Clocks, Dice and Processes

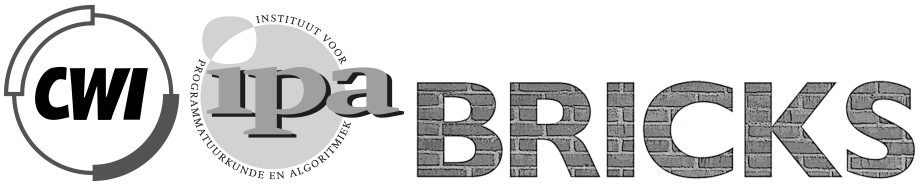
Taolue Chen

ISBN 978-90-8659-339-2
IPA dissertation series 2009-17

Typeset by LaTeX2$_\varepsilon$
Printed by Ponsen & Looijen B.V., Wageningen
Cover designed by Tingting Han, images © 2009 Creative kids

Promotion committee:

Prof. dr. W.J. Fokkink	Vrije Universiteit Amsterdam
Prof. dr. J.C. van de Pol (promotors)	Universiteit of Twente
Prof. dr. L. Aceto	Reykjavík University
Prof. dr. J.A. Bergstra	Universiteit van Amsterdam
Prof. dr. J.-P. Katoen	RWTH Aachen University
Prof. dr. J.W. Klop	Vrije Universiteit Amsterdam
Dr. S.P. Luttik	Technische Universiteit Eindhoven
Dr. M.I.A. Stoelinga	Universiteit of Twente



The work reported in this dissertation has been carried out at the *Centrum voor Wiskunde en Informatica* (CWI) in Amsterdam, under the auspices of the *Instituut voor Programmatuurkunde en Algoritmiek* (IPA). The research has been supported by the BRICKS project (Basic Research in Informatics for Creating the Knowledge Society) funded by the *Besluit Subsidies Investeringen Kennisinfrastructuur* (BSIK).

VRIJE UNIVERSITEIT

Clocks, Dice and Processes

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. L.M. Bouter,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de faculteit der Exacte Wetenschappen
op maandag 21 september 2009 om 13:45 uur
in de aula van de universiteit,
De Boelelaan 1105

door

Taolue Chen

geboren te Taizhou, Jiangsu, China

promotoren: prof.dr. W.J. Fokkink
prof.dr. J.C. van de Pol

Contents

Preface

This dissertation can be viewed as an end point of my journey of being at CWI as an *onderzoeker in opleiding*.

First I would like to provide some very general remarks regarding my scientific career and the contents of the dissertation. The decision to work on it four years ago was mostly driven by my strong interests in mathematics and natural science in general. This interest is persistent, starting when I was a kid in primary school. I cannot recall which subject exactly held a great fascination for me at that time, but what I do realize afterward is that it shapes my life. A witness might be that during my education, it is always the mathematics-oriented subjects that allure me in the end. Whether it is a gift or a curse is left as an open problem. My conjecture (or hope) is that it is a gift.

The particular subjects in this dissertation are of purely theoretical consideration, although I hope (and believe) at least part of them can be applied in practice somehow. I study axiomatization problems of process algebras because of the beauty and elegance they bring to me, while the work on probabilistic systems contents partially my long-term quest on the nature of randomness. However, on top of them I have to admit that an important reason is that fortunately they fall into the scope of my expertise. The studies are largely problem-driven, on which I also would like to comment. It is well-recognized that there are two main axes to mathematical research: (1) Refining and building upon existing mathematical theories, e.g. trying to prove or disprove the remaining conjecture in well explored branches of mathematics (Andrew Wiles's proof of Fermats's Last Theorem is a very typical case of such efforts); and (2) Developing mathematical theories for new areas of interest. In my point of view, mathematicians of the second type are a little bit more appreciated in the community. This might be biased, but seems true at least in the theoretical computer science community. The list of winners of the Turing Award, which is generally recognized as the "Nobel Prize of computing", serves a strong support of my opinion. I fully understand this situation, and I admire the contributions of the second type with the highest respect. Nevertheless, I found that being the first type of mathematician gratifies my need of self-actualization more. In other words, my main scientific interest lies in trying to solve open questions, instead of developing new theories. This tendency decides the style of the current dissertation, and explains the diversity of topics it contains. One might criticize that it leads to fragmented accounts instead of a coherent story of the

PhD study. This is probably true; however, I am proud of the fact that the problems solved here are in general of a complicated nature, which is the virtue I cherish most.

Acknowledgements. This dissertation would not have been accomplished without the help (input) of many people.

First of all, I would like to thank my supervisor and promotor Wan Fokkink, who directed my research in the last four years. His rigorous research attitude and profound professional skills have deeply impressed me, and his high standards of integrity and easygoing manner have given me valuable influences that will surely last throughout my whole life. On the one hand, Wan worked closely with me, always willing to discuss my work and ready to give enlightening suggestions. On the other hand, he gave me the largest liberty to follow my own research interests. It's an honor for me to have had such a precious time with him and I am deeply indebted to him.

I would like to deliver my sincere thanks to my other promotor, Jaco van de Pol. Jaco was the head of our group until the end of 2007 when he became a full professor at the University of Twente. I enjoyed the fruitful discussions in his office and the pleasant conversations during lunch very much. His excellent ideas, brightness, enthusiasm and lots of care, both personally and professionally, remain an enjoyable memory of my PhD study.

I am very grateful to all my co-authors for their productive and pleasant cooperation. Apart from Wan and Jaco, they are Luca Aceto, Jasper Berendsen, Rob van Glabbeek, Tingting Han, Anna Ingólfsdóttir, David Jansen, Joost-Pieter Katoen, Bas Luttik, Alexandru Mereacre, Sumit Nain, Bas Ploeger, Yanjing Wang and Tim Willemse. I enjoyed their contributions and the fruitful discussions a lot. Although I could not fit all the results here, they have certainly influenced the state of my mind. I learned a lot from them, thanks!

Apart from my co-authors, many people also contributed to this dissertation (or my PhD study in general) in different ways. In particular, for Chapter 3, the second meta-theorem was inspired by a comment from an anonymous referee of [CFvG09]. For Chapter 4, Luca Aceto and Anna Ingólfsdóttir had stimulating discussions, and an anonymous referee for [CFLN08] suggested many improvements. For Chapter 5, the research was initiated by a question from Jos Baeten, and Bas Luttik, Marc Voorhoeve and Yanjing Wang provided very useful feedback. For Chapter 6, Jaco van de Pol, Vincent van Oostrom and the other participants in the PAM seminar at CWI provided useful comments. For Chapter 7, Boudewijn Haverkort, Jeremy Sproston, Mariëlle Stoelinga and an anonymous referee of [CHKM09b] provided fruitful discussions and/or insightful comments. For Chapter 8, Jeremy Sproston provided useful suggestion for improvements. Moreover, for some other works (not presented here), many people, in particular, Eric Allender, Christel Baier, Patricia Bouyer, Krishnendu Chatterjee, Flavio Corradini, Luca de Alfaro, Kousha Etessami, Jan Friso Groote, Marcin Jurdziński, Orna Kupferman, Kim Larsen, Nicolas Markey, Gordon Plotkin and Moshe Vardi provided inspiring comments and suggestions. I thank all of them!

I am grateful to the members of the reading committee, Luca Aceto, Joost-Pieter Katoen, Jan Willem Klop and Mariëlle Stoelinga for their extremely careful reviews of the dissertation, constructive comments and inspiring questions; to Jan Bergstra and Bas Luttik as well for their willingness to take part in the opposition.

Collective and individual thanks are also owed to all my (former) colleagues and friends at CWI or VU, in particular, Bahareh Badban, Rena Bakhshi, Jens Calamé, Susanne van Dam, Natalia Ioustinova, Bert Lisser, Bas Luttik, Simona Orzan, Mohammad Torabi Dashti, Yaroslav Usenko, Yanjing Wang, Michael Weber, Anton Wijs, Mike Zonsveld. I also wish to extend my gratitude to other (Chinese and non-Chinese) friends, in particular, Chao Li, Xirong Li, Bo Gao, Fangbin Liu, Huiye Ma, Qian Ma, Alexandru Mereacre, MohammadReza Mousavi, Jun Pang, Bas Ploeger, Judi Romijn, Meng Sun, Mengxiao Wu, Ping Yu, ... (I have a truly long list which this place is too narrow to contain :-) I enjoyed the great time I had with them and thank them for sharing many sides of life with me, both scientifically and personally.

My family in China, especially my parents, deserve my endless thanks for their unconditional love and support.

Last but not the least, I reserve my greatest thanks to Tingting for everything she did for me ...

Taolue Chen
Amsterdam, April 2009

P.S. I would like to explain the cover of the dissertation briefly. Basically, it contains two cartoons, which illustrate two stories, corresponding to the two parts of the dissertation. The hero of them is a mouse named "Process Algebra" (Let us call him Mr. Process Algebra). He is originally from China with a Chinese name "Shu Buqing" (meaning "countless" in English and "Shu" indicating the species in Chinese) and now he lives in the Netherlands. Of course, Mr. Process Algebra likes cheese very much[1] and he prefers a specific brand "Axiomatization". The first cartoon shows that, since Mr. Process Algebra has a good appetite on "Axiomatization" cheese, he always tries his best to seek axiomatizations, preferably finite ones. However, unfortunately, this task is not very easy since he has to go through a maze, which is quite complicated to him. Can he succeed? This is the problem I addressed in **Part I** of the dissertation. The second cartoon on the back cover shows that, YES, our cute mouse is enjoying his favorite cheese! He is very smart since he managed to go though the maze within 12min and 27sec, with the help of a dice by which he can make a random choice at the crossroad. So we can claim that "mouse $\models \mathbb{P}_{>0}(\mathbf{F}^{<15\text{min}}\text{cheese})$". This suggests the contents of **Part II** of the dissertation.

[1] Scientific research reveals that mice do not have a particular interest in cheese, but let us follow the folklore here.

Chapter 1

Introduction

This dissertation summarizes a large part of my PhD study on *concurrency theory* in the past four years. Concurrency theory, in general, aims to develop mathematical accounts of behaviors of *concurrent systems*, which typically consist of a number of components, each of which evolves simultaneously with the others, subject to (typically frequent) interaction amongst the components. In practice, it is fair to say that concurrency theory is mainly concerned with the *modeling* and *verification* of concurrent systems. This is the main objective of the research field of *formal methods*, which serve as the general context of this dissertation as well. In general, formal methods refer to a collection of notations and techniques for describing and analyzing systems. A formal method typically consists of a *formalism* to model a system, a *specification language* to express the desired properties of the system, a *formal semantics* to interpret both the system and the properties, and *verification techniques* to check whether these properties are satisfied by the system.

To obtain a more down-to-earth feeling on what "concurrency theory" concentrates on, one of the best avenues is to go through the themes listed by WG1.8 "Concurrency Theory" of IFIP TC1[1] which, in my opinion, encompass most aspects of concurrency theory and its applications. These themes include (1) process algebras and calculi; (2) expressiveness of formalisms for concurrency; (3) modal and temporal logics for concurrency and their extensions; (4) resource-sensitive approaches to concurrency and their developments; (5) tools for verification and validation of concurrent systems; (6) reactive models for real-time and hybrid systems; (7) calculi and typing systems for mobile processes and global computing; (8) stochastic and probabilistic models of concurrent processes; (9) behavioral relations for processes; and (10) decidability and complexity issues in concurrency theory.

The current dissertation mostly concerns (1), (6), (8) and (9), which presents results obtained along two lines of research: *the axiomatizability of process algebras* and *the verification of probabilistic real-time systems*. This also explains the title, since the contents can be divided into two distinct parts: "clocks"

[1] URL: `www.ru.is/luca/IFIPWG1.8/`.

and "dice", which form a metaphor for time and randomness; and "processes", which reminds of process algebra specifically and denotes the object of the dissertation in general. As a matter of fact, in this dissertation I do *not* pursue any particular link between these two parts, although conceptually they are closely related indeed. The only obvious link is that I carried out both of the research lines during the last four years.

Below I shall offer some general, but more technical introductions regarding these two topics.

1.1 Axiomatizability of Process Algebras

Process algebra, or process theory, constitutes an attempt to reason about "behaviors of systems" in a mathematical framework. Generally speaking, they are prototype specification languages for *reactive systems* – that is, for systems that compute by reacting to stimuli from their environment. There are a plethora of applications of process algebras (see, e.g. [Bae90]), but mostly they are applied to prove the correctness of system behaviors (see, e.g. [Fok07, AILS07]). In a nutshell, they enable us to express (un)desirable properties of the behavior of a system in an abstract way, and to deduce by mathematical manipulations whether or not the behavior satisfies such a property.

To put it more concretely, a process algebra usually consists of a collection of basic operations for constructing new system descriptions from existing ones, together with some facility for the recursive definition of system behaviors. This "algebraic" flavor explains somehow its name – process *algebra*, which was coined by Bergstra and Klop in 1982 [BK82]. (See also the essay [Lut06].) Well-known examples of such languages are CCS [Mil80, Mil89a], CSP [Hoa85] and ACP [BK84, BW90]. Starting from a syntax for a process algebra, each *syntactic* object is supplied with some kind of behavior, which is usually described by *labeled transition systems* (LTSs, [Kel76]). In general, LTSs are a fundamental formalism for the description of concurrent computation, which is widely used in light of its flexibility and applicability. In process algebras, LTSs model processes by explicitly describing their *states* and the *transitions* from state to state, together with the *actions* that produce them. In particular, they underlie Plotkin's *structural operational semantics* (SOS, [Plo04a, Plo04b]) and, following Milner's pioneering work on CCS, LTSs are by now the standard formalism for describing the *operational semantics* of various process algebras.

