)$ which reflects the union of their denotation. The color of a state is given by padding with 0's from either the left or right.

For the modal cases, a new state $s_{new}$ is set as the start state. The EMTS for $\varepsilon([a]\,\Phi)$ has a single may transition for $a$, which is to the set of initial states of $\varepsilon(\Phi)$. This is to ensure all simulated processes satisfy $\Phi$ after engaging in an $a$.

- $\varepsilon(\mathrm{tt}) \triangleq ((\{s_{\mathrm{tt}}\},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \emptyset,\ \{s_{\mathrm{tt}} \mapsto 0\}),\ \{s_{\mathrm{tt}}\},\ \{s_{\mathrm{tt}} \mapsto \emptyset\})$
  where $s_{\mathrm{tt}} \xrightarrow{a}^{\Diamond}_{\mathcal{E}} \{s_{\mathrm{tt}}\}$ for all $a \in A$.

- $\varepsilon(\mathrm{ff}) \triangleq ((\emptyset,\ A,\ \emptyset,\ \emptyset,\ \emptyset),\ \emptyset,\ \emptyset)$

- $\varepsilon(Z) \triangleq ((\{s_{new}, s_{\mathrm{tt}}\},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \emptyset,\ \{s_{\mathrm{tt}} \mapsto 0, s_{new} \mapsto 0\}),\ \{s_{new}\},\ \{s_{new} \mapsto \{Z\}, s_{\mathrm{tt}} \mapsto \emptyset\})$
  where $s_{new} \xrightarrow{a}^{\Diamond}_{\mathcal{E}} \{s_{\mathrm{tt}}\}$ and $s_{\mathrm{tt}} \xrightarrow{a}^{\Diamond}_{\mathcal{E}} \{s_{\mathrm{tt}}\}$ for all $a \in A$.

- $\varepsilon(\Phi_1 \wedge \Phi_2) \triangleq (((S_{\mathcal{E}_1} \times S_{\mathcal{E}_2})|_{\square},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\square}_{\mathcal{E}},\ W),\ (S_{\mathcal{E}_1} \times S_{\mathcal{E}_2})|_{\square} \cap (S_1 \times S_2),\ \lambda)$ where

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \{(s,r) \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S' \times \cup\partial^{\Diamond}_a(r) \mid s \xrightarrow{a}^{\square}_{\mathcal{E}_1} S'\} \\
& \cup & \{(s,r) \xrightarrow{a}^{\Diamond}_{\mathcal{E}} \cup\partial^{\Diamond}_a(s) \times R' \mid r \xrightarrow{a}^{\square}_{\mathcal{E}_2} R'\} \\
& \cup & \{(s,r) \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S' \times R' \mid s \xrightarrow{a}^{\Diamond}_{\mathcal{E}_1} S' \wedge r \xrightarrow{a}^{\Diamond}_{\mathcal{E}_2} R' \wedge S' \notin \partial^{\square}_a(s) \wedge R' \notin \partial^{\square}_a(r)\} \\
\longrightarrow^{\square}_{\mathcal{E}} & \triangleq & \{(s,r) \xrightarrow{a}^{\square}_{\mathcal{E}} (S' \times \cup\partial^{\Diamond}_a(r)) \mid s \xrightarrow{a}^{\square}_{\mathcal{E}_1} S'\} \\
& \cup & \{(s,r) \xrightarrow{a}^{\square}_{\mathcal{E}} (\cup\partial^{\Diamond}_a(s) \times R') \mid r \xrightarrow{a}^{\square}_{\mathcal{E}_2} R'\} \\
c & \triangleq & \{(s,r) \mapsto c_1(s) \cdot c_2(r) \mid s \in S_{\mathcal{E}_1} \wedge r \in S_{\mathcal{E}_2}\} \\
\lambda & \triangleq & \{(s,r) \mapsto \lambda_1(s) \cup \lambda_2(r) \mid s \in S_{\mathcal{E}_1} \wedge r \in S_{\mathcal{E}_2}\}
\end{array}$$

- $\varepsilon(\Phi_1 \vee \Phi_2) \triangleq ((S_{\mathcal{E}_1} \cup S_{\mathcal{E}_2},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\square}_{\mathcal{E}},\ c),\ S_1 \cup S_2,\ \lambda_1 \cup \lambda_2)$ with:

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \longrightarrow^{\Diamond}_{\mathcal{E}_1} \cup \longrightarrow^{\Diamond}_{\mathcal{E}_2} \\
\longrightarrow^{\square}_{\mathcal{E}} & \triangleq & \longrightarrow^{\square}_{\mathcal{E}_1} \cup \longrightarrow^{\square}_{\mathcal{E}_2} \\
c & \triangleq & \{s \mapsto c_1(s) \cdot 0^{k_2} \mid s \in S_{\mathcal{E}_1}\} \cup \{s \mapsto 0^{k_1} \cdot c_2(s) \mid s \in S_{\mathcal{E}_2}\}
\end{array}$$

- $\varepsilon([a]\,\Phi_1) \triangleq ((S_{\mathcal{E}_1} \cup \{s_{new}, s_{\mathrm{tt}}\},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\square}_{\mathcal{E}_1},\ c),\ \{s_{new}\},\ \lambda)$ with:

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \longrightarrow^{\Diamond}_{\mathcal{E}_1} \cup \{s_{\mathrm{tt}} \xrightarrow{a'}^{\Diamond}_{\mathcal{E}} \{s_{\mathrm{tt}}\} \mid a' \in A\} \cup \{s_{new} \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S_1\} \\
& \cup & \{s_{new} \xrightarrow{a'}^{\Diamond}_{\mathcal{E}} \{s_{\mathrm{tt}}\} \mid a' \neq a \wedge a' \in A\} \\
c & \triangleq & c_1 \cup \{s_{new} \mapsto 0^{k_1}\} \cup \{s_{\mathrm{tt}} \mapsto 0^{k_1}\} \\
\lambda & \triangleq & \lambda_1 \cup \{s_{new} \mapsto \emptyset\} \cup \{s_{\mathrm{tt}} \mapsto \emptyset\}
\end{array}$$

- $\varepsilon(\langle a\rangle\,\Phi_1) \triangleq \varepsilon(\mathrm{ff})$ if $S_1 = \emptyset$
  $\varepsilon(\langle a\rangle\,\Phi_1) \triangleq ((S_{\mathcal{E}_1} \cup \{s_{new}, s_{\mathrm{tt}}\},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\square}_{\mathcal{E}},\ c),\ \{s_{new}\},\ \lambda)$ otherwise, with:

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \longrightarrow^{\Diamond}_{\mathcal{E}_1} \cup \{s_{new} \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S_1\} \cup \{s_{new} \xrightarrow{a'}^{\Diamond}_{\mathcal{E}} \{s_{\mathrm{tt}}\} \mid a' \in A\} \cup \{s_{\mathrm{tt}} \xrightarrow{a'}^{\Diamond}_{\mathcal{E}} \{s_{\mathrm{tt}}\} \mid a' \in A\} \\
\longrightarrow^{\square}_{\mathcal{E}} & \triangleq & \longrightarrow^{\square}_{\mathcal{E}_1} \cup \{s_{new} \xrightarrow{a}^{\square}_{\mathcal{E}} S_1\} \\
c & \triangleq & c_1 \cup \{s_{new} \mapsto 0^{k_1}\} \cup \{s_{\mathrm{tt}} \mapsto 0^{k_1}\} \\
\lambda & \triangleq & \lambda_1 \cup \{s_{new} \mapsto \emptyset\} \cup \{s_{\mathrm{tt}} \mapsto \emptyset\}
\end{array}$$

- $\varepsilon(\sigma Z.\Phi_1)\ ((2^{S_{\mathcal{E}_1}}|_{\square}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\square}_{\mathcal{E}}, c_\sigma), 2^{S_{\mathcal{E}_1}}|_{\square} \cap \{\{s\} \mid s \in S_1\}, \lambda)$ where $\sigma \in \{\nu, \mu\}$ with:

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \{\{s_1, \ldots, s_n\} \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S \mid \exists i.\exists S'_i.s_i \xrightarrow{a}^{\square}_{\mathcal{E}_1} S'_i \wedge \\
& & S = \partial_{\mathcal{P}}((\cup\partial^{\Diamond}_a(s_1), \ldots, S'_i, \ldots, \cup\partial^{\Diamond}_a(s_n)), S_1, \lambda_1, Z)\} \\
& \cup & \{\{s_1, \ldots, s_n\} \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S \mid \forall j.\exists S'_j.s_j \xrightarrow{a}^{\Diamond}_{\mathcal{E}_1} S'_j \wedge S'_j \notin \partial^{\square}_a(s_j) \wedge \\
& & S = \partial_{\mathcal{P}}((S'_1, \ldots, S'_n), S_1, \lambda_1, Z)\} \\
\longrightarrow^{\square}_{\mathcal{E}} & \triangleq \{ & \{s_1, \ldots, s_n\} \xrightarrow{a}^{\square}_{\mathcal{E}} S \mid \exists i.\exists S'_i.s_i \xrightarrow{a}^{\square}_{\mathcal{E}_1} S'_i \wedge \\
& & S = \partial_{\mathcal{P}}((\cup\partial^{\Diamond}_a(s_1), \ldots, S'_i, \ldots, \cup\partial^{\Diamond}_a(s_n)), S_1, \lambda_1, Z)\}
\end{array}$$

$$c_\nu(\{s_1, \ldots, s_n\})(j) \triangleq \begin{cases} \underset{1 \le i \le n}{maxodd}(c_1(s_i)(j)) & \text{if } \forall i.Z \notin \lambda_1(s_i) \\ \overset{even}{\underset{s \in S_{\mathcal{E}_1}}{\prod}} c_1(s)(j) & \text{if } \exists i.Z \in \lambda_1(s_i) \end{cases}$$

$$c_\mu(\{s_1, \ldots, s_n\})(j) \triangleq \begin{cases} \underset{1 \le i \le n}{maxodd}(c_1(s_i)(j)) & \text{if } \forall i.Z \notin \lambda_1(s_i) \\ \overset{odd}{\underset{s \in S_{\mathcal{E}_1}}{\prod}} c_1(s)(j) & \text{if } \exists i.Z \in \lambda_1(s_i) \end{cases}$$

$$\lambda \triangleq \{\{s_1, \ldots, s_n\} \mapsto \bigcup_{1 \le i \le n} \lambda_1(s_i) - \{Z\} \mid \{s_1, \ldots, s_n\} \in 2^{S_{\mathcal{E}_1}}\}$$

Figure B.2: Maximal Model Construction

Additionally, there is a may transition to $s_{\mathrm{tt}}$ for all other actions. The EMTS for $\varepsilon(\langle a\rangle\,\Phi)$ includes a must transition for $a$ from this start state to the start states of $\varepsilon(\Phi)$, along with may transitions for all actions to $s_{\mathrm{tt}}$ forcing the simulated processes to have an $a$ transition to some process satisfying $\Phi$ and allowing any other transitions besides.

The construction for fixed point formulae is a powerset construction, i.e. which is similar to the constructions given in [13] and [24] for the purpose of constructing Büchi Automata for linear time and the alternation-depth class $\Pi_2$ fragments of the $\mu$-calculus, respectively. the states of $\varepsilon(\sigma Z.\Phi)$ consist of sets of states of $\varepsilon(\Phi)$ and its start states are singletons containing some start state of $\varepsilon(\Phi)$. An invariant of the maximal model construction is that start states do not have incoming transitions. (The case for $\varepsilon(\mathrm{tt})$ is the only exception and can be easily adapted to satisfy the invariant.) For a transition of state $q = \{s_1, \ldots, s_n\}$ of $\varepsilon(\sigma Z.\Phi)$, each state $s_i$ has a transition in $\varepsilon(\Phi)$. A member state of the target of this transition, then, contains a derivative for each $s_i$. A member of the target state additionally contains an initial state of $\varepsilon(\Phi)$ if one of the derivatives included is labeled by $Z$. The definition of Figure B.2 makes use of the target set function $\partial_{\mathcal{P}}$ defined below.

**Definition B.6** (Target Set Function $\partial_{\mathcal{P}}$)**.** Let $\Phi$ be a modal $\mu$-calculus formula, $\sigma$ be either $\mu$ or $\nu$, $\varepsilon(\Phi)$ be $(\mathcal{E}_1,S,\lambda)$ where $\mathcal{E}_1 = (S_{\mathcal{E}_1}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, c)$ is an EMTS, $S \subseteq S_{\mathcal{E}_1}$ is a set of start states, $\lambda : S_{\mathcal{E}_1} \rightarrow 2^{PropVar}$ is a function that maps states of $\mathcal{E}$ to propositional variables, $c : S_{\mathcal{E}} \rightarrow N^k$ is a coloring function that maps states of $\mathcal{E}$ to $k$-tuples, and let $Z \in PropVar$ be a propositional variable. Given a tuple consisting of a target set for each element of a state of $\varepsilon(\sigma Z.\Phi)$, the function $\partial_{\mathcal{P}} : (2^{S_{\mathcal{E}_1}} \times \ldots \times 2^{S_{\mathcal{E}_1}}) \times 2^{S_{\mathcal{E}_1}} \times (S_{\mathcal{E}_1} \rightarrow 2^{PropVar}) \times PropVar \rightarrow 2^{2^{S_{\mathcal{E}_1}}}$ defines the target set of a transition of $\varepsilon(\sigma Z.\Phi)$ for this state as follows:

$$\begin{aligned}\partial_{\mathcal{P}}((S_1,\ldots,S_n),S,\lambda,Z) \;\triangleq\;& \{\{s_1,\ldots,s_n\} \mid \forall i.s_i \in S_i \wedge\ \nexists j.Z \in \lambda(s_j)\} \cup \\ & \{\{s_1,\ldots,s_n,s_0\} \mid \forall i.s_i \in S_i \wedge \\ & \qquad \exists j.Z \in \lambda(s_j) \wedge s_0 \in S\}\end{aligned}$$

Each component of the color of state $q$ is determined by comparing the corresponding entries of the member states $s_i$. When, for at least one $s_i$, this entry is odd, the greatest of the corresponding odd entries is selected as the entry of $q$, otherwise the maximum entry is selected for the same purpose. In Figure B.2, the function *maxodd* selects the greater of two numbers if both of them are odd or both of them are even, and the odd one otherwise. The color of $q$ is further updated if it contains a state $s_i$ labeled by $Z$. When $Z$ identifies a greatest fixed point formula, each entry of the constructed tuple is defined to be the least even upper bound of the integers used in this entry of $\varepsilon(\Phi)$. Whereas, when $Z$ identifies a least fixed point formula, the least odd upper bound of the integers is the entry for the color of $q$. In Figure B.2, least even and least odd upper bounds are denoted by the operators $\overset{even}{\sqcap}$ and $\overset{odd}{\sqcap}$, respectively.

- $\varepsilon(\Gamma \triangleright \mathbf{0}) \triangleq ((\{s_{new}\},\ A,\ \emptyset,\ \emptyset,\ \{s_{new} \mapsto 0\}),\ \{s_{new}\},\ \{s_{new} \mapsto \emptyset\})$

- $\varepsilon(\Gamma \triangleright X) \triangleq \varepsilon(\Phi)$ if $X \in AssProcVar$
  where $\Phi = \bigwedge_{X:\Psi\,\in\,\Gamma} \Psi$ ( defaults to tt when $\Gamma$ contains no assumption on $X$ ).

- $\varepsilon(\Gamma \triangleright X) \triangleq ((\{s_{new}\},\ A,\ \emptyset,\ \emptyset,\ \{s_{new} \mapsto 0\}),\ \{s_{new}\},\ \{s_{new} \mapsto \{X\}\})$ if $X \in RecProcVar$

- $\varepsilon(\Gamma \triangleright a.E_1) \triangleq ((S_{\mathcal{E}_1} \cup \{s_{new}\},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\Box}_{\mathcal{E}},\ c),\ \{s_{new}\},\ \lambda_1 \cup \{s_{new} \mapsto \emptyset\})$ with:

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \longrightarrow^{\Diamond}_{\mathcal{E}_1} \cup \{s_{new} \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} S_1\} \\
\longrightarrow^{\Box}_{\mathcal{E}} & \triangleq & \longrightarrow^{\Box}_{\mathcal{E}_1} \cup \{s_{new} \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S_1\} \\
c & \triangleq & c_1 \cup \{s_{new} \mapsto 0^{k_1}\}
\end{array}$$

- $\varepsilon(\Gamma \triangleright E_1 + E_2) \triangleq ((S_{\mathcal{E}_1} \cup S_{\mathcal{E}_2} \cup (S_1 \times S_2),\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\Box}_{\mathcal{E}},\ c),\ S_1 \times S_2,\ \lambda)$

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \longrightarrow^{\Diamond}_{\mathcal{E}_1} \cup \longrightarrow^{\Diamond}_{\mathcal{E}_2} \cup \{(s,r) \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} S' \mid s \in S_1 \wedge r \in S_2 \wedge (s \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}_1} S' \vee r \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}_2} S')\} \\
\longrightarrow^{\Box}_{\mathcal{E}} & \triangleq & \longrightarrow^{\Box}_{\mathcal{E}_1} \cup \longrightarrow^{\Box}_{\mathcal{E}_2} \cup \{(s,r) \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S' \mid s \in S_1 \wedge r \in S_2 \wedge (s \xrightarrow{a}{}^{\Box}_{\mathcal{E}_1} S' \vee r \xrightarrow{a}{}^{\Box}_{\mathcal{E}_2} S')\} \\
c & \triangleq & \{s \mapsto c_1(s) \cdot 0^{k_2} \mid s \in S_{\mathcal{E}_1}\} \cup \{r \mapsto 0^{k_1} \cdot c_2(r) \mid r \in S_{\mathcal{E}_2}\} \\
 & \cup & \{(s,r) \mapsto c_1(s) \cdot c_2(r) \mid (s,r) \in S_1 \times S_2\} \\
\lambda & \triangleq & \lambda_1 \cup \lambda_2 \cup \{(s,r) \mapsto \lambda_1(s) \cup \lambda_2(r) \mid s \in S_1 \wedge r \in S_2\}
\end{array}$$

- $\varepsilon(\Gamma \triangleright \mathit{fix}\ X.E_1) \triangleq ((S_{\mathcal{E}_1},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\Box}_{\mathcal{E}},\ c_1),\ S_1,\ \lambda)$ with:

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \{s \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} S \mid (s \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}_1} S) \vee \\
 & & (\exists s_1 \in S_1 . s_1 \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}_1} S \wedge\ X \in \lambda_1(s) \wedge s \text{ is reachable from } s_1)\} \\
\longrightarrow^{\Box}_{\mathcal{E}} & \triangleq & \{s \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S \mid (s \xrightarrow{a}{}^{\Box}_{\mathcal{E}_1} S) \vee \\
 & & (\exists s_1 \in S_1 . s_1 \xrightarrow{a}{}^{\Box}_{\mathcal{E}_1} S \wedge\ X \in \lambda_1(s) \wedge s \text{ is reachable from } s_1)\} \\
\lambda & \triangleq & \{s \mapsto (\lambda_1(s) - \{X\}) \mid s \in S_{\mathcal{E}_1}\}
\end{array}$$

- $\varepsilon(\Gamma \triangleright E_1 \parallel E_2) \triangleq ((S_{\mathcal{E}_1} \times S_{\mathcal{E}_2} \times \{1,2\},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}},\ \longrightarrow^{\Box}_{\mathcal{E}},\ c),\ S_1 \times S_2 \times \{1,2\},\ \lambda)$

$$\begin{array}{rcl}
\longrightarrow^{\Diamond}_{\mathcal{E}} & \triangleq & \{(s,r,x) \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} S' \times \{r\} \times \{1\} \mid s \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}_1} S'\} \\
 & \cup & \{(s,r,x) \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} \{s\} \times R' \times \{2\} \mid r \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}_2} R'\} \\
\longrightarrow^{\Box}_{\mathcal{E}} & \triangleq & \{(s,r,x) \xrightarrow{a}{}^{\Box}_{\mathcal{E}} S' \times \{r\} \times \{1\} \mid s \xrightarrow{a}{}^{\Box}_{\mathcal{E}_1} S'\} \\
 & \cup & \{(s,r,x) \xrightarrow{a}{}^{\Box}_{\mathcal{E}} \{s\} \times R' \times \{2\} \mid r \xrightarrow{a}{}^{\Box}_{\mathcal{E}_2} R'\} \\
c & \triangleq & \{(s,r,1) \mapsto c_1(s) \cdot 0^{k_2} \mid s \in S_{\mathcal{E}_1} \wedge r \in S_{\mathcal{E}_2}\} \\
 & \cup & \{(s,r,2) \mapsto 0^{k_1} \cdot c_2(r) \mid s \in S_{\mathcal{E}_1} \wedge r \in S_{\mathcal{E}_2}\} \\
\lambda & \triangleq & \{(s,r,x) \mapsto \emptyset \mid s \in S_{\mathcal{E}_1} \wedge r \in S_{\mathcal{E}_2} \wedge x \in \{1,2\}\}
\end{array}$$

Figure B.3: EMTS Construction for Process Algebra Terms

## Composing EMTSs

We extend the function $\varepsilon$ to the domain of OTAs so that $\varepsilon(\Gamma \rhd E) = (\mathcal{E},S,\lambda)$, where $\mathcal{E} = (S_{\mathcal{E}}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, c)$ is an EMTS, $S \subseteq S_{\mathcal{E}}$ is the set of *start states* of $\mathcal{E}$, and $\lambda : S_{\mathcal{E}} \to 2^{RecProcVar}$ is a *labeling function*.

The function $\varepsilon$ is defined inductively on the structure of $E$ as shown in Figure B.3. In the definition, we let $\varepsilon(\Gamma \rhd E_1)$ be $((S_{\mathcal{E}_1}, A, \longrightarrow^{\Diamond}_{\mathcal{E}_1}, \longrightarrow^{\Box}_{\mathcal{E}_1}, c_1), S_1, \lambda_1)$ and $\varepsilon(\Gamma \rhd E_2)$ be $((S_{\mathcal{E}_2}, A, \longrightarrow^{\Diamond}_{\mathcal{E}_2}, \longrightarrow^{\Box}_{\mathcal{E}_2}, c_2), S_2, \lambda_2)$, where $S_{\mathcal{E}_1}$ and $S_{\mathcal{E}_2}$ are disjoint sets. The new state $s_{new}$ is not in $S_{\mathcal{E}_1}$. The coloring functions $c_1 : S_{\mathcal{E}_1} \to \mathbb{N}^{k_1}$ and $c_2 : S_{\mathcal{E}_2} \to \mathbb{N}^{k_2}$ color the states of $\mathcal{E}_1$ and $\mathcal{E}_2$ with integer tuples of length $k_1$ and $k_2$ respectively.

The EMTS corresponding to the nil process **0** consists of an abstract state without outgoing transitions, indicating that no transition is allowed for processes simulated by this state. If a process variable $X$ in the term $E$ stands for an underspecified component of the system, that is if $X$ is an assumption process variable, then the EMTS for $X$ is a maximal model for the conjunction of the properties specified for this component in the assumption list $\Gamma$.

The EMTS for a recursion process variable $X$ is a single state without outgoing transitions, since the capabilities of the processes simulated are determined by the binding *fix* -expression. The function $\lambda$ labels the state $X$. Given the EMTS for the term of the *fix* -expression where $X$ is free, the transitions of the start states are transferred to the states labeled by $X$.

The EMTS for a subterm prefixed by an action $a$ is given by a start state with a must $a$-transition to the set of start states of the EMTS for the subterm. The EMTS for the sum operator consists of an EMTS where the start states are the cross product of the start states of the EMTSs for the subterms. It is assumed for this case that there are no incoming transitions to the start states of the EMTSs being combined. This is an invariant of the construction, except the case for tt which can be trivially converted to an equivalent EMTS to satisfy the property.

Finally, the states of the EMTS for a parallel composition of two components consists of a state from each component. Each state has transitions such that one of the components make a transition while the other stays in the same state. Each state is further marked by 1 or 2 to keep track of which component has performed the last transition; this is necessary to enable a run of the composition if the interleaved runs are enabled.

## Correctness Results

The aim of the above construction is to capture, by means of an EMTS, exactly those behaviors denoted by the given OTA. The construction is *sound* (resp. *complete*) if the denotation of the OTA is a subset (resp. superset) of the denotation of the resulting EMTS. Our first result establishes that the first part of the construction is a maximal model construction for the modal $\mu$-calculus.

**Theorem B.7.** *Let $\mathcal{T}$ be a transition-closed LTS, $\Phi$ be a closed and guarded modal $\mu$-calculus formula and $\varepsilon(\Phi) = (\mathcal{E}, S, \lambda)$. Then $[\![S]\!]_{\mathcal{T}} = ||\Phi||^{\mathcal{T}}$.*

Our next result shows that the construction is sound and complete when assumptions exist on only one of the components that are running in parallel and the rest of the system is fully determined.

**Theorem B.8.** *Let $\mathcal{T}$ be a transition-closed LTS, $\Gamma \rhd E \parallel t$ be a guarded linear OTA where $E$ does not contain parallel composition, and $t$ is closed, and let $\varepsilon(\Gamma \rhd E \parallel t) = (\mathcal{E}, S, \lambda)$. Then $[\![S]\!]_{\mathcal{T}}$ is equal to the set $[\![\Gamma \rhd E \parallel t]\!]_{\rho_0}$ up to bisimulation, where $\rho_0$ maps each recursion process variable $X$ to $\mathbf{0}$.*

Theorems B.7 and B.8 are proved by induction on the structure of the logical formula and the process term, respectively, and can be found in Appendix C.

In the general case, when multiple underspecified components run in parallel, we only have soundness: our construction is sound for systems without dynamic process creation. For systems with dynamic process creation, the construction does not terminate.

**Theorem B.9.** *Let $\mathcal{T}$ be a transition-closed LTS, $\Gamma \rhd E$ be a guarded linear OTA where every recursion process variable in the scope of parallel composition is bound by a* fix *operator in the same scope, and let $\varepsilon(\Gamma \rhd E) = (\mathcal{E}, S, \lambda)$. Then the set $[\![S]\!]_{\mathcal{T}}$ includes $[\![\Gamma \rhd E]\!]_{\rho_0}$ up to bisimulation.*

The proof of the theorem is as the proof of Theorem B.8, but includes a more general case for parallel composition and can be found in Appendix C.

Our last result reflects the fact that verification of open systems in the presence of parallel composition is undecidable for the modal $\mu$-calculus in general. Completeness results can, however, be obtained for various fragments of the $\mu$-calculus, such as ACTL, ACTL* and the simulation logic of [35]. In our approach, the tasks of constructing a finite representation of the state space in the form of an EMTS and the task of verifying properties of this representation are separated. This allows different logics to be employed for expressing assumptions on components and for specifying system properties, giving rise to more refined completeness results.

## B.5 A proof system for EMTS

In [2], we presented a proof system for verifying that an abstract state $s$ of an EMTS $\mathcal{E}$ satisfies a modal $\mu$-calculus formula $\Phi$. In this section, we give a summary of this proof system and provide an alternative termination condition that uses the coloring function $c$ instead of the earlier condition that assumed an extensional definition of the set of prohibited runs $W_{\mathcal{E}}$. The system is a specialization of a proof system by Bradfield and Stirling [7, 36] for showing $\mu$-calculus properties for sets of LTS states. The relationship between the two proof systems is clear when one considers that each EMTS state denotes a set of LTS states.

A proof tree is constructed using the rules below, where $\sigma$ ranges over $\mu$ and $\nu$. The construction starts with the goal and progresses in a goal-directed fashion, checking at each step if a terminal node was reached.

$$\frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \wedge \Psi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \quad s \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi} \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \vee \Psi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \vee \Psi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi}$$

$$\frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \sigma Z.\Phi}{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z} \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} [a]\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \ \ldots \ s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} \{s_1, \ldots, s_n\} = \cup\, \partial^{\Diamond}_a(s)$$

$$\frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} Z}{s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} Z \text{ identifies } \sigma Z.\Phi \qquad \frac{s \vdash^{\mathcal{E}}_{\mathcal{V}} \langle a\rangle\,\Phi}{s_1 \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi \ \ldots \ s_n \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi} \{s_1, \ldots, s_n\} \in \partial^{\Box}_a(s)$$

A successful tableau (or proof) is a finite proof tree having successful terminals as leaves. If $n : r \vdash^{\mathcal{E}}_{\mathcal{V}} Z$ is a node where $Z$ identifies a fixed point formula, and there is an identical ancestor node of $n$, $n' : r \vdash^{\mathcal{E}}_{\mathcal{V}} Z$ and for any other fixed point variable $Y$ on this path, $Z$ subsumes $Y$, then node $n$ is called a $\sigma$-terminal. So no further rules are applied to it. The most recent node making $n$ a $\sigma$-terminal is named $n$'s companion. The conditions for a leaf node $r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$ of a proof tree to be a successful terminal are listed below.

**Successful Terminals**

1. $\Psi = \text{tt}$, or else $\Psi = Z$, $Z$ is free in the initial formula, and $r \in \mathcal{V}(Z)$
2. $\Psi = [a]\,\Phi$ and $\cup\partial^{\Diamond}_a(r) = \emptyset$
3. $\Psi = Z$ where $Z$ identifies a fixed point formula $\sigma Z.\Phi$, and the sequent is a $\sigma$-terminal with companion node $n :\ r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$, then
   a) If $\sigma = \nu$, then the terminal is successful.
   b) If $\sigma = \mu$, then the terminal is successful if every infinite run of the EMTS that corresponds to an infinite sequence of trails of the companion node $n_0$ is in $W_{\mathcal{E}}$. (The notion of trail is explained below.) When the set $W_{\mathcal{E}}$ is encoded using a coloring function $c$, the condition is that for any set $S_T$ of trails of $n_0$, there should exist $1 \leq j \leq k$, so that $\mathit{max}(\underset{T \in S_T}{\cup} c(\alpha(T))(j))$ is odd. This ensures, for an infinite run $w_{n_0} = \alpha(T_1) \circ \alpha(T_2) \circ \alpha(T_3) \ldots$ where for all $i \geq 1$, $T_i$ is a trail of $n_0$, that there exists some $1 \leq j' \leq k$ such that $\mathit{max}\,(\mathit{inf}(c(w_{n_0})(j')))$ is odd.

**Unsuccessful Terminals**

1. $\Psi = \text{ff}$, or else $\Psi = Z$, $Z$ is free in the initial formula, and $r \notin \mathcal{V}(Z)$
2. $\Psi = \langle a\rangle\,\Phi$ and $\cup\partial^{\Box}_a(r) = \emptyset$

3. $\Psi = Z$ where $Z$ identifies the least fixed point formula $\mu Z.\Phi$, and the sequent is a $\sigma$-terminal with companion node $n_0$, then the terminal is unsuccessful if there exists a set $S_T$ of trails of $n_0$ such that for every $1 \leq j \leq k$, $max\,(\bigcup_{T \in S_T} c(\alpha(T))(j))$ is even. This means that some infinite run $w_{n_0}$ of the EMTS, which corresponds to an infinite sequence of trails of the companion node $n_0$, is not in $W_{\mathcal{E}}$.

Trails and corresponding runs are defined as follows. Assume that node $n_k{:}r \vdash^{\mathcal{E}}_{\mathcal{V}} Z$ is a $\mu$-terminal and node $n_0{:}r \vdash^{\mathcal{E}}_{\mathcal{V}} Z$ is its companion. A trail $T$ of the companion node $n_0$ is a sequence of state–node pairs $(r, n_0), \ldots, (r, n_k)$ such that for all $0 \leq i < k$, one of the following holds:

1. $n_{i+1} : r_{i+1} \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi_{i+1}$ is an immediate successor of $n_i : r_i \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi_i$, or

2. $n_i$ is the immediate predecessor of a $\sigma$-terminal node $n' : r' \vdash^{\mathcal{E}}_{\mathcal{V}} Z'$ where $n' \neq n_k$ whose companion is $n_j : r' \vdash^{\mathcal{E}}_{\mathcal{V}} Z'$ for some $j : 0 \leq j \leq i$, $n_{i+1} = n_j$, and $r_{i+1} = r'$.

In order to convert a trail to a corresponding run, we use the function $\alpha$, which returns the empty string when the trail contains only one pair, and is defined for longer trails as follows:

$$\alpha((r_1, n_1) \cdot (r_2, n_2) \cdot T) \triangleq \begin{cases} (r_1 \xrightarrow{a}_{\mathcal{E}} r_2) \cdot \alpha((r_2, n_2) \cdot T) & \Box_a \text{ or } \Diamond_a\text{-rule is applied to } n_1 \\ \alpha((r_2, n_2) \cdot T) & \text{otherwise.} \end{cases}$$

A formula is prime if whenever it logically implies a disjunction then it also implies one of the disjuncts. As we show in [2], the proof system is sound and complete for all formulas with only prime subformulas.

## B.6 Conclusion

In this paper we investigate a state space representation for open systems specified as open process terms with behavioural assumptions written in the modal $\mu$-calculus. This representation can serve both as a graphical specification formalism and as a basis for verification, supporting state space exploration based techniques and state visualization for interactive methods. We present a two-phase construction of such a representation from an open term with assumptions, and show it sound for terms without dynamic process creation and complete for systems with a single underspecified component. Finally, we adapt an existing proof system for the task of proving behavioural properties of open systems based on the given state space representation. The relative simplicity of the proof system and its use is an indication of the adequateness of EMTSs for open system state space representation.