Since this LTS view of process behaviors is very detailed, several notions of behavioral semantics, in terms of *equivalences* and *preorders* have been proposed for LTSs. They are largely surveyed by van Glabbeek, forming the celebrated *linear time – branching time spectrum* [vG90, vG93b]. The aim of such behavioral semantics is to identify those (states of) LTSs that afford the same behaviors in some appropriate technical sense. These preorders and equivalences are mostly interesting when they are *precongruences* and *congruences* respectively, i.e., they are closed under the operators of the considered process algebras. In light of this, one may define intuitively appealing *semantic models*

for a process algebra as quotients of the collection of LTSs modulo some behavioral (pre)congruence. Notable examples of semantic equivalences include two extreme cases (in the sense of coarsest and finest), trace equivalence, and concrete (a.k.a. strong) bisimulation equivalence [Par81], together with two *weak* versions of concrete bisimulation equivalence, i.e. weak [Mil89a] and branching [vGW96] bisimulation equivalences.

Process algebra has become a full-fledged branch of theoretical computer science (TCS). Instead of supplying an exhaustive introduction, I refer the readers to [Bae05] for a historical account, [Hoa85, Hen88, Mil89a, BW90, Fok00] for excellent textbooks, [Mil90, BV95] for representative tutorials, and [BPS01] as the encyclopedia.

After this very condensed introduction to process algebra, let us focus on the main topic of the first part of the dissertation, *axiomatizability*. Typically, process algebra expresses a plethora of preorders and equivalences in axioms, or (in)equational laws. Axiom systems (or axiomatizations[2]) arise from the desire of isolating the features that are common to a collection of algebraic structures, namely, their semantics models. One requires that a set of axioms is *sound* (i.e., if two behaviors can be equated, then they are semantically related), and one desires that it is *complete* (i.e., if two behaviors are semantically related, then they can be equated). As a matter of fact, having defined a semantic model for a process algebra in terms of LTSs, it is natural to study its *(in)equational theory*, that is, the collection of (in)equations that are valid in the given model. The key questions here are:

- Are there reasonably informative *sound* and *complete* axiom systems for the chosen semantic model?
- Does the algebra of LTSs modulo the chosen notion of behavioral semantics afford a *finite* (in)equational axiomatization?

A sound and complete axiomatization of a behavioral congruence (resp. precongruence) yields a purely syntactic characterization, independent of LTSs and of the actual details of the definition of the chosen behavioral equivalence (resp. preorder), of the semantics of the process algebra. This bridge between syntax and semantics plays an important role in both the practice and the theory of process algebras. From the point of view of practice, these proof systems can be used to perform system verifications in a purely syntactic way using general purpose theorem provers or proof checkers, and form the basis of purpose-built axiomatic verification tools like, e.g. PAM [Lin95]. A positive answer to the first basic question raised above is therefore not just theoretically pleasing, but has potential practical applications. From the theoretical point of view, complete axiomatizations of behavioral equivalences (resp. preorders) capture the essence of different notions of semantics for processes in terms of a basic collection of identities, and this often allows one to compare semantics which might have been defined in very different styles and frameworks.

[2]Both of these two terminologies will be used interchangeably throughout the dissertation.

In *universal algebra* [BS81], a sound and complete axiomatization is referred to as a *basis* for the (in)equational theory of the algebra it axiomatizes. The existence of a finite basis for an (in)equational theory is a classic topic of study in universal algebra (see, e.g. [MMT87]), dating back to Lyndon [Lyn51]. Murskiĭ [Mur75] proved that "almost all" finite algebras (namely all quasi-primal ones) are finitely based, while in [Mur65] he presented an example of a three-element algebra that has no finite basis. Henkin [Hen77] showed that the algebra of naturals with addition and multiplication is finitely based, while Gurevič [Gur90] showed that after adding exponentiation the algebra is no longer finitely based. McKenzie [McK96] settled *Tarski's Finite Basis Problem* in the negative, by showing that the general question whether a finite algebra is finitely based is undecidable.

Process algebra, as a branch of universal algebra and equational logic, naturally features the study of results pertaining to the existence or non-existence of finite bases for algebras modulo given semantics. After nearly thirty years of intensive research on this topic, numerous problems still remain open (see e.g. [Ace03, AFIL05, AI07] for surveys). The first part of the dissertation is devoted to solving some of them (see Section 1.3.1 for a summary).

1.2 Verification of Probabilistic Real-time Systems

Embedded software is now omnipresent: it controls telephone switches and satellites, drives our climate control, runs pacemakers, and makes our cars and TVs work. It permanently interacts with its – mostly physical – environment via sensors and actuators.

Embedded applications often feature systems which exhibit both *probabilistic* and *real-time* behaviors: Embedded software must robustly and autonomously operate under highly unpredictable environmental conditions, which are typically modeled by randomness. They should also promptly react to stimuli from their environment (timeliness), which requires one to tackle quantitative information about time elapsing explicitly in their proper modeling. In general, systems exhibiting real-time and probabilistic aspects are referred to as *probabilistic real-time systems* in the current dissertation. The increasing reliance on embedded systems in diverse fields has led to growing interest in obtaining formal guarantees of system correctness. In light of this, the second part of the dissertation concentrates on applying formal verification techniques to probabilistic real-time systems.

Model checking is one of the most successful verification techniques for hard- and software[3], which, in a nutshell, is an automatic method for guaranteeing that a mathematical model of a system satisfies a formula representing a desired property. Usually the system is modeled by transition systems (typically, *Kripke structures* or *labeled transition systems*), and the formula is written in temporal logics, typically, *computation tree logic* (CTL, [CES86]), a *branching-time* logic

[3]The pioneers of model checking received the ACM Turing Award in 1996 (Amir Pnueli) and 2007 (Edmund M. Clarke, E. Allen Emerson and Joseph Sifakis).

due to Clarke and Emerson and *linear temporal logic* (LTL, [Pnu77]), a *linear-time* logic due to Pnueli and Manna. For textbooks on model checking, see [CGP99, BK08]; [GV08] presents state-of-the-art surveys. Soon after the birth of the flourishing research area of model checking in the early eighties (which largely focused on finite Kripke structures), researchers started to apply this technique to *real-time systems* extensively, with much attention given to the formalism of *timed automata* and to *probabilistic systems*, which are basically finite automata equipped with probabilities. A brief introduction now follows.

Real-time systems. Timed automata (TA, [AD94]), proposed by Alur and Dill, are a well-established model for real-time systems. This formalism extends classical automata with a set of real-valued variables – called *clocks* – that increase synchronously with time. Moreover, each transition associates a guard specifying when, i.e., for which values of the clocks, the transition can be performed and a reset operation to be applied when the transition is performed. Thanks to these clocks, it becomes possible to express constraints over delays between two transitions.

As for their formal verification, one of the fundamental properties of TA is that, though in general there are (uncountably) infinitely many possible system configurations, many verification problems are still decidable, e.g., reachability and safety properties, untimed ω-regular properties, and branching real-time temporal properties. This mainly goes back to a powerful *region* construction [AD94], yielding a finite-state abstraction for a TA, referred to as *region automaton*.

Various timed temporal logics [AH93] have been proposed, which extend classical untimed temporal logics by enriching modal operators with timing constraints. Examples are a timed extension of CTL called TCTL [ACD93], timed extensions of LTL such as M(I)TL [Koy90, AFH96] and TPTL [AH94], and a timed extension of μ-calculus called L_μ [LLW95]. Region abstraction is also pivotal to the decidability of model checking a TA against TCTL [ACD93]; for the decidability of M(I)TL, see [OW08] for a survey.

TA are a special class of hybrid automata [ACHH92], proposed by Alur *et al.*, which are a formal model for *hybrid systems*. A hybrid system is a dynamic system which exhibits both *continuous* and *discrete* dynamic behaviors – a system that can both flow (described by a differential equation) and jump (described by a difference equation). In general, a hybrid system can be described by a few pieces of information. The state of the system consists of vector signals, which can change according to dynamic laws in the system data. The data includes a flow equation, which describes the continuous dynamics, a flow set, in which flow is permitted, a jump equation, which describes the discrete dynamics, and a jump set, in which discrete state evolution is permitted. Hybrid automata provide an elegant and economical way to model hybrid systems. In contrast to TA, they are equipped with variables that evolve continuously with time according to dynamical laws, instead of simple clocks. Hybrid automata are very expressive, and model checking them is mostly undecidable [Hen96]. To facil-

itate the verification task, numerous subclasses of hybrid automata have been identified [HKPV98], including rectangular hybrid automata, and linear hybrid automata, to name a few.

There is a sea of publications on TA. The readers are referred to, among others, [AH91, AH97, Hen98, Alu99, BY03, AM04, BL08, Bou09] for surveys from different perspectives.

Probabilistic systems. To model random phenomena, transition systems are enriched with probabilities. This can be done in different ways. In *discrete-time Markov chains* (DTMCs), all choices are probabilistic. Roughly speaking, DTMCs are transition systems with probability distributions for the successors of each state. That is, instead of a nondeterministic choice, the next state is chosen probabilistically. DTMCs are not appropriate for modeling randomized distributed systems, since they cannot model the interleaving behavior of the concurrent processes in an adequate manner. For this purpose, *Markov decision processes* (MDPs, [Put94]) are used. In MDPs, both nondeterministic and probabilistic choices co-exist. To put ut in a nutshell, MDPs are transition systems in which in any state a nondeterministic choice between probability distributions exists. Once a probability distribution has been chosen nondeterministically, the next state is selected in a probabilistic manner as in DTMCs. Any DTMC is thus an MDP in which in any state the probability distribution is uniquely determined.

The verification of probabilistic systems can be focused on either *quantitative* properties or *qualitative* properties (or both). Quantitative properties typically put constraints on the probability or expectation of certain events. Example instances of quantitative properties are the requirement that the probability of delivering a message within the next t time units is at least 0.98, or that the expected number of unsuccessful attempts to find a leader in a concurrent system is at most seven. Qualitative properties, on the other hand, typically assert that a certain (good) event will happen *almost surely*, i.e., with probability one, or dually, that a certain (bad) event almost never occurs, i.e., with zero probability. Typical qualitative properties for Markov models are reachability, persistence (does from some point on a property always hold?), and repeated reachability (can certain states be reached repeatedly?).

One way to specify properties over DTMCs and MDPs is to use LTL, or alternatively ω-regular properties [Tho97]. The quantitative model-checking problem is then to compute the probability for the set of paths in the model that satisfy the property. The readers are referred to [Var85, CY95] for original work. On the other hand, *probabilistic computation tree logic* (PCTL, [HJ94]) proposed by Hansson and Jonsson, is a quantitative variant of CTL where the path quantifiers $\forall$ and $\exists$ are replaced by a probabilistic operator $\mathbb{P}_J(\varphi)$ that specifies lower and/or upper probability bounds (given by J) for the event φ. PCTL model checking for finite DTMCs relies on the standard CTL model-checking procedure in combination with methods for computing reachability probabilities [HJ94]. When interpreting PCTL on MDPs, the formula $\mathbb{P}_J(\varphi)$

ranges over all schedulers (a.k.a. strategies, policies, adversaries). The PCTL model-checking problem for MDPs is reducible to the reachability problem, as shown by Bianco and de Alfaro [BdA95].

Probabilistic real-time systems. Unsurprisingly, a number of models encompassing both probabilistic and real-time aspects have appeared already in the literature. In particular, here I mention two classes of them:

- *Probabilistic timed automata* (PTA) extend both TA and MDPs. In contrast to TA, the edge relation of PTA is both *nondeterministic* and *probabilistic* in nature. More precisely, in a location one nondeterministically chooses amongst the set of enabled *discrete* probability distributions, each of which is defined over a finite set of locations. Once a selection has been made, the next location is randomly determined according to the selected distribution. This formalism has proved particularly useful for the study of timed randomized algorithms, such as the IEEE1394 FireWire standard [KNS03]. PTA have been verified against quantitative probabilistic timed properties [KNSS02, KNSW07], which refer to the satisfaction of a property with a certain probability. An example of such a property is "with probability at least 0.99, a request will be followed by a response within 5 milliseconds". Model checking a PTA against a probabilistic extension of TCTL (i.e., PTCTL) is decidable using a small adaptation of the region construction for TA [KNSS02]. Recently, a natural extension of PTA, i.e., *priced* PTA (PPTA) has been defined, and a semi-decidable algorithm has been provided for cost-bounded probabilistic reachability [BJK06].

- *Continuous-time stochastic models.* The aforementioned probabilistic models are *discrete*, as the notion of time involved is of a discrete nature: each transition represents a single time step. This contrasts with *continuous-time models* where *state residence times* are determined by some probability distribution like a normal, uniform, or negative exponential one.

  One of the most fundamental models is *continuous-time Markov chains* (CTMCs). They can be seen as generalizations of DTMCs, by enhancing them with negative exponential state residence time distributions. With the similar extension to MDPs, one can obtain *continuous-time Markov decision processes* (CTMDPs). CTMCs play an essential role in performance and dependability analysis. They are exploited in a broad range of applications, and constitute the underlying semantical model of a plethora of modeling formalisms for probabilistic real-time systems such as Markovian queueing networks, stochastic Petri nets, stochastic variants of process algebras, and, more recently, calculi for systems biology.

  CTMC model checking has been focused on the temporal logic *continuous stochastic logic* (CSL, [ASSB00, BHHK03]), a variant of TCTL where the CTL path quantifiers are replaced by a probabilistic operator as in the case of PCTL. CSL model checking proceeds – like CTL model checking – by a recursive descent over the parse tree of the formula. One of the

key ingredients is that the reachability probability for a time-bounded until-formula can be characterized as the least solution of a system of integral equations and approximated arbitrarily closely by a reduction to *transient analysis* in CTMCs. This results in a polynomial-time algorithm that has been realized in model checkers such as PRISM [HKNP06] and MRMC [KKZ05].

Numerous (more general or incomparable) models do exist. They include, for instance, probabilistic timed I/O automata [ML07], generalized semi-Markov processes (studied in [ACD91a, ACD91b] almost twenty years ago and still a very active area of research [AB06, BA07]), stochastic transition systems [CSKN05], labeled Markov processes [DEP02], to name a few. To give an exhaustive survey of research on probabilistic (real-time) systems is far beyond the scope of the dissertation. For further information, the readers are referred to [BHH$^+$04] for general tutorials (see also the reference therein). Moreover, a leisurely introduction for (discrete-time) probabilistic models can be found in [Sto02]; [Hav00] offers a tutorial for Markovian models, [Kat08a, Kat08b] provide state-of-the-art surveys regarding the probabilistic verification, and [Spr04, KNPS08] supply extensive introductions to probabilistic real-time systems.