Future work is required to characterize more precisely the construction and the $\mu$-calculus fragments for which it is complete, taking into account that the fragment for specifying component assumptions need not be the same as the fragment chosen for specifying system properties. In addition to automatic state space construction, interactive state space exploration will be considered, allowing a wider class of open systems to be handled. Finally, we plan to demonstrate the utility of the proposed approach by means of tool support and case studies.

# Appendix C

# Proofs

**Proposition C.1.** *For any $s \in S_{\mathcal{E}}$ and LTS $\mathcal{T}$, $\partial_a(\llbracket s \rrbracket_{\mathcal{T}}) \subseteq \llbracket \cup\, \partial_a^{\Diamond}(s) \rrbracket_{\mathcal{T}}$*

*Proof.* Assume $t' \in \partial_a(\llbracket s \rrbracket_{\mathcal{T}})$. Then, by the notion of $a$-derivatives for LTS, there is a $t \in \llbracket s \rrbracket_{\mathcal{T}}$ s.t. $t \xrightarrow{a}_{\mathcal{T}} t'$. Then, by the definition of denotation, $t \preceq s$ and hence, by the definition of simulation, there are $S'$ and $s' \in S'$ such that $s \xrightarrow{a}{}^{\Diamond}_{\mathcal{E}} S'$ and $t' \preceq s'$. Therefore $t' \in \llbracket s' \rrbracket_{\mathcal{T}}$ and hence, by the definition of $a$-derivatives for EMTS, $t' \in \llbracket \cup \partial_a^{\Diamond}(s) \rrbracket_{\mathcal{T}}$. □

**Lemma A.10** *For each rule instance in $\Sigma_{\mathcal{E}}$, translation $\pi$ assigns a correct proof tree in $\Sigma_{\mathcal{T}}$, so that each premise (resp. conclusion), $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$, of the rule is matched by a leaf (resp. root), $\llbracket s \rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi$, and all unmatched leaves of the constructed proof tree are successful terminals.*

*Proof.* We consider each rule of $\Sigma_{\mathcal{E}}$ in turn. The cases for $\wedge$**-rule**, $\vee$**-rule**, $\sigma$**-rule**, and $Z$**-rule** are trivial.

For the $\Box_a$**-rule**, in the first case we have $\cup\partial_a^{\Diamond}(s) = \emptyset$ Then $\Box_a$ rule is applied to node $\llbracket r \rrbracket_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} [a]\,\Psi'$ in $\Sigma_{\mathcal{T}}$. Since none of the states implementing $r$ have $a$-derivatives, the resulting node is $\emptyset \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi'$, which is a successful terminal. For the second case, we use $\partial_a(\llbracket s \rrbracket_{\mathcal{T}}) \subseteq \llbracket \cup\partial_a^{\Diamond}(s) \rrbracket_{\mathcal{T}}$ by Proposition C.1. Then, the Thin rule is only applied if the subset relation between the range of $f_a$ and $\llbracket \{s_1 \ldots s_n\} \rrbracket_{\mathcal{T}}$ is proper. Finally, a sequence of Cut-rules are applied.

For the $\Diamond_a$**-rule**, by the side condition we have $s \xrightarrow{a}{}^{\Box}_{\mathcal{E}} \{s_1, \ldots, s_n\}$, and hence, by the definitions of simulation and denotation, for every $t \in \llbracket s \rrbracket_{\mathcal{T}}$ there must be a $s' \in \{s_1, \ldots, s_n\}$ and $t' \in \llbracket s' \rrbracket_{\mathcal{T}}$ such that $t \xrightarrow{a}_{\mathcal{T}} t'$. Then the mapping $f_a$ mapping each $t \in \llbracket s \rrbracket_{\mathcal{T}}$ to such a corresponding $t'$ is a valid choice function for rule $\Diamond_a$ of $\Sigma_{\mathcal{T}}$. Again the Thin rule is applied only when the subset relation is a proper one. The applications of the Cut-rules follow to split $\llbracket \{s_1, \ldots, s_n\} \rrbracket_{\mathcal{T}}$ to the denotation of individual states. □

**Corollary A.11** For proof tree $A_{\mathcal{E}}$ with root sequent $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ in $\Sigma_{\mathcal{E}}$, $A_{\mathcal{T}} = \pi_{\mathcal{T}}(A_{\mathcal{E}})$ is a proof tree with root sequent $[\![s]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi$ in $\Sigma_{\mathcal{T}}$, such that each leaf $s_i \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi_i$ of $A_{\mathcal{E}}$ is matched by a leaf $[\![s_i]\!]_{\mathcal{T}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Phi_i$ in $A_{\mathcal{T}}$, and all unmatched leaves in $A_{\mathcal{T}}$ are successful terminals.

*Proof.* Follows directly from Lemma A.10 by induction on the depth of $A_{\mathcal{E}}$. □

*Canonical Proof Constructions and the Matching Function* $\gamma$. Let $\mathcal{E}$ be a finite-state EMTS, $s \in S_{\mathcal{E}}$, and $\Phi$ have prime subformulas only. If $s \vDash^{\mathcal{E}}_{\mathrm{V}} \Phi$, then the construction process of the proof trees $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$ in the weakened version of $\Sigma_{\mathcal{T}}$ for the goal $[\![s]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi$ is presented below. The construction is described through the matching function $\gamma : \Gamma_N \to \Gamma_M \cup \{\perp\}$, where $\Gamma_N$ and $\Gamma_M$ are the node spaces of $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$, respectively and $\perp$ stands for the "undefined value". The definition of $\gamma$ is given as the construction progresses.

**Construction**

1. The root nodes $n_0$ of $A_{\mathcal{U}}$ and $m_0$ of $A^*_{\mathcal{U}}$ both contain the sequent: $[\![s]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi$. ($\gamma(n_0) \triangleq m_0$)

2. Thin rule is applied to $n_0$ to produce the subgoal $n_1 : ||\Phi||^{\mathcal{U}}_{\mathbf{V}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi$. ($\gamma(n_1) \triangleq m_0$)

3. If the current subgoal in $A_{\mathcal{U}}$ is $n : S_n \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$:

   a) $\gamma(n) = m$, where $m : [\![s_m]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$

      - $\Psi = \Psi_1 \wedge \Psi_2$ : Apply $\wedge$-rule to $n$ and $\wedge$-macro to $m$ to get the new subgoals $n^1 : S_n \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi_1$,$n^2 : S_n \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi_2$ and $m^1 : [\![s_m]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi_1$, $m^2 : [\![s_m]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi_2$ respectively. ($\gamma(n^1) \triangleq m^1$ and $\gamma(n^2) \triangleq m^2$).
      - $\Psi = \Psi_1 \vee \Psi_2$ : Pick $\Psi_i$ where $i \in 1, 2$ so that $\Psi$ implies $\Psi_i$. Apply $\vee$-rule to $n$ and $\vee$-macro to $m$ to get the new subgoals $n' : S_n \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi_i$ and $m' : [\![s_m]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi_i$ respectively. ($\gamma(n') \triangleq m'$)
      - $\Psi = [a]\,\Psi'$ : Apply $\Box$-rule to $n$ and the corresponding $\Box$-macro[1] to $m$ according to whether $\partial^{\Diamond}_a(s_m) = \emptyset$. Let the *immediate* subgoals of $n$ and $m$ be $n'$ and $m'$, respectively. ($\gamma(n') \triangleq m'$)
        If $\cup\partial^{\Diamond}_a(s_m) = \{s_1 \dots s_k\}$ where $k > 1$, then apply $k-1$ consecutive Cut-rules to $n'$. For every application, if the current subgoal is $n_j$, then the newly produced subgoals are $n_{j+1} : \partial_a(S_n) \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi'$ and $n_{j+2} : \partial_a(S_n) \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi'$ with the subgoals of $\gamma(n_j)$ being $m_{j+1} : [\![\{s_1 \dots s_{k'-1}\}]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi'$ and $m_{j+2} : [\![s_{k'}]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi'$ where $1 \le k' \le k$. Apply the next Cut rule to $n_{j+1}$. ($\gamma(n_{j+1}) \triangleq m_{j+1}$ and $\gamma(n_{j+2}) \triangleq m_{j+2}$)

[1]No application of the Thin rule is needed in this macro. See Proof of Proposition C.1

- $\Psi = \langle a \rangle \, \Psi'$ : Pick $S' \in \partial_a^{\Box}(s_m)$ with for all $s' \in S'$, $s' \vdash_{\mathrm{V}}^{\mathcal{E}} \Psi'$. Apply $\Diamond$-rule to $n$ and $\Diamond$-macro[2] to $m$ using the choice functions $f_a$ and $f_a^{\mathcal{E}}$ respectively, where $f_a$ is some choice function that preserves validity with $f_a|_{[\![s_m]\!]_{\mathcal{U}}} = f_a^{\mathcal{E}}$ and $f_a^{\mathcal{E}}$ maps each $t \in [\![s_m]\!]_{\mathcal{U}}$ to some $t' \in [\![S']\!]_{\mathcal{U}}$ such that $t \xrightarrow{a} t'$ and there exists $s' \in S'$ where $s'$ simulates $t'$. Let the *immediate* subgoals of $n$ and $m$ be $n'$ and $m'$, respectively. $(\gamma(n') \stackrel{\Delta}{=} m')$
  If $S' = \{s_1 \dots s_k\}$ where $k > 1$, then apply $k-1$ consecutive Cut-rules to $n'$. For every application, if the current subgoal is $n_j$, then the newly produced subgoals are $n_{j+1} : f_a(S_n) \vdash_{\mathbf{V}}^{\mathcal{U}} \Psi'$ and $n_{j+2} : f_a(S_n) \vdash_{\mathbf{V}}^{\mathcal{U}} \Psi'$ with the subgoals of $\gamma(n_j)$ being $m_{j+1} : [\![\{s_1 \dots s_{k'-1}\}]\!]_{\mathcal{U}} \vdash_{\mathbf{V}}^{\mathcal{U}} \Psi'$ and $m_{j+2} : [\![s_{k'}]\!]_{\mathcal{U}} \vdash_{\mathbf{V}}^{\mathcal{U}} \Psi'$ where $1 \leq k' \leq i$. Apply the next Cut rule to $n_{j+1}$. $(\gamma(n_{j+1}) \stackrel{\Delta}{=} m_{j+1}$ and $\gamma(n_{j+2}) \stackrel{\Delta}{=} m_{j+2})$
- $\Psi = \sigma Z.\Psi'$ : Apply the Thin rule to $n$ to get the new subgoal $n''$. $(\gamma(n'') \stackrel{\Delta}{=} m)$
  Apply $\sigma Z$ rule to $n''$ and $\sigma Z$ macro to $m$ to get the new subgoals $n'$ and $m'$, respectively. $(\gamma(n') \stackrel{\Delta}{=} m')$
- $\Psi = Z$ where $Z$ identifies the fixed point formula $\sigma Z.\Psi'$: Apply $Z$ rule to $n$ and $Z$ macro to $m$ to get the new subgoals $n'$ and $m'$, respectively. $(\gamma(n') \stackrel{\Delta}{=} m')$

Let $n^1 \dots n^i$ where $i \in \{1, 2\}$ be the current subgoals in $A_{\mathcal{U}}$:
Test for each subgoal $n^j$ if it is a terminal using the conditions below. If $n^j$ is a terminal than no further rules are applied to $\gamma(n^j)$. If $n^j$ is not a terminal, repeat Step (iii) for each $n^j$.

b) $\gamma(n) = m$, where $m : \emptyset \vdash_{\mathrm{V}}^{\mathcal{T}} \Psi$, which means $m$ is a successful terminal of $A_{\mathcal{U}}^*$ and will not be applied any further rules.

   Apply to $n$ the corresponding rule according to the structure of $\Psi$ as in a). If the rule to be applied is $\Diamond$, the choice function $f$ can be picked as any that preserves validity.

   Let $n^1 \dots n^i$ where $i \in \{1, 2\}$ be the subgoals of $n$. A subgoal $n^j$ of n is a terminal if it obeys one of the termination conditions stated in Section A.3 for the original proof system. For the subgoals that are not terminals the process is repeated beginning from Step (iii).$(\gamma(n^j) \stackrel{\Delta}{=} \perp$ where $1 \leq j \leq i)$

c) $\gamma(n) = \perp$ Apply corresponding rule according to the structure of $\Psi$ as in a). If the rule to be applied is $\Diamond$, the choice function $f$ can be picked as any that preserves validity.

[2]No application of the Thin rule is needed in this macro. See proof of Proposition C.3

Let $n^1 \ldots n^i$ where $i \in \{1,2\}$ be the subgoals of $n$. A subgoal $n^j$ of n is a terminal if it obeys one of the termination conditions stated in Section A.3 for the original proof system. For the subgoals that are not terminals the process is repeated beginning from Step (iii).($\gamma(n^j) \triangleq \bot$ where $1 \leq j \leq i$)

Successful Termination for node $n' : S'_n \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi'$ of $A_{\mathcal{U}}$:

1. $\Psi' = \mathrm{tt}$, or else $\Psi' = Z$, $Z$ is free in the initial formula, and $S'_n \subseteq \mathbf{V}(Z)$
2. $S'_n = \emptyset$
3. $\Psi' = Z$ where $Z$ identifies a fixed point formula $\sigma Z.\Phi$, then this sequent is a $\sigma$-terminal if node $\gamma(n') = m'$ in $A^*_{\mathcal{U}}$ where $m' : [\![s_{m'}]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi'$ is a $\sigma$-terminal with companion node $m''$, which mentions same sequent $m'' : [\![s_{m'}]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi'$ and the companion node of $n$ is $\gamma^{-1}(m'')$. The terminal is successful when $\sigma = \nu$. If $\sigma = \mu$, then the terminal is successful if there is no infinite chain of composable trails $T_0 \circ T_1 \circ T_2 \ldots$ of $\gamma^{-1}(m'')$ and $n'$.

It can be shown that the matching function $\gamma$ is surjective, that is for each node $m$ of $A^*_{\mathcal{U}}$ there exists at least one node $n$ of $A_{\mathcal{U}}$ such that $\gamma(n) = m$. Furthermore, one can say that for all nodes m of $A^*_{\mathcal{U}}$, the set $\gamma^{-1}(m)$ is either a singleton or $\gamma^{-1}(m)$ has two elements $n_1$ and $n_2$, when the rule applied to $n_1$ is a Thin, with $n_2$ as the resulting subgoal.

**Proposition C.2.** *Let $\mathcal{E}$ be an EMTS. For universal LTS $\mathcal{U}$ and $s \in S_{\mathcal{E}}$, $\partial_a([\![s]\!]_{\mathcal{U}}) = [\![\cup\, \partial^{\Diamond}_a(s)]\!]_{\mathcal{U}}$*

*Proof.* We know $\partial_a([\![s]\!]_{\mathcal{T}}) \subseteq [\![\cup\, \partial^{\Diamond}_a(s)]\!]_{\mathcal{T}}$ for any LTS $\mathcal{T}$ by Proposition C.1. We go on to prove that $[\![\partial^{\Diamond}_a(s)]\!]_{\mathcal{U}} \subseteq \partial_a([\![s]\!]_{\mathcal{U}})$. Suppose for some LTS $\mathcal{T}$ there exists $t' \in [\![\partial^{\Diamond}_a(s)]\!]_{\mathcal{U}}$ where $t'$ is not an element of $\partial_a([\![s]\!]_{\mathcal{U}}$ and let $t$ be some state simulated by $s$. Let us built $\mathcal{T}'$ with the addition of a new state $t_{new}$, which have the same transitions with $t$, and additionally the transition $t_{new} \xrightarrow{a}_{\mathcal{T}} t'$. $t_{new}$ is also simulated by $s$ and must be in $[\![s]\!]_{\mathcal{U}}$. Then its $a$-derivative $t'$ must be in $\partial_a([\![s]\!]_{\mathcal{U}}$, which is a contradiction. □

**Proposition C.3.** *Let $\mathcal{E}$ be an EMTS. For universal LTS $\mathcal{U}$, $s \in S_{\mathcal{E}}$, for every $S' \in \partial^{\Box}_a(s)$, there exists a choice function $f_a$ such that $f_a([\![s]\!]_{\mathcal{U}}) = [\![S']\!]_{\mathcal{U}}$.*

*Proof.* By Definition A.2, for every $t \in [\![s]\!]_{\mathcal{U}}$ there exists $t' \in [\![S']\!]_{\mathcal{U}}$ such that $t \xrightarrow{a}_{\mathcal{T}} t'$. So we know that at least one choice function $f_a$ exists which takes the elements of $[\![s]\!]_{\mathcal{U}}$ to a subset of $[\![S']\!]_{\mathcal{U}}$. What is more, we can construct such an $f_a$ whose range covers all elements of $[\![S']\!]_{\mathcal{U}}$: Take any $t \in [\![s]\!]_{\mathcal{U}}$ and let $t' \in \partial_a(t)$ and $t \in [\![S']\!]_{\mathcal{U}}$. For each state $t'' \in [\![S']\!]_{\mathcal{U}}$, define $f_a(t_{new}) = t''$ where $t_{new}$ is a state that has all transitions of $t$ plus the transition $t_{new} \xrightarrow{a}_{\mathcal{T}} t''$. It is clear that each $t_{new}$ defined in this manner is simulated by $s$ and exists in $\mathcal{U}$.

□

**Proposition C.4.** *Let $\mathcal{E}$ be an EMTS. For universal LTS $\mathcal{U}$ and $s \in S_{\mathcal{E}}$, if $s \vDash^{\mathcal{E}}_{\mathrm{V}} \langle a \rangle \Psi$, then there exists $S' \in \partial^{\Box}_a(s)$ such that for all $s' \in S'$, $s' \vDash^{\mathcal{E}}_{\mathrm{V}} \Psi$.*

*Proof.* First, we prove that $\partial^{\Box}_a(s)$ is not empty. Assume $\partial^{\Box}_a(s) = \emptyset$. Then, the state $t_{new}$ which has all the transitions of some $t \in [\![s]\!]_{\mathcal{U}}$, with the exception of $a$-transitions would still be simulated by $s$. Then by Definition A.8, $t_{new} \models^{\mathcal{U}}_{\mathbf{V}} \langle a \rangle \Psi'$, but this is clearly not the case so we reach a contradiction.

Next, we prove that there exists $S' \in \partial^{\Box}_a(s)$ such that for all $s' \in S'$, $s' \vDash^{\mathcal{E}}_{\mathrm{V}} \Psi$. Assume for all $S' \in \partial^{\Box}_a(s)$, $S'$ does not satisfy $\Psi'$. In such a case the state $t_{brandnew}$ which has all the transitions of a state $t \in [\![s]\!]_{\mathcal{U}}$ excluding $a$-transitions of $t$ and with the addition of the transition $t_{brandnew} \xrightarrow{a}_{\mathcal{T}} t'$ for some $t' \in [\![S']\!]_{\mathcal{U}}$, is still simulated by $s$. But then $t_{brandnew}$ does not satisfy $\langle a \rangle \Psi'$, which is a contradiction. □

**Lemma C.5.** *Let $\mathcal{E}$ be a finite-state EMTS and $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$ be proof trees constructed as described above. If for node $n : S_n \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi_n$ in $A_{\mathcal{U}}$, $\gamma(n) = m$ where $m : S_m \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi_m$ in $A^*_{\mathcal{U}}$, then $S_m \subseteq S_n$.*

*Proof.* In order to make our proof, we can use induction on the depth of the rule applications in $A_{\mathcal{U}}$.

As a base case, $n_0$ and $\gamma(n_0)$ are the roots of the trees and mention the same sets, so initially $S_m = S_n$.

Suppose node $n : S_n \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$ of $A_{\mathcal{U}}$ matches $m : S_m \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$ of $A^*_{\mathcal{U}}$ and $S_m \subseteq S_n$ we show the same subset condition holds for each subgoal produced by rule induction. If the rule applied to $n$ is:

- Thin, then let the immediate successor of $n$ be $n' : ||\Psi||^{\mathcal{U}}_{\mathbf{V}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$. We know that $\gamma(n') = m$ by definition and that $S_n \subseteq ||\Psi||^{\mathcal{U}}_{V}$, hence $S_m \subseteq ||\Psi||^{\mathcal{U}}_{V}$

- $\vee$, $\wedge$, $\sigma Z$ or $Z$, the sets $S_n$ and $S_m$ also occur in the immediate successors, so the property is preserved.

- Cut, then the set $S_n$ occurs in both subgoals $n^1$ and $n^2$, meanwhile the sets mentioned in $\gamma(n^1)$ and $\gamma(n^2)$ are both subsets of $S_m$, so they are also subsets of $S_n$.

- $\Box_a$, then $S_m = [\![s_m]\!]_{\mathcal{U}}$ for some state $s_m$ of $\mathcal{E}$. Then $\partial_a([\![s_m]\!]_{\mathcal{U}}) = [\![\cup \partial^{\Diamond}_a(s_m)]\!]_{\mathcal{U}}$ by Proposition C.4. Since $[\![s]\!]_{\mathcal{U}} \subseteq S_n$, $\partial_a([\![s]\!]_{\mathcal{U}}) \subseteq \partial_a(S_n)$. Hence $[\![\cup \partial^{\Diamond}_a(s_m)]\!]_{\mathcal{U}} \subseteq \partial_a(S_n)$.

- $\Diamond_a$ This is the case since we have selected $f_a|_{[\![s_m]\!]_{\mathcal{U}}} = f^{\mathcal{E}}_a$ in the construction.

□

**Lemma C.6.** *Let $\mathcal{E}$ be a finite-state EMTS, $s \in S_{\mathcal{U}}$, and $\Phi$ have prime subformulas only. If $s \vDash^{\mathcal{E}}_{\mathrm{V}} \Phi$, then let $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$ be proof trees constructed as described above. If $A_{\mathcal{U}}$ is a finite proof tree, than $A^*_{\mathcal{U}}$ is also a finite proof tree.*

*Proof.* First, we have to show the correctness of the application of the rules involved in macro applications, i.e. $A^*_{\mathcal{U}}$ is indeed a proof tree. Next, we show that validity is preserved with each macro application. Finally. we show that the application of macros guided by the rule applications in $A_{\mathcal{U}}$ can not go on forever, i.e. $A^*_{\mathcal{U}}$ is finite.

Let $m : [\![s_m]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$ be a node of $A^*_{\mathcal{U}}$ where $[\![s]\!]_{\mathcal{U}} \models^{\mathcal{U}}_{\mathbf{V}} \Psi$. If the macro applied to $m$ is:

- $\wedge$, $\sigma Z$ or $Z$, the application is identical to the corresponding rule application in $\Sigma_{\mathcal{T}}$.

- $\vee$ and $\Psi = \Psi_1 \vee \Psi_2$, the goal is guaranteed to be reduced to a single subgoal because $\Psi$ is a subformula of $\Phi$ and hence prime.

- $\Box_a$ where $\Psi = [a]\,\Psi'$ and $[\![s]\!]_{\mathcal{U}} \models^{\mathcal{U}}_{\mathbf{V}} [a]\,\Psi'$. We first have to show that no application of the Thin rule is needed. This is the case since $\partial_a([\![s]\!]_{\mathcal{U}}) = [\![\cup\partial^{\Diamond}_a(s)]\!]_{\mathcal{U}}$ by the definition of satisfaction, Definition A.8. This automatically shows that $[\![\cup\, \partial^{\Diamond}_a(s)]\!]_{\mathcal{U}} \models^{\mathcal{U}}_{\mathbf{V}} \Psi'$ by the preservation of validity in $A_{\mathcal{U}}$.

- $\Diamond_a$ where $\Psi = \langle a\rangle\,\Psi'$ and $[\![s]\!]_{\mathcal{U}} \models^{\mathcal{U}}_{\mathbf{V}} \langle a\rangle\,\Psi'$. The existence of a proper choice function $f^{\mathcal{E}}_a$ is a result of Proposition C.3 and Proposition C.4.

This concludes the proof of the correctness of the macro applications.

The fact that validity is preserved comes from Lemma C.5 and that validity is preserved in canonical proofs and hence is preserved in $A_{\mathcal{U}}$.

The fact that the tree is finite is obvious, because a macro is applied to a sequent of $A^*_{\mathcal{U}}$, if a rule is applied to the matching sequent in $A_{\mathcal{U}}$. Cut-rules are applied in $A_{\mathcal{U}}$ when a macro includes them so that after the whole construction each rule application in $A_{\mathcal{U}}$ is matched by a rule application in $A^*_{\mathcal{U}}$ except for Thin. The possible causes of nontermination could be the modifications we made on the proof system: the addition of the Cut-rule applications and the new termination condition.

The modified proof system allows for infinitely many Cut-rule applications since the sequent is not changed with each application, but the number of consecutive Cut-rule applications allowed in the construction is equal to the number of states the particular state of the $\mathcal{E}$ has transition to. So the number of applications is guaranteed to be finite since $\mathcal{E}$ is finite state.

The new termination condition used in the construction requires that for a repeating node in $A_{\mathcal{U}}$ to be a terminal, its matching node in $A^*_{\mathcal{U}}$ must be a $\sigma$-terminal which mentions exactly the same sequent with its companion. The set mentioned in repeating nodes is always the denotation of a single set and the number of states of the $\mathcal{E}$ is finite, so eventually we are guaranteed to reach the identical sequent in $A^*_{\mathcal{U}}$. Therefore the modified termination condition does not cause infinite proof trees. $\Box$

**Definition C.7** (Trail Translation)**.** Let $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$ be constructed as described above, and $T = (t_i, m_i), \ldots, (t_k, m_k)$ be a trail in $A^*_{\mathcal{U}}$ of the terminal node $m_k$ and its companion $m_i$. The function $\delta$ converts $T$ to a trail of the matching $\sigma$-terminal and its companion in $A_{\mathcal{U}}$ by replacing each node with the matching one(s) in $A_{\mathcal{U}}$:

$$\begin{array}{rl} \delta(\varepsilon) & \triangleq \varepsilon \\ ((t,m)\cdot T) & \triangleq \begin{cases} ((t,n)\cdot\delta(T)) & \gamma^{-1}(m) = \{n\} \\ ((t,n)\cdot(t,n')\cdot\delta(T)) & \gamma^{-1}(m) = \{n,n'\} \text{ with } n \text{ above } n' \text{ in } A_{\mathcal{U}} \end{cases} \end{array}$$

It is possible to use the same state $t_i$ for the trail of $A_{\mathcal{U}}$ because for each $m_i$ that is matched by a $n_i$, a state $t_i$ that occurs in node $m_i$ is also in $n_i$, by Lemma C.5.

**Lemma C.8.** *If $A_{\mathcal{U}}$ is a successful proof tree, then $A^*_{\mathcal{U}}$ is also a successful proof tree.*

*Proof.* We know by Lemma C.6 that we will get a correct tableaux $A^*_{\mathcal{U}}$. It remains to prove that if the first tableaux is successful, then the second is also successful.

Let $m : S_m \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$ be a leaf of $A^*_{\mathcal{U}}$, and $n : S_n \vdash^{\mathcal{U}}_{\mathbf{V}} \Psi$ where $\gamma(n) = m$ and the rule applied to $n$ is not Thin. Assume $n$ is a successful terminal.

If $S_m = \emptyset$, then it is trivially a successful terminal. For all other cases, since the termination is checked only after (possible) Cut-rule applications, $S_m = [\![s_m]\!]_{\mathcal{U}}$ for some state $s_m$ of $\mathcal{E}$.

If $\Psi$ is tt, $m$ is trivially successful. If $\Psi$ is a variable $Z$ which is free in the initial formula, then it should be the case that $S_n \subseteq V(Z)$, and by Lemma C.5, $[\![s]\!]_{\mathcal{U}} \subseteq V(Z)$.

In the case where $\Psi$ is a variable $Z$ that identifies $\sigma Z.\Psi'$, let $n'$ be the companion node of $n$. By construction, $\gamma(n') = m'$ where $m' : [\![s_m]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} Z$ in $A^*_{\mathcal{U}}$, and $m'$ a predecessor of $m$. Then $m$ is a successful terminal if $Z$ identifies $\nu Z.\Psi'$.

If $Z$ identifies $\mu Z.\Psi'$, it has to be shown that there is no infinite chain of composable trails of the companion node $m'$. Suppose there is such an infinite chain of composable trails, $\kappa_{m'}{=}T_0 \circ T_1 \circ T_2 \ldots$ in $A^*_{\mathcal{U}}$. Then there is a corresponding infinite chain of composable trails in $A_{\mathcal{U}}$ given by $\kappa_{n'}{=}\delta(T_0) \circ \delta(T_1) \circ \delta(T_2) \ldots$. However, $n$ is a successful terminal, therefore no such infinite chain of $n'$ exists. Hence we reach a contradiction. □

**Lemma C.9.** *If proof trees $A_{\mathcal{U}}$ and $A^*_{\mathcal{U}}$ constructed as described above are successful, then $A_{\mathcal{E}} = \pi^{-1}_{\mathcal{U}}(A^*_{\mathcal{U}})$ is also a successful proof tree.*

*Proof.* If $A_{\mathcal{U}}$ is a successful proof tree, $A^*_{\mathcal{U}}$ is a successful proof tree by Lemma C.8. The side conditions of the rules of $\Sigma_{\mathcal{E}}$ are satisfied by the correctness of the macro applications in $A^*_{\mathcal{U}}$. Thus, to establish the rest of the result, it suffices to show that each leaf of $A_{\mathcal{E}} = \pi^{-1}_{\mathcal{U}}(A^*_{\mathcal{U}})$ is a successful terminal.

By the definition of $\pi_{\mathcal{T}}$, it is easy to observe that each leaf of $A_{\mathcal{E}}$ is matched by a leaf of $A^*_{\mathcal{U}}$, except the case where the leaf node is $m : r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$ with the matching node being $n : [\![r]\!]_{\mathcal{U}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi$ with $\cup\partial^{\diamond}_{a}(r) = \emptyset$. Since $r$ does not have *may*-transitions, $m$ is a successful terminal.

Consider the successful terminal $n : [\![r]\!]_{\mathcal{U}} \vdash^{\mathcal{T}}_{\mathrm{V}} \Psi$ in $A^*_{\mathcal{U}}$ and the matching node of $A_{\mathcal{E}}$, $m : r \vdash^{\mathcal{E}}_{\mathcal{V}} \Psi$. The proof that the latter is also a successful terminal is trivial when $\Psi = \mathrm{tt}$, when $\Psi = Z$ and $Z$ is free in the initial formula, or when $Z$ identifies the formula $\nu Z.\Psi'$.

For the terminal node $m$ to be successful when $\Psi = Z$ and $Z$ identifies $\mu Z.\Psi'$, all runs that correspond to an infinite sequence of $\mathcal{E}$-trails of $m$ and its companion node $m'$ should be in $W$, that is there is no $w_{m'} = E_1 \circ E_2 \ldots$, where for all $i \geq 1$, $E_i$ begins with $(r, m')$ and ends with $(r, m)$, and $\alpha(w_{m'})$ is not in $W$.

Assume that such an infinite sequence, $w_{m'} = E_1 \circ E_2 \ldots$ exists, where $\alpha(w_{m'})$ is not in $W$. Then an LTS, $\mathcal{T}$, can be constructed such that for each $r_i$ in $S_{\mathcal{E}}$, there exists $r_{i'}$ in $S_{\mathcal{T}}$ and there exists a transition $r_{i'} \xrightarrow{a}_{\mathcal{T}} r_{j'}$, if and only if $r_i \xrightarrow{a}^{\Diamond}_{\mathcal{E}} r_j$.

It is clear that each state, $r_{i'}$, of $\mathcal{T}$ is simulated by a state $r_i$ of $\mathcal{E}$, so $r_{i'} \in [\![r_i]\!]_{\mathcal{T}}$. Since $\mathcal{U}$ contains $\mathcal{T}$, there is an infinite sequence of composable trails in $A^*_{\mathcal{U}}$ of the terminal node $n$ and its companion $n'$ that mentions the same states, $r_{i'}$. But since $n$ is a successful terminal, there can be no such infinite sequence of trails, hence we reach a contradiction.

□

**Theorem A.15 (Completeness)** *Let $\mathcal{E}$ be a finite–state EMTS, $s \in S_{\mathcal{E}}$, and let $\Phi$ have prime subformulas only. Then $s \models^{\mathcal{E}}_{\mathrm{V}} \Phi$ implies $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$.*

*Proof.* Assume $s \models^{\mathcal{E}}_{\mathrm{V}} \Phi$. Then, by Definition A.8, $[\![s]\!]_{\mathcal{T}} \models^{\mathcal{T}}_{\mathbf{V}} \Phi$ for any $\mathcal{T}$, and hence also $[\![s]\!]_{\mathcal{U}} \models^{\mathcal{U}}_{\mathbf{V}} \Phi$. By completeness of $\Sigma_{\mathcal{T}}$, there exists a family of canonical proofs for the goal $[\![s]\!]_{\mathcal{U}} \vdash^{\mathcal{U}}_{\mathbf{V}} \Phi$. $A_{\mathcal{U}}$, which can be viewed as a combination of several of these canonical proofs, can be constructed with $A^*_{\mathcal{U}}$ as described above. Then, by Lemma C.9, $\pi^{-1}_{\mathcal{U}}(A^*_{\mathcal{U}})$ is a proof of $s \vdash^{\mathcal{E}}_{\mathcal{V}} \Phi$ in $\Sigma_{\mathcal{E}}$. □

## Correctness of Maximal Model Construction

Before we proceed to the proof of the correctness, we give some definitions. We extend the notion of simulation for "labelled" EMTSs, that is for an EMTS $\mathcal{E}$ that is accompanied by a labeling function $\lambda : S_{\mathcal{E}} \to 2^{PropVar}$ which labels the states of $\mathcal{E}$ with propositional variables.