1.3 Overview of the Dissertation

This section lists the results that are achieved in this dissertation.

1.3.1 Part I: Axiomatizability of Process Algebras

Chapters 3 – 6 concentrate on the axiomatizability of process algebras, mostly on two basic process algebras BCCSP and BCCS (see Section 2.1.4 for their formal accounts).

Chapter 3: Meta-theories for Axiomatizability. This chapter mainly presents two meta-theorems regarding axiomatizability. The first one concerns the relationship between preorders and equivalences. We show that the same algorithm proposed by Aceto *et al.* and de Frutos Escrig *et al.* for concrete semantics, which transforms an axiomatization for a preorder to the one for the corresponding equivalence, applies equally well to weak semantics. This makes it applicable to all 87 preorders surveyed in the "linear time – branching time spectrum II" that are at least as coarse as the ready simulation preorder. We also extend the scope of the algorithm to infinite processes, by adding recursion constants. As an application of both extensions, we provide a ground-complete axiomatization of the CSP failures equivalence for BCCS processes with divergence. The second meta-theorem concerns the relationship between concrete and weak semantics. For any semantics which is not finer than *failures* or *impossible futures* semantics, we provide an algorithm to transform an axiomatization for the *concrete* version to the one for the *weak* counterpart. As an application of this algorithm, we derive ground- and ω-complete axiomatizations for the

weak failure, weak completed trace, weak trace preorders. Finally, we provide an adaptation of Groote's inverted substitution technique from equivalence to preorder.

Chapter 4: On Finite Alphabets and Infinite Bases. Van Glabbeek (1990) presented the "linear time – branching time spectrum I" of concrete semantics. He studied these semantics in the setting of the process algebra BCCSP, and gave finite, sound and ground-complete axiomatizations for most of these semantics. Groote (1990) proved for some of van Glabbeek's axiomatizations that they are ω-*complete*, meaning that an equation can be derived if (and only if) all of its closed instantiations can be derived. We settle the remaining open questions for all the semantics in the linear time – branching time spectrum I, either positively by giving a finite sound and ground-complete axiomatization that is ω-complete, or negatively by proving that such a finite basis for the equational theory does not exist. We show that in case of a finite alphabet with at least two actions, failure semantics affords a finite basis, while for ready simulation, completed simulation, simulation, possible worlds, ready trace, failure trace and ready semantics, such a finite basis does not exist. Completed simulation semantics also lacks a finite basis in case of an infinite alphabet of actions.

Chapter 5: Impossible Futures. This chapter studies the (in)equational theories of concrete and weak impossible futures semantics over the process algebras BCCSP and BCCS. We present a finite, sound, ground-complete axiomatization for BCCSP modulo the concrete impossible futures *preorder*, which implies a finite, sound, ground-complete axiomatization for BCCS modulo the *weak* impossible futures preorder. By contrast, we prove that no finite, sound axiomatization for BCCS modulo the weak impossible futures *equivalence* is ground-complete, and this negative result carries over to the concrete case. If the alphabet of actions is infinite, then the aforementioned ground-complete axiomatizations are shown to be ω-complete. However, if the alphabet is finite and non-empty, we prove that the inequational (resp. equational) theories of BCCSP and BCCS modulo the impossible futures preorder (resp. equivalence) lack such a finite basis. Finally, we show that the negative result regarding impossible futures equivalence extends to all n-nested impossible futures equivalences for $n \geq 2$, and to all n-nested impossible futures preorders for $n \geq 3$.

Chapter 6: Priority. This chapter studies the equational theory of bisimulation equivalence over the process algebra BCCSP_Θ, i.e., BCCSP extended with the priority operator Θ of Baeten *et al.* It is proven that, in the presence of an infinite set of actions, bisimulation equivalence has no finite, sound, ground-complete axiomatization over that language. This negative result applies even if the syntax is extended with an arbitrary collection of auxiliary operators, and motivates the study of axiomatizations using equations with action predicates as conditions. In the presence of an infinite set of actions, it is shown that,

in general, bisimulation equivalence has no finite, sound, ground-complete axiomatization consisting of equations with action predicates as conditions over BCCSP_Θ. Finally, sufficient conditions on the priority structure over actions are identified that lead to a finite, sound, ground-complete axiomatization of bisimulation equivalence using equations with action predicates as conditions.

1.3.2 Part II: Verification of Probabilistic Real-time Systems

Chapter 7 – 8 concentrate on the verification of probabilistic real-time systems.

Chapter 7: Model Checking of CTMCs Against DTA Specifications. This chapter investigates the following problem: given a continuous-time Markov chain (CTMC) $\mathcal{C}$, and a linear real-time property provided as a deterministic timed automaton (DTA) $\mathcal{A}$, what is the probability of the set of paths of $\mathcal{C}$ that are accepted by $\mathcal{A}$ ($\mathcal{C}$ satisfies $\mathcal{A}$)? It is shown that this set of paths is measurable and computing its probability can be reduced to computing the reachability probability in a piecewise deterministic Markov process (PDP). The reachability probability is characterized as the least solution of a system of integral equations and is shown to be approximated by solving a system of partial differential equations. For the special case of single-clock DTA, the system of integral equations can be transformed into a system of linear equations where the coefficients are solutions of ordinary differential equations.

Chapter 8: Probabilistic Time-abstracting Bisimulation for PTA. This chapter focuses on probabilistic timed automata (PTA), an extension of timed automata with discrete probabilistic branchings. As the region construction of these automata often leads to an exponential blow-up over the size of original automata, reduction techniques are of the utmost importance. In this chapter, we investigate probabilistic time-abstracting bisimulation (PTAB), an equivalence notion that abstracts from exact time delays. PTAB is proven to preserve probabilistic computation tree logic. The region equivalence is a (very refined) PTAB. Furthermore, we provide a non-trivial adaptation of the traditional partition-refinement algorithm to compute the quotient under the PTAB. This algorithm is symbolic in the sense that equivalence classes are represented as polyhedra.

1.4 Origins of the Chapters and Credits

Chapter 3 is based on [CFvG08] and [CFvG09]:

> [CFvG08] Taolue Chen, Wan Fokkink, and Rob J. van Glabbeek. Ready to preorder: The case of weak process semantics. *Inf. Process. Lett.*, 109(2):104–111, 2008.
>
> [CFvG09] Taolue Chen, Wan Fokkink, and Rob J. van Glabbeek. On finite bases for weak semantics: Failures versus impossible futures. Proc. of *SOFSEM*, LNCS 5404, pages 167–180. Springer, 2009.

In 2007, Wan Fokkink, Rob van Glabbeek and I were studying the axiomatization for *weak* failures equivalence. We found that the elegant algorithm given in [AFI07] can be applied equally well to weak semantics, which gives rise to the results in [CFvG08] and now the first meta-theorem in this chapter. The second meta-theorem is inspired by an anonymous referee's comment for [CFvG09]. Thanks to this result, part of the proofs in [CFvG09] can be simplified considerably.

Chapter 4 is based on [CFLN08]:

> [CFLN08] Taolue Chen, Wan Fokkink, Bas Luttik, and Sumit Nain. On finite alphabets and infinite bases. *Inf. Comput.*, 206(5):492–519, 2008.

It subsumes a series of "on finite alphabets and infinite bases" papers [FN04, FN05, CFN06, CF06] trying to solve the open problem posed by Rob van Glabbeek dating back to 1990. The results on failures, ready pairs, ready traces and possible worlds presented in [FN04, FN05] are due to Wan Fokkink and Sumit Nain and were finished before I started the PhD study, although in the end I did contribute to writing the journal version [CFLN08], including rewriting these parts. Nevertheless, I am not supposed to get any credit regarding these results. However, I decided to include them here for the only sake of completeness, since without them the chapter would have become fragmented. [CFN06, CF06] are:

> [CFN06] Taolue Chen, Wan Fokkink, and Sumit Nain. On finite alphabets and infinite bases II: Completed and ready simulation. Proc. of *FoSSaCS*, LNCS 3921, pages 1–15. Springer, 2006.
>
> [CF06] Taolue Chen and Wan Fokkink. On finite alphabets and infinite bases III: Simulation. Proc. of *CONCUR*, LNCS 4137, pages 421–434. Springer, 2006.

Chapter 5 is based on two conference papers [CF08] and [CFvG09]:

> [CF08] Taolue Chen and Wan Fokkink. On the axiomatizability of impossible futures: Preorder versus equivalence. In *LICS*, pages 156–165. IEEE Computer Society, 2008.
>
> [CFvG09] Taolue Chen, Wan Fokkink, and Rob J. van Glabbeek. On finite bases for weak semantics: Failures versus impossible futures. Proc. of *SOFSEM*, LNCS 5404, pages 167–180. Springer, 2009.

In 2007, spurred by a question from Jos Baeten, Wan Fokkink and I started to investigate the impossible futures semantics, which can be viewed as a continuation of the work in [CFLN08]. To our great surprise, this semantics enjoys a special (at least in the aforementioned linear time – branching time spectrum) in the sense that it affords a ground-complete axiomatization for the *preorder* but lacks one for the *equivalence*. Our results regarding the concrete impossible

futures semantics were reported in [CF08]. After that, we were trying to push the work to the weak version and realized that, surprisingly, it resembles the failure semantics in definition, but differs in the axiomatizability considerably. Together with Rob van Glabbeek, we obtained a first result regarding weak impossible futures and failures semantics, reported in [CFvG09]. As said, after that, inspired by a referee's comment, we found that [CFvG09] can be simplified, which leads to the current formulation of the results.

Chapter 6 is based on [ACFI08], which is the extended version of [ACFI06]:

> [ACFI06] Luca Aceto, Taolue Chen, Wan Fokkink, and Anna Ingólfsdóttir. On the axiomatizability of priority. Proc. of *ICALP (2)*, LNCS 4052, pages 480–491. Springer, 2006.
>
> [ACFI08] Luca Aceto, Taolue Chen, Wan Fokkink, and Anna Ingólfsdóttir. On the axiomatisability of priority. *Mathematical Structures in Computer Science*, 18(1):5–28, 2008.

The question of the axiomatization for the priority operator appears as an open problem in [AFIN06]. In 2005, Wan Fokkink brought this question into my attention and together with Luca Aceto, Anna Ingólfsdóttir and him, I finally managed to solve it in a large sense.

Chapter 7 is based on [CHKM09b]:

> [CHKM09b] Taolue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre. Quantitative model checking of continuous-time Markov chains against timed automata specifications. Proc. of *LICS*. IEEE Computer Society, 2009.

This work was inspired by a question posed by myself on how to verify *linear real-time* properties for CTMCs, during a visit to RWTH Aachen in August, 2008. While the initial attempt on MTL specifications failed in the end, together with Tingting Han, Joost-Pieter Katoen and Alexandru Mereacre, I found an approach to tackle DTA specifications which gives rise to the results presented in this chapter.

Chapter 8 is based on [CHK08]:

> [CHK08] Taolue Chen, Tingting Han, and Joost-Pieter Katoen. Time-abstracting bisimulation for probabilistic timed automata. Proc. of *TASE*, pages 177–184. IEEE Computer Society, 2008.

The question was posed by Joost-Pieter Katoen in 2007. Together with Tingting Han and him, I worked out the definition of PTaB over PTA, and a zone-based algorithm for computing it.

Besides the aforementioned work, I also studied some other problems. For instance, together with Bas Ploeger, Jaco van de Pol and Tim Willemse, I

studied equivalence checking for infinite systems using *parameterized boolean equation systems*, which is reported in [CPvdPW07]. Together with Jaco van de Pol and Yanjing Wang, I studied the *propositional dynamic logic* over *accelerated labeled transition systems*, including the model checking, satisfiability checking and axiomatization problems, which is reported in [CvdPW08]. Together with Jasper Berendsen and David Jansen, I proved the undecidability of cost-bounded probabilistic reachability problem for PPTA, which is reported in [BCJ09]. Together with Tingting Han, Joost-Pieter Katoen and Alexandru Mereacre, I studied LTL model checking for *inhomogeneous* CTMCs, which is reported in [CHKM09a]. However, these studies are not included in the current dissertation. The references are in order:

[CPvdPW07] Taolue Chen, Bas Ploeger, Jaco van de Pol, and Tim A. C. Willemse. Equivalence checking for infinite systems using parameterized boolean equation systems. Proc. of *CONCUR*, LNCS 4703, pages 120–135. Springer, 2007.

[CvdPW08] Taolue Chen, Jaco van de Pol, and Yanjing Wang. PDL over accelerated labeled transition systems. Proc. of *TASE*, pages 193–200. IEEE Computer Society, 2008.

[BCJ09] Jasper Berendsen, Taolue Chen, and David N. Jansen. Undecidability of cost-bounded reachability in priced probabilistic timed automata. Proc. of *TAMC*, LNCS 5532, pages 128–137. Springer, 2009.

[CHKM09a] Taolue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre. LTL model checking of time-inhomogeneous Markov chains. Proc. of *ATVA*, To appear in LNCS. Springer, 2009.

1.5 Suggested Way of Reading

This dissertation is written as a collection of independent chapters. Each chapter (from Chapter 3 to Chapter 8) comes with its own introduction and road map. Related and future work and conclusions, when relevant, appear for each chapter. These chapters are however not entirely self-contained. Some preliminary notions and notations are given in Chapter 2. (Note that only (very) basic notions which will be addressed in multiple chapters later are collected in this chapter, while those pertaining to an individual chapter are included in the preliminary section of the corresponding chapter.) Together with them, each chapter can be read separately. If the reader is interested in the first part, I suggest to first read Chapter 3, since it gives a flavor of the whole of part I, and contains results which suggest the later developments.

Due to the diversity of the topics and heavy usage of symbols in the dissertation, I found it difficult to unify the notations for all the chapters. In other words, overloading is unavoidable. However, for each chapter, the notation is consistent and unambiguous, so readers are kindly requested to take a *local* instead of a global view of notations throughout this dissertation. Moreover, since

most of results presented here are fruits of collaborations (see Section 1.4), to emphasize this fact, I shall use "we" instead of "I" in the remainder of this dissertation. (Note that this already applies to Section 1.3.)

Chapter 2

Preliminaries

2.1 Background for Part I

2.1.1 Labeled Transition Systems

We assume a non-empty, countable set A of *concrete* (a.k.a. *observable*, *external*, *visible*) actions, not containing the distinguished symbol τ. Following Milner, the symbol τ will be used to denote a *hidden* (a.k.a. *unobservable*, *internal*, *invisible*) action of a system. We define $A_\tau \stackrel{\text{def}}{=} A \cup \{\tau\}$, and use $a, b, \ldots$ to range over A and α, β to range over A_τ.

Definition 2.1.1 (Labeled transition system) A *labeled transition system* (LTS) consists of

- A set of *states* S, with typical element s; and
- A *transition relation* $\rightarrow \subseteq S \times L \times S$, where $L \subseteq A_\tau$ is a set of *labels* ranged over by ℓ.

We write $s \xrightarrow{\ell} s'$ if the triple (s, ℓ, s') is an element of $\longrightarrow$. Let $\ell_1 \cdots \ell_k$ be a sequence of labels; we write $s \xrightarrow{\ell_1 \cdots \ell_k} s'$ if there exist states $s_0, \ldots, s_k$ such that $s = s_0 \xrightarrow{\ell_1} \cdots \xrightarrow{\ell_k} s_k = s'$.

Given any $X \subseteq A_\tau$, we refer to an X-labeled transition system, if the label set L is X. For the case that $\tau \notin X$, we define $\mathcal{I}(s) = \{\ell \in X \mid s \xrightarrow{\ell} s' \text{ for some state } s'\}$. For the case that $\tau \in X$, we write $\Rightarrow$ for the transitive closure of $\xrightarrow{\tau}$, i.e., $\Rightarrow \stackrel{\text{def}}{=} (\xrightarrow{\tau})^*$, and define $\mathcal{I}(s) = \{\ell \in A \mid s \Rightarrow \xrightarrow{\ell} s' \text{ for some state } s'\}$.

2.1.2 The Linear Time – Branching Time Spectrum

A large number of notions of behavioral semantics have been proposed, with the aim to identify those states of LTSs that afford the same behaviors. The diversity of the proposals is mainly due to the lack of consensus on what constitutes

the most appropriate notion of observable behaviors for reactive systems. Van Glabbeek presented in [vG90, vG01] the linear time – branching time spectrum I of behavioral semantics for finitely branching, *concrete*, sequential processes. In this section, we first define the preorder and equivalence relations in this spectrum.

First we define six semantics based on decorated versions of execution traces. Note that here we only consider *finite* (decorated) traces. However, it suffices for the later development, since in this dissertation, we only tackle finite process algebras which give rise to tree-like LTSs (see Section 2.1.4).

Definition 2.1.2 (Decorated traces) Assume an A-labeled transition system.

- A sequence $a_1 \cdots a_k$, with $k \geq 0$, is a *trace* of a state s if there is a state s' such that $s \xrightarrow{a_1 \cdots a_k} s'$. It is a *completed trace* of s if moreover $\mathcal{I}(s') = \emptyset$. We write $\mathcal{T}(s)$ (resp. $\mathcal{CT}(s)$) for the set of traces (resp. completed traces) of s.

- A pair $(a_1 \cdots a_k, B)$, with $k \geq 0$ and $B \subseteq A$, is a *ready pair* of a state s_0 if there is a sequence of transitions $s_0 \xrightarrow{a_1} \cdots \xrightarrow{a_k} s_k$ with $\mathcal{I}(s_k) = B$. It is a *failure pair* of s_0 if there is such a sequence with $\mathcal{I}(s_k) \cap B = \emptyset$.

- A sequence $B_0 a_1 B_1 \ldots a_k B_k$, with $k \geq 0$ and $B_0, \ldots, B_k \subseteq A$, is a *ready trace* of a state s_0 if there is a sequence of transitions $s_0 \xrightarrow{a_1} \cdots \xrightarrow{a_k} s_k$ with $\mathcal{I}(s_i) = B_i$ for $i = 0, \ldots, k$. It is a *failure trace* of s_0 if there is such a sequence with $\mathcal{I}(s_i) \cap B_i = \emptyset$ for $i = 0, \ldots, k$.

We write $s \precsim_{\square} s'$ with $\square \in \{\mathrm{T}, \mathrm{CT}, \mathrm{R}, \mathrm{F}, \mathrm{RT}, \mathrm{FT}\}$ if the traces, completed traces, ready pairs, failure pairs, ready traces or failure traces, respectively, of s are included in those of s'.

Next we define five semantics based on simulation.

Definition 2.1.3 (Simulations) Assume an A-labeled transition system.

- A binary relation $\mathcal{R}$ on states is a *simulation* if $s_0 \ \mathcal{R} \ s_1$ and $s_0 \xrightarrow{a} s_0'$ imply $s_1 \xrightarrow{a} s_1'$ for some state s_1' with $s_0' \ \mathcal{R} \ s_1'$.

- A simulation $\mathcal{R}$ is a *completed simulation* if $s_0 \ \mathcal{R} \ s_1$ and $\mathcal{I}(s_0) = \emptyset$ imply $\mathcal{I}(s_1) = \emptyset$.

- A simulation $\mathcal{R}$ is a *ready simulation* if $s_0 \ \mathcal{R} \ s_1$ and $a \notin \mathcal{I}(s_0)$ imply $a \notin \mathcal{I}(s_1)$.

- A simulation $\mathcal{R}$ is a *2-nested simulation* if $\mathcal{R}^{-1}$ is included in a simulation.

- A *bisimulation* is a symmetric simulation.

We write $s \precsim_\Box s'$ with $\Box \in \{\mathrm{S}, \mathrm{CS}, \mathrm{RS}, \mathrm{2N}\}$ if there exists a simulation, completed simulation, ready simulation or 2-nested simulation $\mathcal{R}$, respectively, with $s \mathcal{R} s'$. We write $s \leftrightarroweq s'$ if there exists a bisimulation $\mathcal{R}$ with $s \mathcal{R} s'$.

Finally, we define three semantics based on *(im)possible futures* and on *possible worlds*.

Definition 2.1.4 ((Im)Possible futures/worlds) Assume an A-labeled transition system.

- A pair $(a_1 \cdots a_k, X)$, with $n \geq 0$ and $X \subseteq A^*$ is a *possible future* of a state s_0 if there is a sequence of transitions $s_0 \xrightarrow{a_1} \cdots \xrightarrow{a_k} s_k$ where X is the set of traces of s_k.
- A pair $(a_1 \cdots a_k, X)$, with $k \geq 0$ and $X \subseteq A^*$, is an *impossible future* of a state s_0 if there is a sequence of transitions $s_0 \xrightarrow{a_1} \cdots \xrightarrow{a_k} s_k$ for some state s_k with $\mathcal{T}(s_k) \cap X = \emptyset$.
- A state s is *deterministic* if for each $a \in \mathcal{I}(s)$ there is exactly one state s_0 such that $s \xrightarrow{a} s_0$, and moreover s_0 is deterministic. A state s is a *possible world* of a state s_0 if s is deterministic and $s \mathcal{R} s_0$ for some ready simulation $\mathcal{R}$.

We write $s \precsim_\Box s'$ with $\Box \in \{\mathrm{PF}, \mathrm{IF}, \mathrm{PW}\}$ if the possible futures, impossible futures or the possible worlds, respectively, of s are included in those of s'.

In general, we write $s \simeq_\Box s'$ if both $s \precsim_\Box s'$ and $s' \precsim_\Box s$ for $\Box \in \{\mathrm{T}, \mathrm{CT}, \mathrm{R}, \mathrm{F}, \mathrm{RT}, \mathrm{FT}, \mathrm{S}, \mathrm{CS}, \mathrm{RS}, \mathrm{2N}, \mathrm{PF}, \mathrm{IF}, \mathrm{PW}\}$.

Fig. 2.1 depicts the linear time – branching time spectrum I[1], where an arrow from one semantics to another means that the source of the arrow is finer than the target.

In contrast, in [vG93b], 155 *weak* semantics, which take into account the hidden action τ, are surveyed. They constitute the "linear time – branching time spectrum II" for finitely branching, *abstract*, sequential processes. To give a complete description of this spectrum falls outside the scope of the current dissertation. Below we only select some of the semantics in the spectrum which will be used in the later chapters. We first define weak versions of decorated traces.

Definition 2.1.5 (Weak decorated traces) Assume an A_τ-labeled transition system.

- A sequence $a_1 \cdots a_k$, with $k \geq 0$, is a *weak trace* of a state s if there is a state s' such that $s \Rightarrow \xrightarrow{a_1} \Rightarrow \cdots \Rightarrow \xrightarrow{a_k} \Rightarrow s'$. It is a *weak completed trace* of s if moreover $\mathcal{I}(s') = \emptyset$. We write $\mathcal{WT}(s)$ (resp. $\mathcal{WCT}(s)$) for the set of weak traces (resp. completed traces) of s.

[1] Note that the completed simulation and impossible futures semantics were missing in the original spectrum I [vG90, vG01], but are included here.

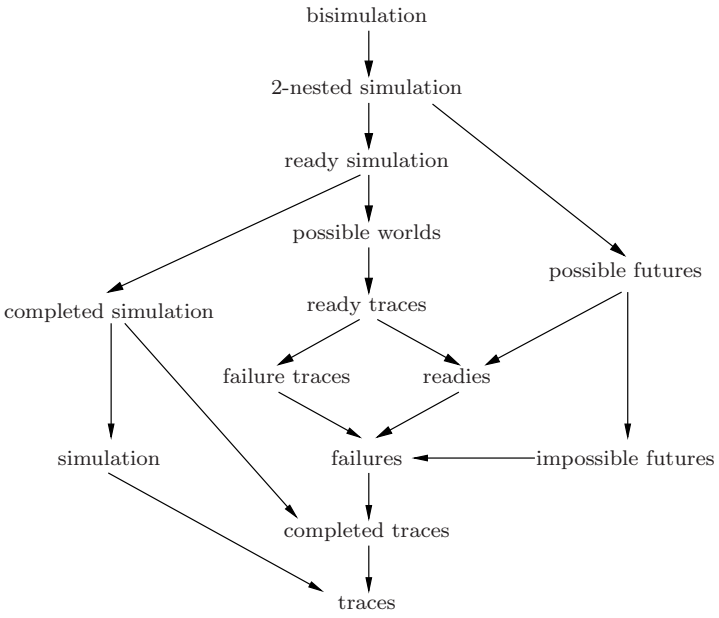


Figure 2.1: The linear time – branching time spectrum I

- A pair $(a_1 \cdots a_k, B)$, with $k \geq 0$ and $B \subseteq A$, is a *weak failure pair* of a state s_0 if there is a sequence of transitions $s_0 \Rightarrow \xrightarrow{a_1} \Rightarrow \cdots \Rightarrow \xrightarrow{a_k} \Rightarrow s'$ with $\mathcal{I}(s') \cap B = \emptyset$.
- A pair $(a_1 \cdots a_k, X)$, with $k \geq 0$ and $X \subseteq A^*$, is a *weak impossible future* of a state s if there is a sequence of transitions $s \Rightarrow \xrightarrow{a_1} \Rightarrow \cdots \Rightarrow \xrightarrow{a_k} \Rightarrow s'$ with $\mathcal{WT}(s') \cap X = \emptyset$.

As in the concrete case, the notion of weak decorated traces naturally induces weak (linear-time) semantics, as follows.

Definition 2.1.6
- We write $s \precsim_\Box s'$ with $\Box \in \{\mathrm{WT}, \mathrm{WCT}, \mathrm{WF}, \mathrm{WIF}\}$ if the weak traces, weak completed traces, weak failure pairs, or weak impossible futures, respectively, of s are included in those of s'.
- *Weak trace preorder* $\preceq_{\mathrm{WT}}$, *weak completed trace preorder* $\preceq_{\mathrm{WCT}}$ and *weak failures preorder* $\preceq_{\mathrm{WF}}$ are defined as: for $\Box \in \{\mathrm{WT}, \mathrm{WCT}, \mathrm{WF}\}$, $s \preceq_\Box s'$ iff (1) $s \precsim_\Box s'$ and (2) $s \xrightarrow{\tau}$ implies that $s' \xrightarrow{\tau}$.
- *Weak impossible future preorder* $\preceq_{\mathrm{WIF}}$ is defined as follows: $s \preceq_{\mathrm{WIF}} s'$ if (1) $s \precsim_{\mathrm{WIF}} s'$; (2) $s \xrightarrow{\tau}$ implies that $s' \xrightarrow{\tau}$ and (3) $\mathcal{WT}(s) = \mathcal{WT}(s')$.
- *Weak trace equivalence* $\simeq_{\mathrm{WT}}$, *weak completed trace equivalence* $\simeq_{\mathrm{WCT}}$, *weak failures equivalence* $\simeq_{\mathrm{WF}}$ and *weak impossible futures equivalence* $\simeq_{\mathrm{WIF}}$ are defined as: for $\Box \in \{\mathrm{WT}, \mathrm{WCT}, \mathrm{WF}, \mathrm{WIF}\}$, $\simeq_\Box = \preceq_\Box \cap \preceq_\Box^{-1}$.

2.1.3 Universal Algebra and Equational Logic

The aim of this section is to present a suitable general framework within which the necessary technical algebraic background can be described. This framework consists of the classic fields of *universal algebra* and *equational logic*. We therefore begin by introducing the basic notions from these areas of mathematical research that will be used throughout the first part of the dissertation. Most of material here is excerpted from [AFIL05]. We state at the outset that we shall not need very deep results or constructions from universal algebra in what follows, and that much more on it could be found in, e.g., the classic reference [BS81, MMT87] (see also [DW02] for applications in TCS). A self-contained presentation from a computer science perspective of the topics we now proceed to introduce may be found in [Hen88].