**Definition C.10** (Simulation For a Valuation). *$R \subseteq S_{\mathcal{T}} \times S_{\mathcal{E}}$ is a simulation relation* for valuation $\mathrm{V} : PropVar \to 2^{S_{\mathcal{T}}}$ if whenever $tRs$, $a \in A$ and $Z \in PropVar$:

1. if $t \xrightarrow{a}_{\mathcal{T}} T$, then there is an $S$ such that $s \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S$ and for each $t' \in T$, there exists a $s' \in S$ such that $t'Rs'$;

2. if $s \xrightarrow{a}^{\Box}_{\mathcal{E}} S$, then there is a $T$ such that $t \xrightarrow{a}^{\Box}_{\mathcal{E}} T$ and for each $t' \in T$, there exists a $s' \in S$ such that $t'Rs'$;

3. If for the run $\rho_s = s \xrightarrow{a_1}_{\mathcal{E}} s_1 \xrightarrow{a_2}_{\mathcal{E}} s_2 \xrightarrow{a_3}_{\mathcal{E}} \ldots \in W$ then there does not exist an infinite run $\rho_t = t \xrightarrow{a_1}_{\mathcal{T}} t_1 \xrightarrow{a_2}_{\mathcal{T}} t_2 \xrightarrow{a_3}_{\mathcal{T}} \ldots$ such that $t_i R s_i$ for all $i \geq 0$.

4. If $Z \in \lambda(s)$, then $t \in \mathrm{V}(Z)$.

We write $tR^{\mathrm{V}}s$ when $tRs$ for valuation V. We say that abstract state $s$ *simulates* state $t$ with respect to valuation V, denoted $t \preceq_{\mathrm{V}} s$, if there is a simulation relation $R$ such that $tR^{\mathrm{V}}s$.

We define the notion of denotation of an EMTS $\mathcal{E}$ with respect to a valuation as follows:

**Definition C.11** (EMTS Denotation)**.** Let $\mathcal{E}$ an EMTS with $(S_{\mathcal{E}}, A, \longrightarrow^{\Diamond}_{\mathcal{E}}, \longrightarrow^{\Box}_{\mathcal{E}}, c)$, $S \subseteq S_{\mathcal{E}}$ a set of start states of $\mathcal{E}$, and $\lambda : S_{\mathcal{E}} \rightarrow 2^{PropVar}$ a labeling function.

The denotation of a state $s \in S_{\mathcal{E}}$ with respect to an LTS $\mathcal{T}$ and valuation function $\mathrm{V} : PropVar \rightarrow 2^{S_{\mathcal{T}}}$ is defined as $[\![s]\!]^{\mathcal{T}}_{\mathrm{V}} \triangleq \{t \mid t \preceq_{\mathrm{V}} s\}$. The notion of denotation is lifted to sets of states in the natural way.

The denotation of a triple $(\mathcal{E},\ S,\ \lambda)$, then, is defined as the denotation of the start states:

$$[\![(\mathcal{E}, S, \lambda)]\!]^{\mathcal{T}}_{\mathrm{V}} \triangleq [\![S]\!]^{\mathcal{T}}_{\mathrm{V}}$$

Using these definitions we restate the Theorem 5.1 to include valuation V. Notice that this valuation does not have any effect when the states of the $\mathcal{E}$ are not labeled.

**Theorem 5.1( B.7)** $[\![\varepsilon(\Phi)]\!]^{\mathcal{T}}_{\mathrm{V}} = ||\Phi||^{\mathcal{T}}_{\mathrm{V}}$

*Proof.* The proof proceeds by induction on the structure of $\Phi$. Let $\varepsilon(\Psi_1) = ((S_{\mathcal{E}_1},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}_1},\ \longrightarrow^{\Box}_{\mathcal{E}_1},\ W_1),\ S_1,\ \lambda_1)$ and $\varepsilon(\Psi_2)= ((S_{\mathcal{E}_2},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}_2},\ \longrightarrow^{\Box}_{\mathcal{E}_2},\ W_2),\ S_2,\ \lambda_2)$ be constructed as defined in Figure B.2. In the cases below, IH stands for induction hypothesis and TR for transition rules of CCS.

- $\Phi \equiv \mathrm{tt}$
  $[\![S]\!]_{\mathcal{U}} = S_{\mathcal{U}} = ||\mathrm{tt}||^{\mathcal{U}}_{\mathrm{V}_0}$

- $\Phi \equiv \mathrm{ff}$
  $[\![S]\!]_{\mathcal{U}} = \emptyset = ||\mathrm{ff}||^{\mathcal{U}}_{\mathrm{V}_0}$

- $\Phi \equiv Z$
  $[\![S]\!]_{\mathcal{U}} = [\![\mathbf{0}]\!]_{\approx} = ||Z||^{\mathcal{U}}_{\mathrm{V}_0}$

- $\Phi \equiv \langle a \rangle\, \Psi_1$

  $t \preceq S$
  $\Longleftrightarrow \quad t \preceq s_{new}$ (Construction)
  $\Longleftrightarrow \quad \exists t'.t \xrightarrow{a}_{\mathcal{T}} t'$ where $t' \preceq S_1$ (Def. C.10)
  $\Longleftrightarrow \quad \exists t'.t \xrightarrow{a}_{\mathcal{T}} t'$ where $t' \models^{\mathcal{U}}_{\mathrm{V}_0} \Psi_1$ (Induction Hyp.)
  $\Longleftrightarrow \quad t \models^{\mathcal{U}}_{\mathrm{V}_0} \langle a \rangle\, \Psi_1$

- $\Phi \equiv [a]\,\Psi_1$

  $t \preceq S$
  $\iff \quad t \preceq s_{new}$ (Construction)
  $\iff \quad$ if there exists a transition $t \xrightarrow{a}_{\mathcal{T}} t'$, then $t' \preceq S_1$ (Def. C.10)
  $\iff \quad \forall t'. t \xrightarrow{a}_{\mathcal{T}} t' \implies t' \models^{\mathcal{U}}_{\mathrm{V}_0} \Psi_1$ (Induction Hyp.)
  $\iff \quad t \models^{\mathcal{U}}_{\mathrm{V}_0} [a]\,\Psi_1$

- $\Phi \equiv \Psi_1 \wedge \Psi_2$

  Assume $[\![\varepsilon(\Psi_i)]\!]^{\mathcal{T}}_{\mathrm{V}} = ||\Psi_i||^{\mathcal{T}}_{\mathrm{V}}$ for $i \in \{1,2\}$ (Induction Hyp.)

  We show $[\![\varepsilon(\Psi_1 \wedge \Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}} = ||\Psi_1 \wedge \Psi_2||^{\mathcal{T}}_{\mathrm{V}}$ in two parts.

  ($\subseteq$) Assume $t \in [\![\varepsilon(\Psi_1 \wedge \Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}}$, then there is a simulation relation $R_1 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1 \wedge \Psi_2)}$ for V such that $tR_1^{\mathrm{V}}(s,r)$ for some $(s,r) \in (S_{\varepsilon(\Psi_1)} \times S_{\varepsilon(\Psi_2)})|_{\Box} \cap (S_1 \times S_2)$ by definition of construction. We define $R_2' \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1)}$ as $tR_2's$ if and only if there exists $r \in S_{\varepsilon(\Psi_2)}.tR_1(s,r)$. We prove $R_2'$ is a simulation relation below. Hence, $t \in [\![\varepsilon(\Psi_1)]\!]^{\mathcal{T}}_{\mathrm{V}}$. By a similar argument we can show $t \in [\![\varepsilon(\Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}}$. Therefore, $t \in ||\Psi_1 \wedge \Psi_2||^{\mathcal{T}}_{\mathrm{V}}$

  Proof that $R_2'$ is a simulation relation:

  Assume $t\ R_2'\ s$.

  1. Whenever $a \in A$,
     a) If $t \xrightarrow{a}_{\mathcal{T}} t'$, by definition of simulation there exists $S$ such that $(s,r) \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S$ and $t'R_1(s',r')$ for some $(s',r') \in S$. By construction $(s,r) \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S$ if and only if there exists $S'$ and $R'$ such that $s \xrightarrow{a}^{\Diamond}_{\mathcal{E}_1} S'$, $r \xrightarrow{a}^{\Diamond}_{\mathcal{E}_2} R'$ and $S = S' \times R'$. Thus $s \xrightarrow{a}^{\Diamond}_{\mathcal{E}_1} S'$ where $t'R_2's'$ for some $s' \in S'$.
     b) If $s \xrightarrow{a}^{\Box}_{\mathcal{E}_1} S'$, then $(s,r) \xrightarrow{a}^{\Box}_{\mathcal{E}} S$ where $S = S' \times \cup\partial^{\Diamond}_a(r)$. (The set $\partial^{\Diamond}_a(r)$ can not be empty since in this case this state would not be a part of the constructed EMTS) Since $tR_1(s,r)$, $t \xrightarrow{a}_{\mathcal{T}} t'$ such that $t'R_2'(s',r')$ where $s' \in S'$ and $r' \in \cup\partial^{\Diamond}_a(r)$, so $t'R_2's'$.
  2. If $\rho_s = s \xrightarrow{a_1}_{\mathcal{E}_1} s_2 \xrightarrow{a_2}_{\mathcal{E}_1} s_3 \xrightarrow{a_3}_{\mathcal{E}_1} \ldots$ is in $W$, then no infinite run $\rho_t = t \xrightarrow{a_1}_{\mathcal{T}} t_2 \xrightarrow{a_2}_{\mathcal{T}} t_3 \xrightarrow{a_3}_{\mathcal{T}} \ldots$ of $t$ such that $t_i\ R_2'\ s_i$ for all $i \geq 1$ is possible. Such an infinite run $\rho_t$ is not possible because the run $\rho_{(s,r)} = (s,r) \xrightarrow{a_1}_{\mathcal{E}} (s_2,r_2) \xrightarrow{a_2}_{\mathcal{E}} (s_3,r_3) \xrightarrow{a_3}_{\mathcal{E}} \ldots \in W$ by construction.

  ($\supseteq$) Assume $t \in [\![\varepsilon(\Psi_1)]\!]^{\mathcal{T}}_{\mathrm{V}}$ and $t \in [\![\varepsilon(\Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}}$, then there are simulation relations $R_1 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1)}$ and $R_2 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_2)}$ for V such that $tR_1^{\mathrm{V}}s$ and $tR_2^{\mathrm{V}}s$ for some $s_1 \in S_1$ and $r_1 \in S_2$ by definition of construction. We define $R \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1 \wedge \Psi_2)}$ as $tR(s,r)$ if and only if $tR_1s$ and $tR_1r$. We prove $R$ is a simulation relation below. Hence, $t \in [\![\varepsilon(\Psi_1 \wedge \Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}}$.

Assume $t\ R\ (s, r)$.

1. Whenever $a \in A$,

   a) If $t \xrightarrow{a}_{\mathcal{T}} t'$, by Induction Hypothesis and definition of simulation there exists $S'$ and $R'$ such that $s \xrightarrow{a}^{\Diamond}_{\mathcal{E}_1} S'$, $r \xrightarrow{a}^{\Diamond}_{\mathcal{E}_1} R'$ and $t' R_1 s'$, $t' R_2 r'$ for some $s' \in S'$ and $r' \in R'$. Then by construction there exists $S$ where $(s, r) \xrightarrow{a}^{\Diamond}_{\mathcal{E}} S$ and $(s', r') \in S$ and $t' R(s', r')$.

   b) If $(s, r) \xrightarrow{a}^{\Box}_{\mathcal{E}} S$, then either there exists $S'$ such that $s \xrightarrow{a}^{\Box}_{\mathcal{E}} S'$ and $S = \{(s', r') \mid s' \in S' \wedge r' \in \cup\partial^{\Diamond}_{a}(r)\}$ or there exists $R'$ such that $r \xrightarrow{a}^{\Box}_{\mathcal{E}} R'$ and $S = \{(s', r') \mid s' \in \cup\partial^{\Diamond}_{a}(s) \wedge r' \in R'\}$.
   If it is the first case, by the definition of of simulation there exists $t'$ such that $t \xrightarrow{a}_{\mathcal{T}} t'$, and $t' R_1 s'$ for some $s'$ in $S'$. Since $t$ is also simulated by $r$, there exists some $R' \in \cup\partial^{\Diamond}_{a}(r)$ such that $t' R_2 r'$ for some $r' \in R'$. Then $(s', r') \in S$ and $t' R(s', r')$.
   The second case is similar.

2. If $\rho_{(s,r)} = (s, r) \xrightarrow{a_1}_{\mathcal{E}} (s_2, r_2) \xrightarrow{a_2}_{\mathcal{E}} (s_3, r_3) \xrightarrow{a_3}_{\mathcal{E}} \ldots$ is in $W$, then the infinite run $\rho_t = t \xrightarrow{a_1}_{\mathcal{T}} t_2 \xrightarrow{a_2}_{\mathcal{T}} t_3 \xrightarrow{a_3}_{\mathcal{T}} \ldots$ such that $t_i\ R\ (s_i, r_i)$ for all $i \geq 1$ should not be possible.
   Assume that such an infinite run $\rho_{t_1}$ exists. By construction, $\rho_{(s,r)} \in W$ if and only if $\rho_s = s \xrightarrow{a_1}_{\mathcal{E}_1} s_2 \xrightarrow{a_2}_{\mathcal{E}_1} s_3 \xrightarrow{a_3}_{\mathcal{E})_1} \ldots \in W_1$ or $\rho_r = r \xrightarrow{a_1}_{\mathcal{E}_2} r_2 \xrightarrow{a_2}_{\mathcal{E}_2} r_3 \xrightarrow{a_3}_{\mathcal{E})_2} \ldots \in W_2$. If it is the first case and $\rho_s \in W_1$, by definition of simulation there is no infinite run $\rho_t = t \xrightarrow{a_1}_{\mathcal{T}} t'_2 \xrightarrow{a_2}_{\mathcal{T}} \longrightarrow_{\mathcal{T}} t'_3 \xrightarrow{a_3}_{\mathcal{T}} \ldots$ where $t'_i\ R_1\ s_i$ for all $i \geq 1$. However, $\rho_t$ is such a run hence we reach a contradiction. The argument is similar if $\rho_r \in W_2$.

- $\Phi \equiv \Psi_1 \vee \Psi_2$

  Assume $[\![\varepsilon(\Psi_i)]\!]^{\mathcal{T}}_{\mathrm{V}} = ||\Psi_i||^{\mathcal{T}}_{\mathrm{V}}$ for $i \in \{1, 2\}$ (Induction Hyp.)

  We show $[\![\varepsilon(\Psi_1 \vee \Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}} = ||\Psi_1 \vee \Psi_2||^{\mathcal{T}}_{\mathrm{V}}$ in two parts.

  ($\subseteq$) Assume $t \in [\![\varepsilon(\Psi_1 \vee \Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}}$, then there is a simulation relation $R_1 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1 \vee \Psi_2)}$ for V such that $t R^{\mathrm{V}}_1 s$ for some $s \in S_1 \cup S_2$ by definition of construction. We define $R'_2 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1)}$ as $t R'_2 s$ if and only if $t R_1 s$ and $s \in S_{\varepsilon(\Psi_1)}$ and similarly $R''_2 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_2)}$ as $t R''_2 s$ if and only if $t R_1 s$ and $s \in S_{\varepsilon(\Psi_2)}$. These relations are well defined as $S_{\varepsilon(\Psi_1)}$ and $S_{\varepsilon(\Psi_2)}$ are disjoint sets. As the sets are disjoint, showing these relations are indeed simulation relations is trivial as the conditions are satisfied by $R_1$. Hence $t \in [\![\varepsilon(\Psi_1)]\!]^{\mathcal{T}}_{\mathrm{V}}$ or $t \in [\![\varepsilon(\Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}}$ and by (Induction Hypothesis) $t \in ||\Psi_1||^{\mathcal{T}}_{\mathrm{V}}$ or $t \in ||\Psi_2||^{\mathcal{T}}_{\mathrm{V}}$. Therefore, $t \in ||\Psi_1 \vee \Psi_2||^{\mathcal{T}}_{\mathrm{V}}$.