Σ-algebras We start from a countably infinite set V of *variables*, with typical elements x, y, w, z. A *signature* Σ consists of a set of *operation symbols*, disjoint from V, together with a function *arity* that assigns a natural number to each operation symbol. The set of *terms* over Σ is the least set such that

- Each $x \in V$ is a term.
- If f is an operation symbol of arity n, and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is also a term.

An operation symbol f of arity 0 will be often called a *constant* symbol, and the term $f()$ will be abbreviated as f.

We write $\mathbb{T}(\Sigma)$ for the set of all terms over Σ, and use t, u, v, possibly subscripted and/or superscripted, to range over terms. A term is *closed* (a.k.a. *ground*) if it contains no occurrences of variables. We denote by $\mathrm{T}(\Sigma)$ the set of closed terms over Σ. A *substitution*, denoted by ρ, σ, is a mapping from variables to terms. A substitution is *closed* if it maps variables to closed terms. For every term t and substitution σ, the term obtained by replacing every occurrence of a variable x in t with the term $\sigma(x)$ will be written $\sigma(t)$. Note that $\sigma(t)$ is closed if σ is.

The collection of terms over a signature Σ yields a *language*. The semantics of this language can be defined canonically once we equip the set of intended denotations with the structure of a Σ-algebra. A Σ-*algebra* is a structure

$$\mathcal{A} = (\mathbf{A}, \{f^{\mathcal{A}} \mid f \in \Sigma\}) \ ,$$

where $\mathbf{A}$ is a non-empty set (often called the *carrier* of the algebra), and

$$f^{\mathcal{A}} : \mathbf{A}^n \to \mathbf{A}$$

for each operation symbol $f \in \Sigma$ of arity n. Note that if f is a constant symbol, then $f^{\mathcal{A}}$ can be viewed as an element of $\mathbf{A}$.

In order to interpret terms in $\mathbb{T}(\Sigma)$ in a Σ-*algebra* $\mathcal{A} = (\mathbf{A}, \{f^{\mathcal{A}} \mid f \in \Sigma\})$ we need the notion of an environment. An *environment* is a function ρ mapping

variables to elements of $\mathbf{A}$. The mapping ρ can be extended homomorphically to $\mathbb{T}(\Sigma)$ in a unique way by stipulating that

$$\rho(f(t_1,\ldots,t_n)) = f^{\mathcal{A}}(\rho(t_1),\ldots,\rho(t_n))$$

for each operation symbol f of arity n and terms $t_1,\ldots,t_n$. Note that $\rho(t)$ is independent of ρ whenever t is closed. For each closed term t, we write $t^{\mathcal{A}}$ for the element of $\mathcal{A}$ that is the interpretation of t in the algebra $\mathcal{A}$. An element of the carrier set of $\mathcal{A}$ is *denotable* if it is the interpretation of some closed term. The interpretation of the language $\mathbb{T}(\Sigma)$ in a Σ-algebra $\mathcal{A} = (\mathbf{A}, \{f^{\mathcal{A}} \mid f \in \Sigma\})$ naturally induces a *congruence* relation $=_{\mathcal{A}}$ over $\mathbb{T}(\Sigma)$, which is defined as:

$$t =_{\mathcal{A}} u \text{ if, and only if, } \rho(t) = \rho(u), \text{ for each environment } \rho \ .$$

The results presented in the first part of the dissertation all aim at using the classical logic of equality to offer a syntactic characterization of the relation $=_{\mathcal{A}}$ for algebras of processes. The study of such axiomatic characterizations of semantic equivalences (or preorders) falls therefore within the realm of (in)equational logic, whose basics we now proceed to present.

Equational Logic. An *equational axiom system* is a collection E of equations $t \approx u$ and an *inequational axiom system* is a collection of inequations $t \preccurlyeq u$, both over the language $\mathbb{T}(\Sigma)$. The (in)equations in E are often referred to as *axioms*. An equation $t \approx u$ is derivable from an equational axiom system E, notation $E \vdash t \approx u$, if it can be proven from the axioms in E using the rules of equational logic (viz. reflexivity, symmetry, transitivity, substitution and closure under Σ-contexts), as in Tab. 2.1.

$$\frac{}{t \approx t} \qquad \frac{t \approx u}{u \approx t} \qquad \frac{t \approx u \quad u \approx v}{t \approx v} \qquad \frac{t \approx u}{\sigma(t) \approx \sigma(u)}$$

$$\frac{t_i \approx u_i \quad (1 \leq i \leq n)}{f(t_1,\ldots,t_n) \approx f(u_1,\ldots,u_n)}$$

Table 2.1: Equational logic

The first three rules above state that $\approx$ is an equivalence relation, whereas the latter two state that $\approx$ is closed under substitutions, and is a congruence. For the derivation of an *inequation* $t \preccurlyeq u$ from an *inequational* axiom system E, denoted by $E \vdash t \preccurlyeq u$, the second rule, for *symmetry*, is omitted. Below, for simplicity, we only speak the *equational* case while the *inequational* case can be defined accordingly.

Formally, a proof of an equation $t \approx u$ from E is a sequence $t_i \approx u_i$ $(1 \leq i \leq n)$ of equations such that

- $t_n = t$ and $u_n = u$, and
- for each $1 \leq i \leq n$, the equation $t_i \approx u_i$ is obtained by applying one of the aforementioned inference rules using equations in E or some of the equations that precede it in the sequence as premises.

Without loss of generality one may assume that the substitution rule is only applied to axioms, i.e., that the rule

$$\frac{t \approx u}{\sigma(t) \approx \sigma(u)}$$

may only be used when $(t \approx u) \in E$. In this case, the equation $\sigma(t) \approx \sigma(u)$ is called a *substitution instance* of an axiom in E.

Moreover, by postulating that for each axiom in E also its symmetric counterpart is present in E, one may assume that there are no applications of the symmetry rule in equational proofs. It is also well-known (see, e.g., Section 2 in [Gro90]) that if an equation relating two *closed* terms can be proven from an axiom system E, then there is a closed proof for it. These facts can be used to simplify proofs by induction on equational derivations. Let E' be the collection of equations that consists of all substitution instances of the axioms in E and their symmetric variants, i.e.,

$$E' = \{\rho(t) \approx \rho(u) \mid (t \approx u) \in E \text{ or } (u \approx t) \in E,\ \rho \text{ a substitution}\} \ .$$

By a *normalized derivation* of an equation $t \approx u$ from E we shall henceforth mean a derivation of the equation $t \approx u$ from E' by means of the rules of equational logic but not using the symmetry and substitution rules. Now if $E \vdash t \approx u$, then there exists a normalized derivation of $t \approx u$ from E.

Definition 2.1.7 (Soundness) Let $\mathcal{A}$ be a Σ-algebra. An equation $t \approx u$ is *sound* with respect to $=_{\mathcal{A}}$ iff $t =_{\mathcal{A}} u$. An axiom system is sound with respect to $=_{\mathcal{A}}$ iff so is each of its equations.

The collection of *all* equations that are sound with respect to $=_{\mathcal{A}}$ is called the *equational theory* of $\mathcal{A}$.

In other words, an axiom system is sound with respect to $=_{\mathcal{A}}$ if it can only be used to prove equations that are valid in the algebra $\mathcal{A}$. This is, of course, a most natural requirement on an axiom system. However, ideally an axiom system should also allow us to prove all of the equations that hold in a given algebra. This is captured by the technical requirement of *completeness*.

Definition 2.1.8 (Completeness) Let $\mathcal{A}$ be a Σ-algebra. An axiom system E is *ground-complete* with respect to $=_{\mathcal{A}}$ iff $E \vdash t \approx u$ whenever $t =_{\mathcal{A}} u$, for all *closed* terms t, u.

E is *complete* with respect to $=_{\mathcal{A}}$ iff $E \vdash t \approx u$ whenever $t =_{\mathcal{A}} u$, for all (open) terms t, u.

Definition 2.1.9 (Equational bases, finitely based algebras) An *equational basis* for an algebra $\mathcal{A}$ is a sound axiom system E that is complete with respect to $=_{\mathcal{A}}$. We say that an algebra $\mathcal{A}$ is *finitely based* if it has a finite equational basis.

The notion of completeness of an axiom system relates the *proof-theoretic* notion of derivability using the rules of equational logic with the *model-theoretic* one of "validity in a model". From a proof-theoretic perspective, a useful property of an axiom system E is that, for all terms $t, u \in \mathbb{T}(\Sigma)$,

$$E \vdash t \approx u \text{ iff } E \vdash \sigma(t) \approx \sigma(u), \text{ for each closed substitution } \sigma \ . \tag{2.1}$$

An axiom system with the above property is called *ω-complete*. In theorem proving applications, it is convenient if an axiomatization is ω-complete, because this means that proofs by (structural) induction can be avoided in favor of purely equational reasoning; see [LLT90]. In [Hee86] it was argued that ω-completeness is desirable for the partial evaluation of programs. In fact, suppose that $\sigma(t) \approx \sigma(u)$ is provable from an axiom system E, for each closed substitution σ. If E is ω-complete, then we know that an equational proof of the actual equation $t \approx u$ from E exists. However, in general the equation $t \approx u$ might not be derivable from E if E is just ground-complete. In that case, we might have to content ourselves with showing that all closed instantiations of that equation are derivable from E, and this is usually done by induction on the structure of the closed terms that can be substituted for the variables occurring in t and u. In the computer science community, notable examples of ω-incomplete axiomatizations in the literature are the $\lambda K\beta\eta$-calculus [Plo74] and the equational theory of CCS [Mol90a]. Therefore, laws such as commutativity of parallelism, which are valid in the initial model but which cannot be derived, are often added to the latter equational theory. For such extended equational theories, ω-completeness results were presented in the setting of CCS [Mol89, AFIL09] and ACP [FL00]. Moreover, [Mil84, Mil89b] presented pioneering work regarding ω-complete axiomatization (although not equational) for CCS.

It turns out that completeness and ω-completeness are closely related properties of an axiom system. Indeed, assume that $\mathcal{A}$ is a Σ-algebra each of whose elements is denotable. Suppose that E is sound and complete with respect to $=_{\mathcal{A}}$. It is not hard to argue that, in this case, E is also ω-complete. Moreover, consider the Σ-algebra obtained by quotienting the set of closed terms $\mathrm{T}(\Sigma)$ with respect to the congruence relation that equates two closed terms t, u iff the equation $t \approx u$ is provable from an axiom system E. We have that an equational basis for that algebra is also ω-complete.

2.1.4 Process Algebras BCCSP and BCCS

BCCSP and BCCS[2] are two universal algebras, specific to the realm of process algebras where they play a fundamental role.

[2]The name of BCCSP was coined by van Glabbeek [vG90], abbreviating **B**asic **CCS** and **CSP**. Accordingly, BCCS abbreviates **B**asic **CCS**.

BCCSP is a common fragment of CCS and CSP for describing finite synchronization trees [Mil89a]. Its signature consists of the constant $\mathbf{0}$, the binary operator $\_ + \_$, and unary prefix operators $a\_$, where a ranges over a nonempty set A of actions, called the *alphabet*, with typical elements a, b, c. In other words, the language is given by the following grammar:

$$t ::= \mathbf{0} \mid at \mid t + t \mid x \ .$$

Intuitively, closed BCCSP terms, denoted by p, q, r, represent finite process behaviors, where $\mathbf{0}$ does *not* exhibit any behavior, $p + q$ offers a choice between the behaviors of p and q, and ap executes action a to transform into p. This intuition is captured by the transition rules given in Tab. 2.2 (see [AFV01] for an extensive introduction). They give rise to A-labeled transitions between closed BCCSP terms, which are of the form (p, a, p'), where p, p' are closed terms and $a \in A$. Henceforth, as usual, we shall use the suggestive notation $p \xrightarrow{a} p'$ in lieu of (p, a, p'). A *transition relation* is a collection of A-labeled transitions. It is well-known that the transition relation $\longrightarrow$ is the one defined by structural induction over closed terms using the above rules.

$$\frac{}{ax \xrightarrow{a} x} \qquad \frac{x_1 \xrightarrow{a} y}{x_1 + x_2 \xrightarrow{a} y} \qquad \frac{x_2 \xrightarrow{a} y}{x_1 + x_2 \xrightarrow{a} y}$$

Table 2.2: SOS for BCCSP

We also assume a countably infinite set V of variables; x, y, z denote elements of V, and X, Y, Z denote finite subsets of V. Open BCCSP terms, which may contain variables from V, are denoted by t, u, v, w. Following the convention of the previous section, we use T(BCCSP) and $\mathbb{T}$(BCCSP) to denote the collection of closed and open BCCSP terms respectively.

In order to study weak semantics, BCCSP can be extended with the unary prefix operator $\tau\_$, which constitutes BCCS. In other words, its signature includes, in addition to BCCSP, the unary prefix operator $\tau\_$ and the language is thus given by the following grammar:

$$t ::= \mathbf{0} \mid \alpha t \mid t + t \mid x \ .$$

Closed BCCS terms are ranged over by p, q, r as well and share the same intuition as BCCSP terms. In particular, τp executes action τ to transform into p. The operational semantics can be specified smoothly in Tab. 2.3, which gives rise to A_τ-labeled transitions between closed BCCS terms. Accordingly, we use T(BCCS) and $\mathbb{T}$(BCCS) to denote the collection of closed and open BCCS terms respectively.

It is technically convenient to extend the operational semantics to open BCCS(P) terms. There are two options: either (1) we do not include additional rules for variables, which effectively means that they do not exhibit any

$$\frac{}{\alpha x \xrightarrow{\alpha} x} \qquad \frac{x_1 \xrightarrow{\alpha} y}{x_1 + x_2 \xrightarrow{\alpha} y} \qquad \frac{x_2 \xrightarrow{\alpha} y}{x_1 + x_2 \xrightarrow{\alpha} y}$$

Table 2.3: SOS for BCCS

behavior; or (2) we treat each variable occurrence x in a term as if it were a subterm $x\mathbf{0}$ with x a concrete action. For open BCCSP (resp. BCCS) terms t and u, and a preorder $\precsim$ (or equivalence $\simeq$) on closed BCCSP (resp. BCCS) terms, we define $t \precsim u$ (or $t \simeq u$) if $\rho(t) \precsim \rho(u)$ (resp. $\rho(t) \simeq \rho(u)$) for all closed substitutions ρ. Note that one can alternatively define $\precsim$ (or $\simeq$) directly over open terms via (2), and it is not very difficult to verify that for BCCS(P), both of these two ways yield the same preorder (or equivalence) for each semantics, unless the underlying alphabet $A = \emptyset$, which is not interesting.

A *context* $C[\cdot]$ of BCCSP (resp. BCCS) is a closed BCCSP (resp. BCCS) term with exactly one occurrence of a hole $[\cdot]$ in it. For every context $C[\cdot]$ and closed term p, we write $C[p]$ for the closed term that results by placing p in the hole in $C[\cdot]$.

The preorders $\precsim$ in the linear time – branching time spectrum I are all *precongruences* with respect to BCCSP, meaning that $p_1 \precsim q_1$ and $p_2 \precsim q_2$ imply $p_1 + p_2 \precsim q_1 + q_2$ and $ap_1 \precsim aq_1$ for $a \in A$. Recall that the *kernel* of a preorder $\precsim$ is an equivalence $\simeq = \precsim \cap \precsim^{-1}$. Likewise, the equivalences in the spectrum are all *congruences* with respect to BCCSP.

Similarly, $\preceq_\Box$ for $\Box \in \{\mathrm{WT}, \mathrm{WCT}, \mathrm{WF}, \mathrm{WIF}\}$ are *precongruences* with respect to BCCS. And the corresponding equivalences $\simeq_\Box$ are *congruences* for BCCS.

Remark 2.1.10 The requirements (2) in Def. 2.1.6 are called *root conditions*, which are imposed for weak semantics routinely to make sure that the preorder under consideration is a precongruence (see [Mil89a]). Moreover, the requirement (3) $\mathcal{CT}(s) = \mathcal{CT}(s')$ for the weak impossible futures preorder is indispensable because otherwise that definition would fail to yield a precongruence for BCCS. For instance, in that case we would have $\tau a\mathbf{0} \preceq_{\mathrm{WIF}} \tau a\mathbf{0} + b\mathbf{0}$. However, clearly $c(\tau a\mathbf{0}) \not\preceq_{\mathrm{WIF}} c(\tau a\mathbf{0} + b\mathbf{0})$.

The core axioms A1-4 [Mil89a] for BCCSP given in Tab. 2.4 are ω-complete, and sound and ground-complete modulo bisimulation equivalence. Since every equivalence in the linear time – branching time spectrum I (see Fig. 2.1) includes bisimulation equivalence, and each weak semantics is coarser than its concrete counterpart, it follows that the axioms A1-4 are sound modulo every equivalence in the spectrum. Furthermore, each of the axioms A1-4 induces two inequations, obtained by replacing $\approx$ by $\preccurlyeq$ or $\succcurlyeq$, that are both sound modulo every preorder in the linear time – branching time spectrum. We write $t = u$ if terms t and u are equal modulo A1-4, namely, the associativity, commutativity

A1	$x + y$	$\approx$	$y + x$
A2	$(x + y) + z$	$\approx$	$x + (y + z)$
A3	$x + x$	$\approx$	x
A4	$x + \mathbf{0}$	$\approx$	x

Table 2.4: Core axioms A1-4

and idempotence of $+$, and the absorption of $\mathbf{0}$ summands. For every preorder $\precsim$ and equivalence $\simeq$ in the linear time – branching time spectrum, soundness of the axioms A1-4 ensures that whenever we write $t = u$, then also $t \precsim u$ and $t \simeq u$. Furthermore, we will (tacitly) assume that the axioms A1-4 above are included in every axiomatization E considered in what follows, so that from $t = u$ we may always conclude $t \preccurlyeq u$ and $t \approx u$.

Notions on process terms. The *depth* of a BCCSP term t, denoted by $depth(t)$, is the length of the longest trace that t can exhibit, i.e.,

$$depth(t) = \max\{k \mid \exists a_1 \cdots a_k, t'.t \xrightarrow{a_1 \cdots a_k} t'\} \ .$$

Alternatively it is defined inductively as follows: $depth(\mathbf{0}) = depth(x) = 0$; $depth(at) = 1 + depth(t)$; and $depth(t + u) = \max\{depth(t), depth(u)\}$.

Let $k \geq 0$. If $t \xrightarrow{a_1 \cdots a_k} t'$ for some sequence of actions $a_1 \cdots a_k$, and t' has the variable x as a summand, then we say that x *occurs in* t *at depth* k. The set of variables with an occurrence in t at depth k will be denoted by $var_k(t)$; the set of *all* variables with an occurrence in t will be denoted by $var(t)$. Similarly, if $t \xrightarrow{a_1 \cdots a_k} t'$ for some sequence of actions $a_1 \cdots a_k$, and the action a is an element of $\mathcal{I}(t')$, then we say that a *occurs in* t *at depth* k. The set of actions with an occurrence in t at depth k will be denoted by $act_k(t)$.

For BCCS terms, $t \xrightarrow{\alpha}$ denotes that there is a term u with $t \xrightarrow{\alpha} u$, and likewise $t \Rightarrow\!\!\xrightarrow{\alpha}$ denotes that there are a terms u, v with $t \Rightarrow u \xrightarrow{\alpha} v$. The *depth* of a term t, denoted by $depth(t)$, is the length of the *longest* trace of that t can exhibit, *not counting* τ*-transitions*. It is defined inductively as follows: $depth(\mathbf{0}) = depth(x) = 0$; $depth(at) = 1 + depth(t)$; $depth(\tau t) = depth(t)$; and $depth(t + u) = \max\{depth(t), depth(u)\}$.

Given any equation $t \approx u$ or inequation $t \preccurlyeq u$, we define its *depth* as $\max\{depth(t), depth(u)\}$.

In general, let $\{t_1, \ldots, t_n\}$ be a finite set of terms; we use summation $\sum\{t_1, \ldots, t_n\}$ to denote $t_1 + \cdots + t_n$, adopting the convention that the summation of the empty set denotes $\mathbf{0}$. A term t is called a *prefix* if $t = \alpha t'$ for some $\alpha \in A_\tau$ and for some term t'. We write $\alpha^n t$ to denote the term obtained from t by prefixing it n times with α, i.e., $\alpha^0 t = t$ and $\alpha^{n+1} t = \alpha(\alpha^n t)$. When writing terms, we adopt as binding convention that $\_ + \_$ and summation bind weaker than $\alpha\_$. With abuse of notation, we often let a finite set of variables

X denote the term $\sum_{x \in X} x$. Note that, with the above notational conventions, for every BCCSP term t there always exist a unique finite family of actions $\{a_i \mid i \in I\} \subseteq A$, terms $\{t_i \mid i \in I\}$, and a finite set of variables $X \subseteq V$ such that

$$t = \sum_{i \in I} a_i t_i + X \ ,$$

while for every BCCS term t can be further written as

$$t = \sum_{i \in I} a_i t_i + \sum_{j \in J} \tau t_j + X \ ,$$

with a finite family of terms $\{t_j \mid j \in J\}$. A term t is called a *summand* of u (notation: $t \Subset u$) if it is a variable or a prefix and $u = u + t$.

2.1.5 Two Proof Techniques

We give a short introduction to two proof techniques that will be exploited extensively in the first part of the dissertation. The first technique, called *cover equations*, is especially designed for BCCSP, while the second technique, which is based on proof theory, is more generally applicative.

Cover equations. This technique, which is introduced in [FN04], aims to obtain an explicit description of the equational theory of BCCSP modulo some equivalence.

The central idea is that if an equation $t \approx u$ is sound for BCCSP modulo some equivalence in the linear time – branching time spectrum I, then $u + t \approx t$ and $t + u \approx u$ are sound as well; and from the last two equations one can derive $t \approx u$. Therefore, to extend an axiomatization consisting of A1-4 to a complete axiomatization of some equivalence in the linear time – branching time spectrum I, it suffices to add sound equations of the form $x + u \approx u$ and $at + u \approx u$; such equations are called *cover equations*.

In order to further limit the form of the cover equations that need to be considered, one usually tries to establish the following properties for the equivalence $\simeq$ at hand:

1. If $at + u + bv \simeq u + bv$ with $a \neq b$, then $at + u \simeq u$.
2. If $t \simeq u$, then t and u contain the same variables, at the same depths.
3. If $t + x \simeq u + x$, and x is not a summand of $t + u$,[3] then $t \simeq u$.

If the properties above hold, then it suffices to only consider cover equations of the form $at + au_1 + \cdots + au_n \approx au_1 + \cdots + au_n$.

As we will see in Chapter 4, the second property holds for all equivalences finer than or as fine as trace equivalence, in case $|A| > 1$. The first and third

[3] To see that this side condition is needed, note that, in general, $x + x \simeq \mathbf{0} + x$ but $x \not\simeq \mathbf{0}$.

properties have to be proved for each equivalence separately. Proving the first property is generally easy, but proving the third property can be a challenge (see Section 4.3 and [AFI07]).

When the cover equations have been identified, one can proceed in two ways. Either one can determine a finite basis among the cover equations, or one can determine an infinite family of cover equations that obstructs a finite basis. We will follow the latter approach in Section 4.5, considering only equations of depth at most one, for congruences that are finer than or as fine as ready equivalence and coarser than or as coarse as possible worlds equivalence. Moreover, the cover equations technique turned out to be helpful in finding the infinite families of equations that obstruct a finite basis in Chapter 4 and Chapter 5.

Proof-theoretic technique. To prove that no finite basis exists for an equivalence $\simeq$, it suffices to provide an infinite family of equations $t_n \approx u_n$ $(n = 1, 2, 3, \dots)$ that are all sound modulo $\simeq$, and to associate with every finite set of sound equations E a property P_E that holds for all equations derivable from E, but does *not* hold for at least one of the equations $t_n \approx u_n$. It then follows that for every finite set of sound equations E there exists a sound equation $t_n \approx u_n$ that is not derivable from E. It follows that every finite set of sound equations is necessarily incomplete, and hence $\simeq$ is not finitely based.

We shall apply this proof-theoretic technique in Section 4.4 with Sections 4.6–4.8, in Section 5.2.3 and Section 5.3 (with straightforward adaption for preorders), and in Section 6.5.1. In each case we proceed in the following three steps (usually with separated lemmas):

1. We provide an infinite family of sound equations $t_n \approx u_n$ $(n = 1, 2, 3, \dots)$ and a suitable family of properties P_n $(n = 1, 2, 3, \dots)$ such that the property P_n fails for all the equations $t_i \approx u_i$ with $i \geq n$.

2. We establish that the property P_n holds for every substitution instance of any sound equation $t \approx u$ with $\mathit{depth}(t), \mathit{depth}(u) \leq n$.

3. We prove that P_n holds for every equation derivable from a collection E of sound equations $t \approx u$ with $\mathit{depth}(t), \mathit{depth}(u) \leq n$; the latter proof is by induction on normalized derivations, using (2) for the base case.

2.2 Background for Part II

2.2.1 Preliminaries for Probability Theory

The development of Part II needs some basic concepts from measure theory, in particular probability spaces and σ-algebras. Below we give a short summary. For further details, one can consult, e.g. [AD00, Bil95]; [Pan01] offers a gentle introduction for computer scientists, in particular for concurrency theorists.

Definition 2.2.1 A *Borel space* is a pair $(\mathcal{O}, \mathfrak{C})$, where $\mathcal{O}$ is a nonempty set, and $\mathfrak{C} \subseteq 2^{\mathcal{O}}$ is a *σ-algebra* associated with $\mathcal{O}$. Namely, it is a set consisting of

subsets of $\mathcal{O}$ which contains the empty set $\emptyset$ and is closed under *complementation* and *countable union*, i.e.,

- $\emptyset \in \mathfrak{C}$;
- $C \in \mathfrak{C}$ implies that $\overline{C} = \mathcal{O} \setminus C \in \mathfrak{C}$; and
- $C_1, C_2, \cdots \in \mathfrak{C}$ implies that $\bigcup_{i \geq 1} C_i \in \mathfrak{C}$.

Occasionally, the set $\mathcal{O}$ is supposed to be fixed and $\mathfrak{C}$ by itself is called a σ-algebra. The elements of $\mathcal{O}$ are often called *outcomes*, while the elements of $\mathfrak{C}$ are called *events*.

A *probability measure* on $(\mathcal{O}, \mathfrak{C})$ is a function $\mathbb{P} : \mathfrak{C} \to [0, 1]$ such that $\mathbb{P}(\mathcal{O}) = 1$, and if $(C_n)_{n \geq 1}$ is a family of pairwise disjoint events $C_n \in \mathfrak{C}$, then:

$$\mathbb{P}(\bigcup_{n \geq 1} C_i) = \sum_{n \geq 1} \mathbb{P}(C_n) \ .$$

A *probability space* is a σ-algebra equipped with a probability measure (thus it is a special case of *measure space*), i.e., it is a triple $(\mathcal{O}, \mathfrak{C}, \mathbb{P})$ where $(\mathcal{O}, \mathfrak{C})$ is a σ-algebra and $\mathbb{P}$ is a probability measure on $(\mathcal{O}, \mathfrak{C})$. For an event $E \in \mathfrak{C}$, the value $\mathbb{P}(E)$ is called the probability measure of E, or simply the probability of E. In the context of probability measures, the events (i.e., the elements of $\mathfrak{C}$) are often said to be *measurable*.