  ($\supseteq$) Assume $t \in [\![\varepsilon(\Psi_1]\!]^{\mathcal{T}}_{\mathrm{V}}$. Then there is a simulation relation $R'_2 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1)}$ for V such that $t R'^{\mathrm{V}}_2 s$ for some $s \in S_1$. We define $R_1 \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi_1 \vee \Psi_2)}$ as

$tR_1s$ if and only if $tR'_2s$. It is again trivial to show that $R_1$ is a simulation relation since it is identical to $R'_2$. Hence $t \in [\![\varepsilon(\Psi_1 \vee \Psi_2)]\!]^{\mathcal{T}}_{\mathrm{V}}$. The case for $t \in [\![\varepsilon(\Psi_2]\!]^{\mathcal{T}}_{\mathrm{V}}$ is similar.

- $\Phi \equiv \nu Z.\Psi_1$
  Dual to least fixed point case.

- $\Phi \equiv \mu Z.\Psi$
  Let $\varepsilon(\Psi) = ((S_{\varepsilon(\Psi)},\, A,\, \longrightarrow^{\Diamond}_{\mathcal{E}},\, \longrightarrow^{\Box}_{\mathcal{E}},\, c),\, S_1,\, \lambda)$ and $\varepsilon(\mu Z.\Psi)$= $((S_{\varepsilon(\mu Z.\Psi)},\, A,\, \longrightarrow^{\Diamond}_{\mathcal{E}'},\, \longrightarrow^{\Box}_{\mathcal{E}'},\, c'),\, S'_1,\, \lambda')$ be constructed as defined in Figure B.2. We will prove
  $$[\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}} = ||\mu Z.\Psi||^{\mathcal{T}}_{\mathrm{V}}$$
  in two steps.

  ($\subseteq$) We make use of ordinal approximants and the unfolding theorem for fixed point formulae.

  We show
  $$[\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}} \subseteq ||\mu Z.\Psi||^{\mathcal{T}}_{\mathrm{V}}$$
  Note first that
  $$||(\mu Z.\Psi)^{\kappa}||^{\mathcal{T}}_{\mathrm{V}} = \bigcup_{\beta<\kappa} ||(\mu Z.\Psi)^{\beta}||^{\mathcal{T}}_{\mathrm{V}}$$
  and so
  $$||(\mu Z.\Psi)^{\kappa}||^{\mathcal{T}}_{\mathrm{V}} = ||\Psi||^{\mathcal{T}}_{\mathrm{V}[\bigcup\limits_{\beta<\kappa}||(\mu Z.\Psi)^{\beta}||^{\mathcal{T}}_{\mathrm{V}}/Z]}$$
  By the Induction Hypothesis then, we replace the formula with its EMTS
  $$||(\mu Z.\Psi)^{\kappa}||^{\mathcal{T}}_{\mathrm{V}} = [\![\varepsilon(\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}[\bigcup\limits_{\beta<\kappa}||(\mu Z.\Psi)^{\beta}||^{\mathcal{T}}_{\mathrm{V}}/Z]}$$
  Assume $t \in [\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}$. Then there is a simulation relation $R_1 \in S_{\mathcal{T}} \times S_{\varepsilon(\mu Z.\Psi)}$ for V such that $tR_1\{s\}$ for some state state $s \in S_1$.

  Then there is a mapping $ord : S_{\mathcal{T}} \rightarrow Ord_{+}$ such that whenever $t'R_1q'$, $t' \xrightarrow{a}_{\mathcal{T}} t''$, $q' \xrightarrow{a}^{\Diamond}_{\mathcal{E}} q''$ and $t''R_1q''$, then $ord(t'') < ord(t')$ whenever $Z \in \lambda_1(\cup q'')$ and $ord(t'') \leq ord(t')$ otherwise.

  Define $S^{\kappa}_{\mathcal{T}} \overset{def}{=} \{t' \in S_{\mathcal{T}} \mid ord(t') \leq \kappa\}$ and $R^{\kappa}_1 \overset{def}{=} R_{1|_{S^{\kappa}_{\mathcal{T}}}}$

  We claim that $R^{\kappa}_1$ is a simulation relation for V. We show the following by induction on $ord(t')$.

  $$\forall t' \in S_{\mathcal{T}}.(\exists q' \in S_{\varepsilon(\mu Z.\Psi)}.q' \cap S_1 \neq \emptyset \wedge t'R^{ord(t')}_1 q') \implies t' \in ||(\mu Z.\Psi)^{ord(t')}||^{\mathcal{T}}_{\mathrm{V}}$$

Assume it holds for all $t'' \in S_{\mathcal{T}}$ such that $ord(t'') \leq \kappa$ by ordinal induction hypothesis. Assume $t' \in S_{\mathcal{T}}$ is such that $ord(t') = \kappa$, and there exists $q' \in S_{\varepsilon(\mu Z.\Psi)}.\exists q' \in S_{\varepsilon(\mu Z.\Psi)}.q' \cap S_1 \neq \emptyset \wedge t' R_1^{ord(t')} q')$. we show $t' \in [\![\varepsilon(\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}^\kappa}$ where $\mathrm{V}^\kappa = \mathrm{V}[\bigcup\limits_{\beta<\kappa} ||(\mu Z.\Psi)^\beta||^{\mathcal{T}}_{\mathrm{V}}/Z]$, and then by the original induction hypothesis and the unfolding of formula illustrated above we obtain $t' \in ||(\mu Z.\Psi)^{ord(t')}||^{\mathcal{T}}_{\mathrm{V}}$ since $\kappa = ord(t')$.

Define $R_2^\kappa \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi)}$ as $\{(t', s') \mid \exists q' \in S_{\varepsilon(\mu Z.\Psi)}.(s' \in real(q') \wedge t' R_1^\kappa q')\}$, where $real(q)$ is defined inductively as follows:

- $real(\{s_1\}) = \{s_1\}$
- Consider the state $q$. Let $Q$ be the set of states in $S_{\varepsilon(\mu Z.\Psi)}$ such that $\exists Q'.\ q'' \xrightarrow{a}{}^{\diamond}_{\mathcal{E}} Q'$ where $q \in Q'$ if and only if $q'' \in Q$. Then $real(q) \subseteq q$ is the set of derivatives of $\bigcup Q$ in $q$.

We show that $R_2^\kappa \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi)}$ is a simulation relation for $\mathrm{V}^\kappa$. Then, by the assumption on $t'$, we have $t' R_2^\kappa s'$ for some $s' \in S_1$, and hence $t' \in [\![\varepsilon(\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}^\kappa}$

($\supseteq$) We show

$$[\![\varepsilon(\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}[[\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}/Z]} \subseteq [\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}} (*)$$

Then by Induction Hypothesis

$$||\Psi||^{\mathcal{T}}_{\mathrm{V}[[\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}/Z]} \subseteq [\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}$$

The result hence holds since then $[\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}$ is a prefixed point of the function $\lambda S.||\Psi||^{\mathcal{T}}_{[S/X]}$, and since $||\mu Z.\Psi||^{\mathcal{T}}_{\mathrm{V}}$ is the least such prefixed point by the semantics of modal $\mu$-calculus.

Let $\mathrm{V}^* = \mathrm{V}[[\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}/Z]$. Proof for $[\![\varepsilon(\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}^*} \subseteq [\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}$ is as follows: We have to prove that if there exists an $s \in S_1$ such that $tR^{\mathrm{V}^*}s$ for some simulation relation $R \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\Psi)}$, then there exists a simulation relation $R' \subseteq S_{\mathcal{T}} \times S_{\varepsilon(\mu Z.\Psi)}$ and a state $\{q\} \in S'_1$ such that $tR'^{\mathrm{V}}\{q\}$.

We define $R'$ for $^{\mathrm{V}}$ using $R$ for $\mathrm{V}^*$ as the least relation that satisfies the following:

- If $tRs$ and $s \in S_1$, $tR'\{s\}$
- If $t'R'\{q\}$ and $t' \xrightarrow{a}_{\mathcal{T}} t''$
  since $t'Rq$ (INV. 1) there exists $q'$ such that $q' \in Q'$ for some $Q'$ where $q \xrightarrow{a}{}^{\diamond}_{\mathcal{E}} Q'$ and $t''Rq'$. Then,

1. if $Z \in \lambda(q')$, then we know by the definition of simulation that $t \in [\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}$. Then there exists a simulation relation $R''$ such that and for some $s'' \in S_1$ $t'' R''^{\mathrm{V}}\{s''\}$ by construction and the def. of denotation. Then $t'' R'\{q', s''\}$.
2. if $Z \notin \lambda(q')$, then $t'' R'^{\mathrm{V}}\{q'\}$.

– If $t' R'^{\mathrm{V}}\{q_1, \ldots, q_n\}$, $n > 1$ and $t' \xrightarrow{a}_{\mathcal{T}} t''$ since there exists
  - $*$ $q_i \notin S_1$ such that $t' R q_i$ (INV. 1a), there exists $q'$ such that $q' \in Q'$ for some $Q'$ where $q \overset{a}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}} Q'$ and $t'' R q'$.
  - $*$ $Q_1, \ldots, Q_m \in S_{(\varepsilon(\mu Z.\Psi))}$ such that $(\bigcup_{1 \le j \le m} Q_j) \cup q_i = \{q_1, \ldots, q_n\}$ and for $1 \le j \le m$, $t' R'^{\mathrm{V}}_j Q_j$ for some simulation relation $R'_j \subseteq S_{\mathcal{T}} \times S_{(\varepsilon(\mu Z.\Psi))}$. (INV 2.) There exists $Q'_1, \ldots, Q'_m$ such that for $1 \le j \le m$ $Q'_j \in S_{Qj}$ for some $S_{Qj}$ where $Q_j \overset{a}{\longrightarrow}{}^{\Diamond}_{\varepsilon(\mu Z.\Psi)} S_{Qj}$ and $t'' R'^{\mathrm{V}}_j Q'_j$.

1. If $Z \in \lambda(q')$, then we know by the definition of simulation that $t \in [\![\varepsilon(\mu Z.\Psi)]\!]^{\mathcal{T}}_{\mathrm{V}}$, then there exists a simulation relation $R$ such that and for some $s'' \in S_1$ $t'' R''^{\mathrm{V}}\{s''\}$ by construction and the def. of denotation. In this case $t'' R'(\bigcup_{1 \le j \le m} Q'_j \cup \{q'\} \cup \{s''\})$.
2. If $Z \notin \lambda(q')$, then $t'' R'(\bigcup_{1 \le j \le m} Q'_j \cup \{q'\})$.

INV 1a. is justified through an invariant of the construction namely that for every EMTS in the range of $\varepsilon$, the start states of the EMTS do not have incoming transitions.

That the resulting states are actually reachable in $\varepsilon(\mu Z.\Psi)$ is by the definition of $\varepsilon$.

The proof that $R'$ is a simulation for V is as follows:

Assume $tR'\{q_1, \ldots, q_n\}$ where $n > 1$, $a \in A$ and $X \in PropVar$, then by the invariants 1 and 2:

– there exists $q_i \in \{q_1, \ldots, q_n\}$ such that $q_i \notin S_1$ and $tRq_i$,

– $Q_1, \ldots, Q_m \in S_{(\varepsilon(\mu Z.\Psi))}$ such that $(\bigcup_{1 \le j \le m} Q_j) \cup q_i = \{q_1, \ldots, q_n\}$ and for $1 \le j \le m$, $t R'^{\mathrm{V}}_j Q_j$ for some simulation relation $R'_j \subseteq S_{\mathcal{T}} \times S_{(\varepsilon(\mu Z.\Psi))}$.

1. Straightforward.
2. If $\{q_1, \ldots, q_n\} \overset{a}{\longrightarrow}{}^{\Box}_{\varepsilon(\mu Z.\Psi)} S$, then by construction there exists a $q \in \{q_1, \ldots, q_n\}$ such that $q \overset{a}{\longrightarrow}{}^{\Box}_{\varepsilon(\Psi)} S'$ and $S = \partial_{\mathcal{P}}((\cup \partial^{\Diamond}_a(q_1), \ldots, S', \ldots, \cup \partial^{\Diamond}_a(q_n)), S_1, \lambda, Z)$.

- If $q = q_i$, then by the definition of simulation there exists $t \xrightarrow{a}_{\mathcal{T}} t'$ such that $t'Rq'_i$ for some $q'_i \in S'$. Again by the definition of simulation, if $t \xrightarrow{a}_{\mathcal{T}} t'$, then there should exist $Q'_1 \in \cup\partial_a^{\Diamond}(Q_1), \dots, Q'_{Q-1} \in \cup\partial_a^{\Diamond}(q_{i-1}), q'_{i+1} \in \cup\partial_a^{\Diamond}(q_{i+1}), \dots, q'_n \in \cup\partial_a^{\Diamond}(q_n)$
- If $q \in Q_j$ for some $j$, then let $Q_j = \{s_1, \dots, s_k\}$. By construction $Q_j \overset{\Box}{\xrightarrow{a}}_{\varepsilon(\mu Z.\Psi)} S'_j$ where $S = \partial_{\mathcal{P}}((\cup\partial_a^{\Diamond}(s_1), \dots, S', \dots, \cup\partial_a^{\Diamond}(s_k)), S_1, \lambda, Z)$.

3. If for the run $\rho_{\mu Z.\Psi} = \{q_1, \dots, q_n\} \xrightarrow{a_1}_{\mathcal{E}} S_1^{\mu} \xrightarrow{a_2}_{\mathcal{E}} S_2^{\mu} \xrightarrow{a_3}_{\mathcal{E}} \dots$ there exists $1 \le j \le k$ such that $max(inf(c_{\mu}(\rho_{\mu Z.\Psi})(j)))$ is odd then there does not exist an infinite run $\rho_t = t \xrightarrow{a_1}_{\mathcal{T}} t_1 \xrightarrow{a_2}_{\mathcal{T}} t_2 \xrightarrow{a_3}_{\mathcal{T}} \dots$ such that $t_i R' S_i^{\mu}$ for all $i \ge 0$.
4. If $X \in \lambda'(\{q_1, \dots, q_n\})$ then there is at least one member $q_j$ of this set for which $X \in \lambda(q_j)$ and $X \ne Z$ by construction. By the invariants 1 and 2, $tRq_i$ for some $q_i \in \{q_1, \dots, q_n\}$ and there exists a set $Q \subseteq \{q_1, \dots, q_n\}$ where $q_j \in Q$ and $tR''Q$ for some $R''$:

- If $q_j = q_i$ then $t \in \mathrm{V}^*(X)$ but since $\mathrm{V}(X) = \mathrm{V}^*(X)$ when $Z \ne X$, so $t \in \mathrm{V}(X)$, or
- Then by construction, $X \in \lambda'(Q)$, so $t \in \mathrm{V}(X)$.

□

## Correctness of Construction for Process Terms

**Proposition C.12.** *Let* fix $X.E$ *be a guarded term.* $E'[\text{fix } X.E/X]\rho_A\rho_0 \xrightarrow{a} E''[\text{fix } X.E/X]\rho_A\rho_0$ *if and only if* $E'[\mathbf{0}/X] \xrightarrow{a} E''[\mathbf{0}/X]$ *and* $E' \not\equiv X$.

*Proof.* This can be proved by induction on the structure of $E'$.
We will illustrate by giving one case, the others are similar. Suppose $E' \equiv a.F$. By transition rules of CCS $a.F[\mathit{fix}\ X.E/X] \xrightarrow{a}_{\mathcal{T}} F[\mathit{fix}\ X.E/X]$. Then $a.F[\mathbf{0}/X] \xrightarrow{a}_{\mathcal{T}} F[\mathbf{0}/X]$.

Suppose $E' \equiv F+G$. By transition rules of CCS either $(F+G)[\mathit{fix}\ X.E/X] \xrightarrow{a}_{\mathcal{T}} F'[\mathit{fix}\ X.E/X]$ or $(F+G)[\mathit{fix}\ X.E/X] \xrightarrow{a}_{\mathcal{T}} G'[\mathit{fix}\ X.E/X]$ and this is the case if $F[\mathit{fix}\ X.E/X] \xrightarrow{a}_{\mathcal{T}} F'[\mathit{fix}\ X.E/X]$ or $G[\mathit{fix}\ X.E/X] \xrightarrow{a}_{\mathcal{T}} G'[\mathit{fix}\ X.E/X]$ respectively. And since $G, F \not\equiv X$, we can use the induction hypothesis. Suppose $E' \equiv F \mid G$. This is the case if $F$ and $G$ does not contain occurrences of $X$, because of the assumption on the syntax of the algebra terms in the theorem. Then $(F \mid G)[\mathit{fix}\ X.E/X] \equiv (F \mid G)[\mathbf{0}/X]$. So the claim holds trivially. □

**Lemma C.13.** *Let* $\Gamma \triangleright E$ *be a guarded OTA without composition and let* $\varepsilon(\Gamma \triangleright E) = (\mathcal{E},\ S,\ \lambda)$. *Then* $[\![S]\!]_{\mathcal{U}}$ *is equal to the set* $[\![\Gamma \triangleright E]\!]_{\rho_0}$ *up to bisimulation denoted* $[\![S]\!]_{\mathcal{U}} \simeq [\![\Gamma \triangleright E]\!]_{\rho_0}$ *where* $\rho_0$ *maps each recursion process variable* $X$ *to* $\mathbf{0}$.

*Proof.* The proof proceeds on the structure of $E$. Note that in the proofs below $t \approx t'$ denotes that $t$ is bisimilar to $t'$. Let $\varepsilon(\Gamma \rhd E_1)$=(($S_{\mathcal{E}_1}$, $A$, $\longrightarrow^{\Diamond}_{\mathcal{E}_1}$, $\longrightarrow^{\Box}_{\mathcal{E}_1}$, $W_1$), $S_1$, $\lambda_1$) and $\varepsilon(\Gamma \rhd E_2)$= (($S_{\mathcal{E}_2}$, $A$, $\longrightarrow^{\Diamond}_{\mathcal{E}_2}$, $\longrightarrow^{\Box}_{\mathcal{E}_2}$, $W_2$), $S_2$, $\lambda_2$) be constructed as defined in Figure B.3. In the cases below, Induction Hypothesis stands for induction hypothesis and TR for transition rules of CCS.

- $E \equiv \mathbf{0}$

  $t \preceq S$
  $\iff$ $t \preceq s$ (Construction)
  $\iff$ for no $a \in A$ exists $t'$ such that $t \xrightarrow{a}_{\mathcal{T}} t'$. (Def. C.10)
  $\iff$ $t \approx \mathbf{0}$ (Def. 3.5), $\{\mathbf{0}\} = [\![\Gamma \rhd \mathbf{0}]\!]_{\rho_0}$

- $E \equiv X$

  1. $X \in AssProcVar$
     $[\![\Gamma \rhd X]\!]_{\rho_0}$=$\varepsilon(\bigwedge_{X:\Psi\,\in\,\Gamma} \Psi)$ and we can directly use Theorem 5.1. Note that since the logical formulae $\Psi$ are closed, the returned labeling function labels all states with the empty set.
  2. $X \in RecProcVar$
     $[\![\Gamma \rhd X]\!]_{\rho_0}$=$\rho_0(X)$=$\{\mathbf{0}\}$ by Def. 3.5.

- $E \equiv a.E_1$

  $t \preceq S$
  $\iff$ $t \preceq s_{new}$ (Construction)
  $\iff$ $\exists t'.t \xrightarrow{a}_{\mathcal{T}} t'$ where $t' \preceq S_1$ and
  $\forall k \in A$ where $k \neq a$ there exists no $t''$ s.t. $t \xrightarrow{k}_{\mathcal{T}} t''$ (Def. C.10)
  $\iff$ $\exists t'.t \xrightarrow{a}_{\mathcal{T}} t'$ where $t' \approx u'$ where $u' \in [\![\Gamma \rhd E_1]\!]_{\rho_0}$ and $\forall k \in A$ where $k \neq a$ there exists no $t''$ s.t. $t \xrightarrow{k}_{\mathcal{T}} t''$ (Induction Hyp.)
  $\iff$ $\exists u'.t \approx a.u'$ and $u' \in [\![\Gamma \rhd E_1]\!]_{\rho_0}$
  $\iff$ $\exists u.t \approx u$ and $u \in [\![\Gamma \rhd a.E_1]\!]_{\rho_0}$ (Def. 3.5)

- $E \equiv E_1 + E_2$

  $t \preceq S$
  $\iff$ $\exists s_1 \in S_{v_1}, r_1 \in S_{v_2}.\ t \preceq (s_1, r_1)$
  $\iff$ $\exists t_1, u_1, s_1 \in S_{v_1}, r_1 \in S_{v_2}.t \approx t_1 + u_1 \wedge t_1 \preceq s_1$
  $\wedge u_1 \preceq r_1$ (See below)
  $\iff$ $\exists t_1, u_1.t \approx t_1 + u_1 \wedge (\exists v_1.t_1 \approx v_1 \wedge v_1 \in [\![\Gamma \rhd E_1]\!]_{\rho_0})$
  $\wedge(\exists v_2.u_1 \approx v_2 \wedge v_2 \in [\![\Gamma \rhd E_2]\!]_{\rho_0})$ (Induction Hyp.)
  $\iff$ $\exists v_1, v_2.t \approx v_1 + v_2 \wedge v_1 \in [\![\Gamma \rhd E_1]\!]_{\rho_0} \wedge v_2 \in [\![\Gamma \rhd E_2]\!]$
  $\iff$ $\exists v_1, v_2.t \approx v_1 + v_2 \wedge v_1 + v_2 \in [\![\Gamma \rhd E_1 + E_2]\!]_{\rho_0}$
  $\iff$ $\exists v.t \approx v \wedge v \in [\![\Gamma \rhd E_1 + E_2]\!]_{\rho_0}$ (Def. 3.5)

  The second equivalence is established by separate proofs of the two directions:

($\Rightarrow$)Consider some $t$ such that $t \preceq (s_1, r_1)$ where $s_1 \in S_{\upsilon_1}$ and $r_1 \in S_{\upsilon_2}$. By the Expansion Theorem and by our construction definition, we take $t \approx (a_0.t_0 + \ldots + a_k.t_k) + (a_{k+1}.t_{k+1} + \ldots + a_n.t_n)$ for some processes $t_i \in \mathcal{U}$ and where for each $a_i.t_i$ in this sum, $\exists S'.(s_1, r_1) \xrightarrow{a_i}_{\mathcal{E}}^{\Diamond} S'$ and $t_i \preceq S'$ where $S' \subseteq S_{\mathcal{E}_1}$ if $0 < i \leq k$, and $S' \subseteq S_{\mathcal{E}_2}$ if $k < i \leq n$ by the definition of simulation. The arguments for $(a_0.t_0 + \ldots + a_k.t_k) \preceq S_{\upsilon_1}$ and $(a_{k+1}.t_{k+1} + \ldots + a_n.t_n) \preceq S_{\upsilon_2}$ are then trivial.

($\Leftarrow$) By Induction Hypothesis there exists simulation relations $R_1$ and $R_2$ between elements of $[\![\Gamma \rhd E_1]\!]_{\rho_0}$ and $\mathcal{E}_{\upsilon_1}$, $[\![\Gamma \rhd E_2]\!]_{\rho_0}$ and $\mathcal{E}_{\upsilon_2}$ respectively. We define the relation $R' \subseteq S_{\mathcal{U}} \times S_{\mathcal{E}}$ using $R_1$ and $R_2$ as follows:

$$tR'q \quad \triangleq \begin{cases} t_1R_1s_1 \wedge t_2R_2r_1 & \text{if } t = t_1 + t_2 \text{ and } q = (s, r) \text{ for} \\ & s_1 \in S_{\upsilon_1}, r_1 \in S_{\upsilon_2} \\ tR_1q & \text{if } q \in S_{\mathcal{E}_1} \\ tR_2q & \text{if } q \in S_{\mathcal{E}_2} \end{cases}$$

We show that $R'$ is a simulation relation.

Assume $tR'q$. The argument is obvious if $q \notin S_{\upsilon_1} \times S_{\upsilon_2}$. So we assume $q = (s_1, r_1)$ for some $s_1 \in S_{\upsilon_1}$ and some $r_1 \in S_{\upsilon_2}$.

$$\begin{array}{lll}
\text{(i)(a)} & \exists t'.(t_1 + u_1) \xrightarrow{a}_{\mathcal{T}} t' & \\
& \Longrightarrow & \exists t'.(t_1 \xrightarrow{a}_{\mathcal{T}} t' \vee u_1 \xrightarrow{a}_{\mathcal{T}} t') \text{ (TR)} \\
& \Longrightarrow & \exists S'.(s_1 \xrightarrow{a}_{\mathcal{E}}^{\Diamond} S' \wedge t'R_1S') \text{ or} \\
& & \exists S'.(r_1 \xrightarrow{a}_{\mathcal{E}}^{\Diamond} S' \wedge t'R_2S') \text{ (Def C.10)} \\
& \Longrightarrow & \exists S'.((s_1, r_1) \xrightarrow{a}_{\mathcal{E}}^{\Diamond} S') \wedge t'RS' \text{ (Construction)}
\end{array}$$

$$\begin{array}{lll}
\text{(i)(b)} & \exists S'.(s_1, r_1) \xrightarrow{a}_{\mathcal{E}}^{\Box} S' & \\
& \Longrightarrow & \exists S'.s_1 \xrightarrow{a}_{\mathcal{E}_1}^{\Box} S' \text{ or } \exists S'.r_1 \xrightarrow{a}_{\mathcal{E}_2}^{\Box} S' \text{ (Construction)} \\
& \Longrightarrow & \exists t'.t_1 \xrightarrow{a}_{\mathcal{T}} t' \wedge t'R_1S' \text{ or} \\
& & \exists t'.u_1 \xrightarrow{a}_{\mathcal{T}} t' \wedge t'R_2S' \text{ (Def. 4.2)} \\
& \Longrightarrow & t_1 + u_1 \xrightarrow{a}_{\mathcal{T}} t' \wedge t'RS'
\end{array}$$

(ii) We show that if $(s_1, r_1) \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \ldots \in W$, then $\rho_t = t \xrightarrow{a_1}_{\mathcal{T}} t_1 \xrightarrow{a_2}_{\mathcal{T}} t_2 \ldots$ where $t_iRq_i$ for all $i > 1$ is not a run of $t$.

Assume such a $\rho_t$ exists, and let $t = u_1 + u_2$ where $u_1R_1s_1$ and $u_2R_2r_1$. By the construction either $s_1 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \ldots \in W_{\upsilon_1}$ or $r_1 \xrightarrow{a_0} q_1 \xrightarrow{a_2} q_2 \ldots \in W_{\upsilon_2}$. Assume it is the first case. Then $t_iRq_i$ where $q_i \in S_{\mathcal{E}_1}$ for all $i > 0$. For this to be the case, $t_iR_1q_i$ for all $i > 0$. Then $u_1 \xrightarrow{a_0}_{\mathcal{T}} t_1 \xrightarrow{a_2}_{\mathcal{T}} t_2 \ldots$ is an infinite run where $t_iR_1q_i$ for all $i > 0$ but such an infinite run can not exist by the definition of simulation.

- $E \equiv \mathit{fix}\ X.E_1$

$$\begin{array}{ll} \multicolumn{2}{l}{v \in [\![\Gamma \rhd \mathit{fix}\ X.E_1]\!]_{\rho_0}} \\ \iff & v \equiv (\mathit{fix}\ X.E_1\rho_A\rho_0) \text{ for some } \rho_A \text{ (Def. 3.5)} \\ \iff & v \equiv X[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 \text{ for some } \rho_A \\ \iff & \exists t.t \equiv E_1[0/X]\rho_A\rho_0 \wedge t \in [\![\Gamma \rhd E_1]\!]_{\rho_0} \text{ (Def. 3.5)} \\ \iff & \exists s_1 \in S_1.t \equiv E_1[0/X]\rho_A\rho_0 \wedge t \preceq s_1 \text{ (Def. 4.2)} \\ \iff & \exists s_1 \in S_1.v \preceq s_1 \text{ (See below)} \end{array}$$

We establish the last equivalence as follows. By induction hypothesis, there exists a simulation relation $R_1$ between processes in $[\![\Gamma \rhd E_1]\!]_{\rho_0}$ and states of $\mathcal{E}_1$. Using this relation $R_1$, we define $R \subseteq S_{\mathcal{U}} \times S_{\mathcal{E}}$ as follows:

$$(E_1'[\mathit{fix}\ X.E_1/X])\rho_A\rho_0 Rs \quad \triangleq \begin{cases} E_1'[\mathbf{0}/X]\rho_A\rho_0\ R_1\ s & \text{if } E_1' \not\equiv X \\ E_1[\mathbf{0}/X]\rho_A\rho_0\ R_1\ s & \text{if } E_1' \equiv X \end{cases}$$

We show that $R$ is a simulation relation.
Assume $E_1'[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 Rs$
(i)(a) First let us take the case where $E_1' \equiv X$, then
$E_1'[\mathit{fix}\ X.E/X]\rho_A\rho_0 \equiv (\mathit{fix}\ X.E_1)\rho_A\rho_0$.

$$\begin{array}{ll} \multicolumn{2}{l}{(\mathit{fix}\ X.E_1)\rho_A\rho_0 t \xrightarrow{a}_{\mathcal{T}} E_1''[\mathit{fix}\ X.E_1/X]\rho_A\rho_0} \\ \iff & E_1[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 t \xrightarrow{a} E_1''[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 \text{ (TR)} \\ \iff & E_1[0/X]\rho_A\rho_0 \xrightarrow{a}_{\mathcal{T}} E_1''[0/X]\rho_A\rho_0 \text{ } (\mathit{fix}\ X.E_1 \text{ is guarded, Prop. C.12)} \\ \iff & E_1[0/X]\rho_A\rho_0 \in [\![\Gamma \rhd E_1]\!]_{\rho_0} \\ \iff & s \in S_1 \wedge E_1[0/X]\rho_A\rho_0 R_1 s \text{ (Def. 3.5)} \\ \iff & \exists S'.s \xrightarrow{a}{}_{\mathcal{E}}^{\diamond} S' \text{ where } E_1''[0/X]\rho_A\rho_0 R_1 s' \text{ for some } s' \text{ in } S' \text{ (Def. 4.2)} \\ \iff & E_1''[\mathit{fix}\ X.E/X]\rho_A\rho_0 Rs' \end{array}$$

Argument for $E_1' \not\equiv X$ is similar.

b)First let us take the case where $s \in S_1$.

$$\begin{array}{ll} \multicolumn{2}{l}{s \xrightarrow{a}{}_{\mathcal{E}}^{\Box} S'} \\ \iff & E_1[0/X]\rho_A\rho_0 \xrightarrow{a}_{\mathcal{T}} E_1''[0/X]\rho_A\rho_0 \\ & \text{and } E_1''[0/X]\rho_A\rho_0 R_1 s' \text{ for some } s' \in S' \text{(Def. 4.2)} \\ \iff & E_1[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 \xrightarrow{a}_{\mathcal{T}} E_1''[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 \\ & \text{and } E_1''[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 R_1 s' \text{ for some } s' \in S' \text{(Prop. C.12, } E_1 \not\equiv X) \\ \iff & \mathit{fix}\ X.E_1\rho_A\rho_0 \xrightarrow{a} E_1''[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 \text{ and } E_1''[\mathit{fix}\ X.E_1/X]\rho_A\rho_0 Rs' \\ & \text{for some } s' \in S' \text{ (TR)} \end{array}$$

Argument for $s \notin S_1$ is similar.

2) Take some prohibited run of $s_1$, $s_1 \xrightarrow{a_1}_{\mathcal{E}} s_2 \ldots \xrightarrow{a_{n-1}}_{\mathcal{E}} s_n).\rho_{s_n}$ where $n \geq 1$ and $\rho_{s_n} \in W_1$. For all mentioned states $s_m$ in this run, where $m > n$, $s_m \notin S_1$. Suppose there is an infinite run of $t = E'[\mathit{fix}\ X.E/X]\rho_A\rho_0$: $(t \xrightarrow{a_1}_{\mathcal{T}} t_2 \ldots \xrightarrow{a_{n-1}}_{\mathcal{T}} t_n).\rho_{t_n}$, such that for all $i > 1$, $t_i R s_i$.

Assume $s_n \in S_1$. We have $E[\mathbf{0}/X]\rho_A\rho_0 R' s_n$ and so
$t_n = X[\mathit{fix}\ X.E/X]\rho_A\rho_0 R s_n$ by Induction Hypothesis. Then, the infinite run $\rho_{t_n}$ has the form $X[\mathit{fix}\ X.E/X]\rho_A\rho_0 \xrightarrow{a_n}_{\mathcal{T}} E_{n+1}[\mathit{fix}\ X.E/X]\rho_A\rho_0 \xrightarrow{a_{n+1}}_{\mathcal{T}} \ldots$

where for all $i > 1$, $E_{n+i}[\mathit{fix}\, X.E/X]\rho_A\rho_0 R s_{n+i}$ and $s_{n+i} \notin S_1$. By the way we defined $R$, this is possible if for all $i > 1$, $E_{n+i}[\mathbf{0}/X]\rho_A\rho_0 R' s_{n+i}$ and $E_{n+1} \not\equiv E$. Then we can construct a run of $E[\mathbf{0}/X]\rho_A\rho_0$, if we replace the *fix* expression substitution at each process with a $\mathbf{0}$ substitution: $E[\mathbf{0}/X]\rho_A\rho_0 \xrightarrow{a_n}_{\mathcal{T}} E_{n+1}[\mathbf{0}/X]\rho_A\rho_0 \xrightarrow{a_{n+1}}_{\mathcal{T}} \ldots$ where $E[\mathbf{0}/X]\rho_A\rho_0 R' s_n$ and for all $i > 1$, $E_{n+i}[\mathbf{0}/X]\rho_A\rho_0 R' s_{n+i}$. This run is a legal run of $E[\mathbf{0}/X]\rho_A\rho_0$ in $\mathcal{E}_1$, since the transitions between these terms exist also in $\mathcal{E}_1$ (Proposition. C.12, TR) This results in an infinite run of $E[\mathbf{0}/X]\rho_A\rho_0$ which is simulated by $\rho_{s_n}$, but this is impossible by Def. 4.2 since $\rho_{s_n} \in W_1$. Hence no such infinite run simulated by $\rho_{s_1}$ is possible. The case for $s_n \notin S_1$ is similar.