Given a *countable* set S, we denote the set of (finite) discrete probability distributions over S by $\mathit{Distr}(S)$. Namely, each $\mu \in \mathit{Distr}(S)$ is a function $\mu{:}S \to [0, 1]$ such that $\sum_{s \in S} \mu(s) = 1$ and the *support* of μ, $\mathit{Supp}(\mu) = \{s \in S \mid \mu(s) > 0\}$, is finite. The *Dirac distribution* μ_s^1 is the discrete probability distribution such that $\mu_s^1(s) = 1$ and $\mu_s^1(t) = 0$ for $t \neq s$.

2.2.2 Discrete-time Markov Chains

Discrete-time Markov chains (DTMCs) behave as transition systems with the only difference that nondeterministic choices among successor states are replaced by probabilistic ones. That is to say, the successor state of a state s is chosen according to a probability distribution. This probability distribution only depends on the current state s, and not on, e.g., the path fragment that led to state s from some initial state. Accordingly, the system evolution does not depend on the history (i.e., the path fragment that has been executed so far), but only on the current state s. This is known as the *memoryless property*. We refer the readers to, among others, [KSK76] as a standard textbook.

Let AP be a fixed, finite set of *atomic propositions* on states ranged over by $a, b, c, \ldots$.

Definition 2.2.2 (DTMCs) A (labeled) *discrete-time Markov chain* (DTMC) is a triple $\mathcal{D} = (S, \mathrm{AP}, L, \alpha, \mathbf{P})$, where:

- S is a (countable) set of states;
- $L : S \to 2^{\mathrm{AP}}$ is a labeling function which assigns to each state $s \in S$ the set $L(s)$ of atomic propositions that are valid in s;
- $\alpha \in \mathit{Distr}(S)$ is the initial distribution; and
- $\mathbf{P} : S \times S \to [0,1]$ is a *stochastic matrix*, i.e., for each state $s \in S$, $\sum_{t \in S} \mathbf{P}(s,t) = 1$.

$\mathcal{D}$ is called *finite* if S and AP are finite.

Intuitively, a DTMC is a Kripke structure in which all transitions are equipped with discrete probabilities such that the sum of outgoing transitions of each state equals one. A state s in $\mathcal{D}$ is called *absorbing* if $\mathbf{P}(s,s) = 1$. Without loss of generality, we often assume a DTMC to have a unique initial state, i.e. the initial distribution α is a Dirac distribution.

Definition 2.2.3 (Paths) Let $\mathcal{D} = (S, \mathrm{AP}, L, \mathbf{P})$ be a DTMC.

- An *infinite path* ρ in $\mathcal{D}$ is an infinite sequence $s_0 \cdot s_1 \cdot s_2 \cdots$ of states such that $\forall i \geq 0.\ \mathbf{P}(s_i, s_{i+1}) > 0$.
- A *finite path* σ is a prefix of an infinite path.

Let $\mathit{Paths}^{\omega}_{\mathcal{D}}(s)$ denote the set of all *infinite* paths in $\mathcal{D}$ that start in state s and $\mathit{Paths}^{\star}_{\mathcal{D}}(s)$ denote the set of all *finite* paths of s. The subscript $\mathcal{D}$ is omitted when it is clear from the context. For state s and finite path $\sigma = s_0 \cdots s_n$ with $\mathbf{P}(s_n, s) > 0$, let $\sigma \cdot s$ denote the path obtained by extending σ by s.

Let ρ denote either a finite or infinite path. Let $|\rho|$ denote the length of ρ, i.e., $|s_0 \cdot s_1 \cdots s_n| = n$, $|s_0| = 0$ and $|\rho| = \infty$ for infinite ρ. For finite ρ and $0 \leq i \leq |\rho|$, $\rho[i]$ denotes the $(i{+}1)$-st state in ρ. For $i \leq |\rho|$, we use $\rho{\downarrow_i}$ to denote the prefix of ρ truncated at length i (thus ending in s_i), formally, $\rho{\downarrow_i} = \rho[0] \cdot \rho[1] \cdots \rho[i]$. We use $\mathit{Pref}(\rho)$ to denote the set of prefixes of ρ, i.e., $\mathit{Pref}(\rho) = \{\rho{\downarrow_i} \mid 0 \leq i \leq |\rho|\}$.

Cylinder set. In order to be able to associate probabilities to events in DTMCs, the intuitive notion of probabilities in DTMC $\mathcal{D}$ is formalized by associating a probability space with $\mathcal{D}$. The set $\mathcal{O}$ of outcomes consists of the infinite paths of $\mathcal{D}$. To define an appropriate σ-algebra for $\mathcal{D}$, we will use the fact that for each σ-algebra $(\mathcal{O}, \mathfrak{C})$ and each subset Π of $2^{\mathcal{O}}$, there exists a *smallest σ-algebra* that contains Π. This is due to the facts that (1) $2^{\mathcal{O}}$ itself is a σ-algebra; and (2) the intersection of σ-algebras is a σ-algebra. Thus the intersection $\mathfrak{C}_{\Pi} = \bigcap_{\mathfrak{C}} \mathfrak{C}$, where $\mathfrak{C}$ ranges over all σ-algebras on $\mathcal{O}$ that contain Π, is a σ-algebra and is contained in any σ-algebra $\mathfrak{C}$ such that $\Pi \subseteq \mathfrak{C}$. $\mathfrak{C}_{\Pi}$ is called the σ-algebra *generated* by Π, and Π is the *basis* for $\mathfrak{C}_{\Pi}$. The σ-algebra associated with $\mathcal{D}$ is generated by the *cylinder sets* (a.k.a. basic cylinder set) spanned by the finite path fragments in $\mathcal{D}$.

Definition 2.2.4 (Cylinder set) For $\rho \in Paths^\star$, the cylinder set of ρ is defined as

$$\Delta(\rho) = \{\pi \in Paths^\omega \mid \rho \in Pref(\pi)\} \ .$$

Namely, this cylinder set (spanned by finite path ρ) consists of all infinite paths that start with ρ.

We are now in a position to define the probability space induced by a DTMC $\mathcal{D}$. The underlying σ-algebra is defined as the smallest σ-algebra that contains all the cylinder sets induced by the finite paths starting in the initial state s_0. It follows from classical concepts of probability theory (see e.g. [AD00]), Carathéodory's extension theorem in particular, that there exists a unique probability measure $\Pr_{s_0}^{\mathcal{D}}$ (or, briefly Pr) on the σ-algebra associated with $\mathcal{D}$ where the probabilities for the cylinder sets are given by:

$$\Pr(\{\underbrace{\rho \in Paths^\omega_{\mathcal{D}}(s_0) \mid \rho{\downarrow_n} = s_0 \cdots s_n}_{\Delta(s_0 \cdots s_n)}\}) = \prod_{0 \leq i < n} \mathbf{P}(s_i, s_{i+1}) \ .$$

The probability of finite path $\sigma = s_0 \cdots s_n$ is defined as $\mathbb{P}(\sigma) = \prod_{0 \leq i < n} \mathbf{P}(s_i, s_{i+1})$. Note that although $\Pr(\Delta(\sigma)) = \mathbb{P}(\sigma)$, they have different meanings: Pr is a probability measure on *infinite* paths whereas $\mathbb{P}$ refers to finite ones. For a set C of finite paths which is *prefix containment free*, i.e., for any $\sigma, \sigma' \in C$ with $\sigma \neq \sigma'$, $\sigma \notin Pref(\sigma')$, the probability of C is $\mathbb{P}(C) = \sum_{\sigma \in C} \mathbb{P}(\sigma)$, and paths in C induce disjoint cylinder sets.

2.2.3 Timed Automata

Throughout of the second part of the dissertation, we use $\mathbb{N}$, $\mathbb{Q}_{\geq 0}$ and $\mathbb{R}_{\geq 0}$ to denote the sets of naturals, nonnegative rationals and nonnegative reals, respectively.

Clocks. We consider a finite set $\mathcal{X}$ of real-valued *clocks*. A *clock valuation* over $\mathcal{X}$ is a function $\nu : \mathcal{X} \to \mathbb{R}_{\geq 0}$ which assigns to each clock a time value in $\mathbb{R}_{\geq 0}$. We write $\mathbb{R}^{\mathcal{X}}_{\geq 0}$ for the set of valuations over $\mathcal{X}$. For every $\nu \in \mathbb{R}^{\mathcal{X}}_{\geq 0}$, the valuation $\nu + t$ is defined as $(\nu + t)(x) = \nu(x) + t$ for any $x \in \mathcal{X}$. For $X \subseteq \mathcal{X}$, we write $\nu[X := 0]$ for the valuation which maps each clock in X to the value 0 and agrees with ν over $\mathcal{X} \backslash X$.

Clock constraints. We use $\mathcal{B}(\mathcal{X})$ to denote the set of formulae defined by the grammar

$$g ::= x \bowtie c \mid x - y \bowtie c \mid g \wedge g \ ,$$

where $x, y \in \mathcal{X}$, $\bowtie \in \{\leq, <, =, \geq, >\}$ and $c \in \mathbb{N}$. The elements of $\mathcal{B}(\mathcal{X})$ are called *clock constraints*. An *atomic constraint* does not contain any conjunctions. We also consider (in Chapter 7) a proper subset of *diagonal-free* clock constraints, where constraints of the form $x - y \bowtie c$ are not allowed. This restricted set of constraints is called diagonal-free because constraints of the form $x - y \bowtie c$ are

called *diagonal* clock constraints. We note that TA with diagonal constraints are not more expressive than diagonal-free TA [BPDG98], but they might incur an exponential blowup [BC05] and require a very careful treatment [Bou04].

Clock constraints are interpreted over *valuations* for $\mathcal{X}$. For clock valuation ν and time constraint g, we write $\nu \models g$ meaning ν satisfies g, which is defined by: $\nu \models x \bowtie c$ if $\nu(x) \bowtie c$, $\nu \models x - y \bowtie c$ if $\nu(x) - \nu(y) \bowtie c$, and $\nu \models g_1 \wedge g_2$ if $\nu \models g_1$ and $\nu \models g_2$. We also write $[\![g]\!] = \{\nu \in \mathbb{R}^{\mathcal{X}} \mid \nu \models g\}$, i.e., $[\![g]\!]$ is the set of valuations satisfying g.

An *$\mathcal{X}$-hyperplane* is a set of valuations satisfying an atomic constraint. The class of *$\mathcal{H}_{\mathcal{X}}$-polyhedra* is defined as the smallest subset of $2^{\mathbb{R}^{\mathcal{X}}}$ which contains all $\mathcal{X}$-hyperplanes and is closed under set union, intersection, and complement. In particular, we remark that polyhedra are a geometrical term for *zones* [HNSY94]. Intersection ($\cap$), union ($\cup$) and complement($^-$) are well-defined operations on polyhedra. Given a polyhedron Z and a subset of clocks $X \subseteq \mathcal{X}$, the operation $Z[X := 0]$ is defined as $\{\nu \in \mathbb{R}^{\mathcal{X}}_{\geq 0} \mid \nu[X := 0] \in Z\}$.

Definition 2.2.5 (Timed automata) A *timed automaton* (TA) is a tuple $\mathcal{A} = (\Sigma, \mathcal{X}, Q, Q_0, Q_F, \rightarrow)$ where:

- Σ is a finite alphabet;
- $\mathcal{X}$ is a finite set of clocks;
- Q is a finite set of locations;
- $Q_0 \subseteq Q$ is a nonempty set of initial locations;
- $Q_F \subseteq Q$ is a set of accepting locations;
- $\rightarrow \in Q \times \Sigma \times \mathcal{B}(\mathcal{X}) \times 2^{\mathcal{X}} \times Q$ is an edge relation; and
- $\mathit{Inv} : Q \rightarrow \mathcal{B}(\mathcal{X})$ is an invariant-assignment function.

We refer to $q \xrightarrow{a,g,X} q'$ as a *transition*, where $a \in \Sigma$ is the input symbol, the *guard* g is a clock constraint on the clocks of $\mathcal{A}$, $X \subseteq \mathcal{X}$ is a set of clocks to be reset, and q' the successor location. The intuition is that the timed automaton $\mathcal{A}$ can move from location q to location q' when the input symbol is a and the guard g holds, while the clocks in X should be reset when entering q'.

Operational semantics. The semantics of a timed automaton is defined by associating an (infinite-state) $(\Sigma \cup \mathbb{R}_{\geq 0})$-labeled transition system (a.k.a. *timed transition system*, or TTS) where the state space is $\{(q, \nu) \mid q \in Q \text{ and } \nu \in [\![g]\!]\}$, and there are two types of transitions:

- Elapse of time: $(q, \nu) \xrightarrow{d} (q, \nu + d)$ if $\nu \in [\![\mathit{Inv}(q)]\!]$ and $\nu + d \in [\![\mathit{Inv}(q)]\!]$ (note that it follows that $\nu + d' \in [\![\mathit{Inv}(q)]\!]$ for any $0 \leq d' \leq d$); and
- Location switch: $(q, \nu) \xrightarrow{a} (q', \nu')$ if there is an edge $q \xrightarrow{a,g,X} q'$ such that $\nu \in [\![g]\!]$, $\nu' = \nu[X := 0]$ and $\nu' \in [\![\mathit{Inv}(q')]\!]$.

In general, we assume an initial state of $\mathcal{A}$ is of the form $(q_0, \vec{0})$ where $q_0 \in Q_0$ and $\vec{0}$ denotes the clock valuation that assigns 0 to all clocks in $\mathcal{X}$.

Part I

Axiomatizability of Process Algebras

Chapter 3

Meta-theories for Axiomatizability

3.1 Introduction

This chapter is mainly devoted to two *meta*-theorems regarding the axiomatizability of BCCSP and BCCS. The general motivation is that we intend to obtain new axiomatizability results from existing ones. The first one concerns a relationship between a *preorder* and its corresponding *equivalence*; The second one concerns a relationship between a *concrete* and its corresponding *weak* semantics. Some more detailed introductions are in order.