□

**Lemma C.14.** *Let $\mathcal{T}$ be a transition-closed LTS, $\Gamma \rhd E_1 \parallel E_2$ be a guarded linear OTA where every recursion process variable in the scope of parallel composition is bound by a* fix *operator in the same scope, and let $\varepsilon(\Gamma \rhd E) = (\mathcal{E}, S, \lambda)$. Then the set $[\![S]\!]_{\mathcal{T}}$ includes $[\![\Gamma \rhd E_1 \parallel E_2]\!]_{\rho_0}$ up to bisimulation.*

*Proof.* Let $\varepsilon(\Gamma \rhd E_1)$=$((S_{\mathcal{E}_1}, A, \longrightarrow^{\Diamond}_{\mathcal{E}_1}, \longrightarrow^{\Box}_{\mathcal{E}_1}, W_1), S_1, \lambda_1)$ and $\varepsilon(\Gamma \rhd E_2)$= $((S_{\mathcal{E}_2}, A, \longrightarrow^{\Diamond}_{\mathcal{E}_2}, \longrightarrow^{\Box}_{\mathcal{E}_2}, W_2), S_2, \lambda_2)$ be constructed as defined in Figure B.3.

$v \in [\![\Gamma \rhd E_1 \| E_2]\!]_{\rho_0}$
$\iff \quad \exists t_1, u_1.\, v \approx t_1 \| u_1 \wedge\ t_1 \in [\![\Gamma \rhd E_1]\!]_{\rho_0} \wedge u_1 \in [\![\Gamma \rhd E_2]\!]_{\rho_0}$ (linearity)
$\implies \quad \exists t_1, u_1, s_1, r_1.\, v \approx t_1 \| u_1 \wedge\ s_1 \in S_1 \wedge r_1 \in S_2 \wedge$
$\qquad t_1 \preceq s_1 \wedge u_1 \preceq r_1$ (Induction Hyp.)
$\implies \quad \exists s_1, r_1, x_1.\ \ v \preceq (s_1, r_1, x_1)$(See below)

In order to show the last implication we define a relation $R \subseteq S_{\mathcal{U}} \times S_{\mathcal{E}}$ (see below). By Induction Hypothesis, simulation relations $R_1$ and $R_2$ exist between elements of $[\![\Gamma \rhd E_1]\!]_{\rho_0}$, $[\![\Gamma \rhd E_2]\!]_{\rho_0}$ and states of $\mathcal{E}_1$, $\mathcal{E}_2$ respectively.

$R$ is the least relation that satisfies the following:

- if $tR_1s_1$ and $uR_2r_1$ where $s_1 \in S_1$ and $r_1 \in S_2$, then $(t\|u)R(s_1, r_1, x)$ for all $x \in \{1, 2\}$
- if $tR_1s$ and $uR_2r$ and $t \xrightarrow{a}_{\mathcal{T}} t'$ and $t'R_1s'$, then $(t'\|u)R(s', r, 1)$
- if $tR_1s$ and $uR_2r$ and $u \xrightarrow{a}_{\mathcal{T}} u'$ and $u'R_2r'$, then $(t\|u')R(s, r', 2)$.

We claim that the relation $R$ is a simulation relation.
Assume $(t\|u)\ R\ (s, r, x)$ for some $x \in \{1, 2\}$.

1. Whenever $a \in A$,
   a) If $(t\|u) \xrightarrow{a}_{\mathcal{T}} (t'\|u')$, then by TR either $t \xrightarrow{a}_{\mathcal{T}} t'$ and $u' = u$ or $u \xrightarrow{a}_{\mathcal{T}} u'$ and $t' = t$. In the first case, $tR_1s$ by assumption. Then, by the Def. 4.2 there is a $S'_1 \subseteq S_{\mathcal{E}_1}$ such that $s \overset{a}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}_1} S'_1$ and $t'R_1s'$ for some $s' \in S'_1$. So $(s, r) \overset{a}{\longrightarrow}{}^{\Diamond}_{\mathcal{E}} S$, where $S = \{(s', r, 1) \| s'_1 \in S'_1\}$. Again by assumption $uR_2r$, then $(t'\|u)\ R\ (s', r, 1)$. The second case is similar.

b) If $(s, r, x) \xrightarrow{a}^{\Box}_{\mathcal{E}} S$, then either $s \xrightarrow{a}^{\Box}_{\mathcal{E}_1} S'_1$ and $S = \{(s', r, 1) \| s'_1 \in S'_1)\}$ or $r_1 \xrightarrow{a}^{\Box}_{\mathcal{E}_2} S'_2$ and $S = \{(s, r', 2) \| r' \in S'_2\}$. In the first case, $tR_1s$ by assumption. Then, Def. 4.2 there is $t'$ such that $t \xrightarrow{a}_{\mathcal{T}} t'$ and $t'R_1s'$ for some $s' \in S'_1$. Then by TR, $(t \mid u) \xrightarrow{a}_{\mathcal{T}} (t' \| u)$. Again by assumption $uR_2r$. Then $(t' \| u)\ R\ (s', r, 1)$. The second case is similar.

2. If $\rho_{(s,r,x)} = (s, r, x) \xrightarrow{a_1}_{\mathcal{E}} (s_1, r_1, x_1) \xrightarrow{a_2}_{\mathcal{E}} (s_2, r_2, , x_2) \xrightarrow{a_3}_{\mathcal{E}} \dots$ is in $W$, then no infinite run $\rho_{(t\|u)} = t\|u \xrightarrow{a_1}_{\mathcal{E}} t_1\|u_1 \xrightarrow{a_2}_{\mathcal{E}} t_2\|u_2 \xrightarrow{a_3}_{\mathcal{E}} \dots$ such that $(t_i\|u_i)$ $R$ $(s_i, r_i, x_i)$ for all $i \geq 1$ can exist. Suppose that such a run of $(t\|u)$ exists. This run would clearly be an interleaving of runs of $t$ and $u$, where at least one of these runs are infinite. Since $\rho_{(s,r,x)}$ is prohibited and $S_{\varepsilon(\Gamma \rhd E_1\|E_2)}$ is finite, there exists an infinitely occurring state $(s', r', x')$ in this run such that for some $1 \leq j \leq k$, the color entry of this state $c(s', r', x')(j)$ is odd and larger than the other infinitely occurring integers in the $j^th$ entry of $c(\rho_{(s,r,x)})$.

   If $1 \leq j \leq k_1$, then we know that the infinitely occurring state $(s', r', x')$ is a state where the last transition was performed by the first component, so $x = 1$. By just selecting from the run $\rho_{(s,r,x)}$ those transitions which are followed by some state labeled with 1 and the first component of these states, we can extract a run $\rho_s$ of the first component. By a similar selection of the first component from the same positions of $\rho_{(t\|u)}$, we can build an infinite run of $t$. By our assumption these two runs follow each other, although $\rho_s$ is in $W$, $\rho_t$ is not. But again by assumption $tR_1s$, so we reach a contradiction. Same argument then applies to $u$ if $k_1 < j \leq k_2 + k_1$.

□

**Theorem 5.2(B.8)** See page 27.

*Proof.* This is similar to the proof of Lemma C.13 with the additional case of the parallel composition.

Let $\varepsilon(\Gamma \rhd E)$=$((S_{\mathcal{E}_1},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}_1},\ \longrightarrow^{\Box}_{\mathcal{E}_1},\ W_1),\ S_1,\ \lambda_1)$ and $\varepsilon(\Gamma \rhd t)$= $((S_{\mathcal{E}_2},\ A,\ \longrightarrow^{\Diamond}_{\mathcal{E}_2},\ \longrightarrow^{\Box}_{\mathcal{E}_2},\ W_2),\ S_2,\ \lambda_2)$ be constructed as defined in Figure B.3. The *may* and *must* transitions of $\varepsilon(\Gamma \rhd t)$ coincide, and all colors are 0 since no maximal model construction is done for this case.

$v \preceq S$

$\iff$ $\exists s_1 \in S_1, r_1 \in S_2, x_1 \in \{1, 2\}.\ \ v \preceq (s_1, r_1, x_1)$ (Construction)

$\iff$ $\exists t_1, u_1, s_1 \in S_1, r_1 \in S_2.\ v \approx t_1\|u_1 \wedge t_1 \preceq s_1 \wedge u_1 \preceq r_1$ (See below)

$\iff$ $\exists t_1, u_1.\ v \approx t_1\|u_1 \wedge\ t_1 \in [\![\Gamma \rhd E]\!]_{\rho_0} \wedge$
$u_1 \in [\![\Gamma \rhd t]\!]_{\rho_0}$ (Induction Hypothesis)

$\iff$ $v \in [\![\Gamma \rhd E\|t]\!]_{\rho_0}$

We show the second equality in two directions:

( $\Longrightarrow$ ) For this case, $t_1$ and $u_1$ can be constructed inductively, using the state space of $(s_1, r_1, x_1)$. The process $u_1$ is to be chosen bisimilar to $t$. At each step we know which component will make a transition through the last entry of the tuple, $x$.

($\Leftarrow$) Corollary of Lemma C.14. $\square$

**Theorem 5.3(B.9)** See page 27.

*Proof.* This is a direct corollary of Lemma C.13 and Lemma C.14. $\square$

# Bibliography

[1] I. Aktug and D. Gurov. Verification of open systems based on explicit state space representation. Technical report, ICT, KTH, `http://www.nada.kth.se/∽irem/sefros/techrep05.ps`, August 2005.

[2] I. Aktug and D. Gurov. Verification of open systems based on explicit state space representation. In *AVIS'05: Proceedings of Automated Verification of Infinite Systems*, To appear, 2005.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002. ISSN 0004-5411.

[4] H. R. Andersen. Partial model checking. In *Proceedings of The 10th Annual IEEE Symposium on Logic in Computer Science*, pages 398–407, 1995.

[5] O. Bernholtz and O. Grumberg. Branching time temporal logic and amorphous tree automata. In *CONCUR '93: Proceedings of the 4th International Conference on Concurrency Theory*, volume 715, pages 262–277, London, UK, 1993. Springer-Verlag.

[6] G. Boudol and K. G. Larsen. Graphical versus logical specifications. In André Arnold, editor, *CAAP '90, 15th Colloquium on Trees in Algebra and Programming, Copenhagen, Denmark, May 15-18, 1990, Proceedings*, volume 431 of *Lecture Notes in Computer Science*, pages 57–71. Springer-Verlag, 1990. ISBN 3-540-52590-4.

[7] J.C. Bradfield and C.P. Stirling. Local model checking for infinite state spaces. *Theoretical Computer Science*, 96:157–174, 1992.

[8] D. Bustan and O. Grumberg. Applicability of fair simulation. In *TACAS '02: Proceedings of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 2280 of *Lecture Notes in Computer Science*, pages 401–414. Springer-Verlag, 2002. ISBN 3-540-43419-4.

[9] S. Christensen. *Decidability and Decomposition in Process Algebras.* PhD thesis, University of Edinburgh, 1993.

[10] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement. In *Computer Aided Verification*, volume 1855 of *Lecture Notes in Computer Science*, pages 154–169. Springer-Verlag, 2000.

[11] E.M. Clarke, O. Grumberg, and D. E. Long. Model checking and abstraction. *ACM-TOPLAS*, 16(5):1512–1542, 1994. ISSN 0164-0925.

[12] R. Cleaveland, J. Parrow, and B. Steffen. The concurrency workbench. In *Proceedings of the international workshop on Automatic verification methods for finite state systems*, volume ??, pages 24–37, New York, NY, USA, 1990. Springer-Verlag New York, Inc. ISBN 0-387-52148-8.

[13] M. Dam. Fixed points of Büchi automata. In *FSTTCS '92: Proceedings of 12th Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 652 of *Lecture Notes in Computer Science*, pages 39–50, 1992.

[14] M. Dam, L.-Å. Fredlund, and D. Gurov. Toward parametric verification of open distributed systems. In H. Langmaack, A. Pnueli, and W.-P. de Roever, editors, *Compositionality: the Significant Difference*, volume 1536 of *Lecture Notes in Computer Science*, pages 150–185. Springer-Verlag, 1998.

[15] M. Dam and D. Gurov. Compositional verification of CCS processes. In *PSI '99: Proceedings of the Third International Andrei Ershov Memorial Conference on Perspectives of System Informatics*, pages 247–256. Springer-Verlag, 2000. ISBN 3-540-67102-1.

[16] Mads Dam. Proving properties of dynamic process networks. *Information and Computation*, 140(2):95–114, 1998.

[17] E. A. Emerson and C. S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *Proceedings of 32nd Annual Symposium on Foundations of Computer Science*, IEEE, pages 368–377. Computer Society Press, 1991.

[18] P. Godefroid, M. Huth, and R. Jagadeesan. Abstraction-based model checking using modal transition systems. In *CONCUR '01: Proceedings of the 12th International Conference on Concurrency Theory*, volume 2154 of *Lecture Notes in Computer Science*, pages 426–440, 2001.

[19] O. Grumberg and D. Long. Model checking and modular verification. *ACM Transactions on Programming Languages and Systems*, 16(3):843–871, 1994.

[20] O. Grumberg and S. Shoham. Monotonic abstraction-refinement for CTL. In *TACAS'04: Proceedings of the 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 2988 of *Lecture Notes in Computer Science*, pages 546–560. Springer-Verlag, 2004.

[21] D. Harel and A. Pnueli. On the development of reactive systems. 13:477–498, 1985.

[22] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the Association for Computing Machinery*, pages 137–161, 1985.

[23] M. Huth, R. Jagadeesan, and D. A. Schmidt. Modal transition systems: A foundation for three-valued program analysis. In *ESOP '01: Proceedings of the 10th European Symposium on Programming Languages and Systems*, volume 2028, pages 155–169, London, UK, 2001. Springer-Verlag. ISBN 3-540-41862-8.

[24] R. Kaivola. On modal mu-calculus and Büchi tree automata. *Information Processing Letters*, 54(1):17–22, 1995.

[25] D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[26] O. Kupferman and M. Vardi. An automata-theoretic approach to modular model checking. *ACM Transactions on Programming Languages and Systems*, 22(1):87–128, 2000.

[27] O. Kupferman and M. Y. Vardi. Robust satisfaction. In *CONCUR '99: Proceedings of the 10th International Conference on Concurrency Theory*, volume 1664 of *LNCS*, pages 383–398, London, UK, 1999. Springer-Verlag. ISBN 3-540-66425-4.

[28] O. Kupferman, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *Journal of ACM*, 47(2):312–360, 2000. ISSN 0004-5411.

[29] K. G. Larsen. Modal specifications. *Automatic Verification Methods for Finite State Systems*, pages 232–246, 1989.

[30] F. Martinelli. Analysis of security protocols as open systems. *Theor. Comput. Sci.*, 290(1):1057–1106, 2003. ISSN 0304-3975.

[31] F. Martinelli. Analysis of security protocols as open systems. *Theoretical Computer Science*, 290(1):1057–1106, 2003. ISSN 0304-3975.

[32] A. W. Mostowski. Regular expressions for infinite trees and a standard form of automata. *Computation Theory*, 208:pages 157–168, 1984.

[33] D. E. Muller and P. E. Schupp. Alternating automata on infinite objects, determinacy and rabin's theorem. In *Automata on Infinite Words, Ecole de Printemps d'Informatique Théorique,*, volume 192, pages 100–107, London, UK, 1985. Springer-Verlag. ISBN 3-540-15641-0.

[34] A. Pnueli. In transition for global to modular temporal reasoning about programs. In K.R. Apt, editor, *Logics and Models of Concurrent Systems*, volume 13 of *NATO ASI Series*. Springer, 1984.

[35] C. Sprenger, D. Gurov, and M. Huisman. Compositional verification for secure loading of smart card applets. In C. Heitmeyer and J.-P. Talpin, editors, *Proc. MEMOCODE'04*, pages 211–222. IEEE, 2004.

[36] C. Stirling. *Modal and Temporal Properties of Processes.* Texts in Computer Science. Springer-Verlag, 2001. ISBN 0-387-98717-7.

[37] W. Thomas. *Handbook of Theoretical Computer Science (vol. B): Formal Models and Semantics*, chapter Automata on Infinite Objects, pages 133–191. MIT Press, Cambridge, MA, USA, 1990.