Preorder versus equivalence. As said, the lack of consensus on what constitutes an appropriate notion of observable behavior for reactive systems has led to a large number of proposals for behavioral semantics for concurrent processes. These have been surveyed in the linear time-branching time spectrum I for concrete semantics [vG01], and in spectrum II for weak semantics that take into account the internal action τ [vG93b]. Typically, a given semantical notion induces both a *preorder* and an *equivalence*, where the equivalence is the kernel of the corresponding preorder, meaning that two processes are considered equivalent if, and only if, each is a refinement of the other with respect to the preorder.

In the literature, positive and negative axiomatizability results were always proved separately for a preorder and the corresponding equivalence. Recently, Aceto, Fokkink and Ingólfsdóttir [AFI07] showed that for BCCSP such double effort can be avoided, by presenting an algorithm to turn a sound and ground-complete axiomatization of any preorder in the linear time – branching time spectrum I which is at least as coarse as the *ready simulation preorder*, into a sound and ground-complete axiomatization of the corresponding equivalence.[1] Moreover, if the former axiomatization is ω-complete, so is the latter. The requirement that the preorder is at least as coarse as ready simulation is essential; as we shall see in Chapter 5, for impossible futures semantics (which does *not*

[1] Another way to avoid the double effort is by deriving axiomatizations of preorders from those of the corresponding equivalences. This line of research is explored in [dFG09].

satisfy this requirement), there is a finite, ground-complete axiomatization for the preorder, but not for the equivalence.

A serious drawback of the work reported in [AFI07] is that their algorithm requires several properties to hold for the preorders to which it is applied, which have to be checked for each preorder *separately*. Especially their variable cancellation property is usually rather hard to prove, see [AFI08]. Subsequently, de Frutos Escrig, Gregorio Rodríguez and Palomino [dFGP08b, dFGP08a] improved upon this result, so that the algorithm is applicable not only to those preorders specifically mentioned in the "linear time – branching time spectrum I" but to any preorder at least as coarse as the ready simulation preorder, provided it is *initials preserving*, meaning that a preorder relation ($p \sqsubseteq q$) implies inclusion of initial action sets ($I_\tau(p) \subseteq I_\tau(q)$, cf. Def. 3.2.1.) This condition is needed to guarantee soundness of the generated axiomatization.

The first meta-theorem presented in this chapter stems from an effort to apply the algorithm of Aceto *et al.* to weak semantics, which take into account the hidden action τ. The results in [dFGP08b, dFGP08a] do not suffice in this setting, because weak semantics tend to violate the initials preserving condition. So a new round of generalization is needed. To this end, we show that in the setting of BCCS, the algorithm originally proposed in [AFI07] applies equally well to weak semantics; the proviso of initials preserving can be replaced by other conditions. We give three sufficient conditions on the preorder $\sqsubseteq$ and its corresponding equivalence $\equiv$:

1. $p \equiv \tau p$ for all closed terms p;
2. $p \equiv \tau p$ for all p with $I_\tau(p) \neq \emptyset$, and $p \sqsubseteq q$ with $I_\tau(p) \neq \emptyset$ implies $I_\tau(q) \neq \emptyset$;
3. $\tau p \equiv \tau p + p$ for all closed terms p, and $\sqsubseteq$ is *weak initials preserving*, meaning that a preorder relation ($p \sqsubseteq q$) implies inclusion of *weakly* initial action sets ($\mathcal{I}_\tau(p) \subseteq \mathcal{I}_\tau(q)$, cf. Def. 3.2.1).

This makes the algorithm applicable to all 87 preorders surveyed in the "linear time – branching time spectrum II" [vG93b] that are coarser than the (concrete) ready simulation preorder. That is, each of these preorders satisfies either the original initials preserving condition from [dFGP08b, dFGP08a], or one of our three new conditions.

Moreover, we extend the scope of the algorithm to *infinite* processes, by adding recursion constants to BCCS. As an application of both extensions, we provide a ground-complete axiomatization of the CSP failures equivalence, also known as must-testing equivalence, for BCCS processes with divergence.

Concrete versus weak. In process algebras, the dichotomy of concrete and weak semantics is ubiquitous. Typically, a given semantical notion gives rise to both a *concrete* and a *weak* version, where the weak version takes a special treatment of the hidden action τ, which is deemed unobservable. The general idea is to abstract away from internal details of system descriptions. It is well-recognized that to obtain a ground-complete axiomatization for weak semantics

is usually much more difficult than for the concrete semantics. In the literature, this is usually done by adding τ-laws to the existing ground-complete axiomatization for the concrete version. A typical example given by [HM85] is the three τ-laws which lift the ground-complete axiomatization from concrete bisimulation equivalence to weak bisimulation equivalence (observational congruence).

Here we establish a general link between the axiomatizability of concrete and weak semantics partially. Namely for any semantics which is not finer than *failures* or *impossible futures* semantics (cf. Fig. 2.1), we provide an algorithm to turn a sound and ground-complete axiomatization of the *concrete* version into a sound and ground-complete axiomatization of the corresponding *weak* version. Moreover, if the former axiomatization is ω-complete, so is the latter. This result suggests that for positive axiomatizability results, a stronger result is obtained by considering the concrete semantics. On the other hand, negative axiomatizability results become more general if they are proved for the weak counterpart.

As an application of this algorithm, in particular, we derive a ground- and ω-complete axiomatization for the *weak* failures preorder. This preorder plays a prominent role in the process algebra CSP [BHR84]. Moreover, for convergent processes, it coincides with testing semantics [NH84, RV07], and thus its equivalence counterpart is the coarsest congruence for the CCS parallel composition that respects deadlock behavior. Note that a ground-complete axiomatization for the weak failures *equivalence* has appeared in [vG97]. As further applications, we derive complete axiomatizations for weak completed trace and trace semantics. We also mention that in Chapter 5, the link established here will be applied to impossible futures semantics extensively.

In the end, we consider the third meta-theorem, which employs the *inverted substitution* technique to show that an axiomatization is ω-complete. This technique was originally developed by Groote in [Gro90] for *equivalences*; here we adapt it in such a way that it is applicable to *preorders*, which will be exploited in later chapters.

Structure of the chapter. Section 3.2 contains the first meta-theorem regarding a preorder and its corresponding equivalence. Section 3.3 presents the second meta-theorem regarding a concrete and its corresponding weak semantics. Section 3.4 deals with the adaption of the inverted substitution technique. Section 3.5 discusses the related and future work.

3.2 From Preorder to Equivalence

We start from a definition on the *initial actions* of a process term.

Definition 3.2.1 (Initial actions) For any closed term p, the set $I_\tau(p)$ of *strongly initial actions* is $I_\tau(p) = \{\alpha \in A_\tau \mid p \xrightarrow{\alpha}\}$, whereas the set $\mathcal{I}_\tau(p)$ of *weakly initial actions* is $\mathcal{I}_\tau(p) = \{\alpha \in A_\tau \mid p \Rightarrow\!\!\xrightarrow{\alpha}\}$.

A preorder $\sqsubseteq$ is *(strong) initials preserving* if $p \sqsubseteq q$ implies $I_\tau(p) \subseteq I_\tau(q)$ for all p and q; it is *weak initials preserving* if $p \sqsubseteq q$ implies $\mathcal{I}_\tau(p) \subseteq \mathcal{I}_\tau(q)$. With $\mathcal{I}(p)$ we denote $\mathcal{I}_\tau(p) \cap A$, the weakly initial *concrete* actions of p (note that this is in accordance with Def. 2.1.1).

3.2.1 An Algorithm for Producing Equational Axiomatizations

In [AFI07], Aceto, Fokkink and Ingólfsdóttir presented an algorithm that takes as input a sound and ground-complete inequational axiomatization E for BCCSP modulo a precongruence in the linear time – branching time spectrum I that contains the ready simulation preorder, and generates as output an equational axiomatization $\mathcal{A}(E)$ which is sound and ground-complete for BCCSP modulo the corresponding congruence. Moreover, if the original axiomatization E is ω-complete, so is the resulting axiomatization.

The results of [AFI07] were obtained for the language BCCSP. Naturally, these results generalize smoothly to BCCS by treating τ just like any *concrete* action from A. Preorders or equivalences that do so are called *concrete* (or *strong*). Below we rephrase the algorithm in the context of BCCS, in order to smoothen the presentation. The axiomatization $\mathcal{A}(E)$ generated by the algorithm from E contains the axioms A1-4 as well as the axioms:

$$\text{RS}_{\equiv} \qquad \alpha(\beta x + z) + \alpha(\beta x + \beta y + z) \approx \alpha(\beta x + \beta y + z)$$

for $\alpha, \beta \in A_\tau$, that are valid in ready simulation semantics, together with the following equations, for each inequational axiom $t \preccurlyeq u$ in E:

(1) $t + u \approx u$; and

(2) $\alpha(t+x)+\alpha(u+x) \approx \alpha(u+x)$ (for each $\alpha \in A_\tau$, and some variable x that does not occur in $t + u$).

Instead of explicitly adding the axioms $\text{RS}_{\equiv}$ one can equivalently add the axioms

$$\text{RS} \qquad \beta x \preccurlyeq \beta x + \beta y \qquad \text{for } \beta \in A_\tau$$

to E prior to invoking steps (1) and (2) above. Moreover, as observed in [dFGP08b], the conversion from E to $\mathcal{A}(E)$ can be factored into two steps:

- Given an inequational axiomatization E, its *BCCS-context closure* $\overline{E}$ is

$$E \cup \{\alpha(t + x) \preccurlyeq \alpha(u + x) \mid \alpha \in A_\tau \wedge t \preccurlyeq u \in E\} \cup \{\text{RS}\}$$

  where x is a variable not occurring in E.

- Now $\mathcal{A}(E) = \{t + u \approx u \mid t \preccurlyeq u \in \overline{E}\} \cup$ (A1-4).

In [AFI07] the correctness of this algorithm was shown for all precongruences listed in the linear time – branching time spectrum I that are included between trace inclusion and the ready simulation preorder. The proof contained a few arguments that had to be checked for each of these preorders separately. Subsequently, in de Frutos Escrig, Gregorio and Palomino [dFGP08b] the following more general result was obtained:

Theorem 3.2.2 Let $\sqsubseteq$ be an initials preserving precongruence that contains the ready simulation preorder $\precsim_{\mathrm{RS}}$, and let E be a sound and ground-complete axiomatization of $\sqsubseteq$. Then $\mathcal{A}(E)$ is a sound and ground-complete axiomatization of the kernel of $\sqsubseteq$. Moreover, if E is ω-complete, then so is $\mathcal{A}(E)$.

As all preorders in the linear time – branching time spectrum I of [vG01] between trace inclusion and ready simulation are initials preserving, the above theorem strengthens the result of [AFI07].

3.2.2 Correctness Proof of the Algorithm

Below we recreate the proof of Thm. 3.2.2. Lem. 3.2.3 and Prop. 3.2.4 constitute the completeness argument, and are taken directly from [dFGP08b]. However, the proofs below are significantly simpler – in the case of Lem. 3.2.3 employing ideas from the completeness proof in [AFI07]. The essence of Lem. 3.2.5 and its proof come from [dFGP08b] as well; this is the soundness argument. Our rewording of Lem. 3.2.5 allows it to be reused in Sections 3.2.3, 3.2.4 on weak semantics.

Lemma 3.2.3 Let E be an inequational axiomatization. Then for any $t \preccurlyeq u \in E$ and any context $\mathcal{C}[\cdot]$ we have $\mathcal{A}(E) \vdash \mathcal{C}(t) + \mathcal{C}(u) \approx \mathcal{C}(u)$.

Proof: By structural induction on the context $\mathcal{C}[\cdot]$.

In case of the trivial context $\mathcal{C}[\cdot] = \cdot$ we have to show $\mathcal{A}(E) \vdash t + u \approx u$, which follows immediately from step (1) in the construction of $\mathcal{A}(E)$.

For a context $\alpha(\,\cdot + v)$ we have to show $\mathcal{A}(E) \vdash \alpha(t+v)+\alpha(u+v) \approx \alpha(u+v)$, which follows from step (2) in the construction of $\mathcal{A}(E)$, substituting the term v for the variable x.

Now let the result be obtained for a context $\mathcal{D}[\cdot]$ and let $\mathcal{C}[\cdot]$ be of the form $\mathcal{D}[\cdot] + v$, where v is an arbitrary term, possibly $\mathbf{0}$. We have to show that $\mathcal{A}(E) \vdash \mathcal{D}(t) + v + \mathcal{D}(u) + v \approx \mathcal{D}(u) + v$. This follows immediately from the induction hypothesis.

Finally, let the result be obtained for a context $\beta\mathcal{D}[\cdot]$ and let $\mathcal{C}[\cdot]$ be of the form $\alpha(\beta\mathcal{D}[\cdot] + v)$. We have to obtain

$$\mathcal{A}(E) \vdash \alpha(\beta\mathcal{D}(t) + v) + \alpha(\beta\mathcal{D}(u) + v) \approx \alpha(\beta\mathcal{D}(u) + v) \ .$$

By the induction hypothesis we have $\mathcal{A}(E) \vdash \beta\mathcal{D}(t) + \beta\mathcal{D}(u) \approx \beta\mathcal{D}(u)$, so it suffices to obtain

$$\mathcal{A}(E) \vdash \alpha(\beta\mathcal{D}(t) + v) + \alpha(\beta\mathcal{D}(t) + \beta\mathcal{D}(u) + v) \approx \alpha(\beta\mathcal{D}(t) + \beta\mathcal{D}(u) + v) \ .$$

This is an instance of the axiom $\mathrm{RS}_{\equiv}$. ■

Proposition 3.2.4 Let E be an inequational axiomatization. Then whenever $E \vdash t \preccurlyeq u$ we also have $\mathcal{A}(E) \vdash t + u \approx u$.

Proof: If $E \vdash t \preccurlyeq u$ then there is a chain of terms $t_0, \ldots, t_n$ for $n \geq 0$ with $t_0 = t$ and $t_n = u$ such that for $0 \leq i < n$ the inequation $t_i \preccurlyeq t_{i+1}$ is provable from E by one application of an axiom. We now prove the claim by induction on n. The case $n = 0$ is an instance of axiom A3, and the case $n = 1$ is an immediate consequence of Lem. 3.2.3, by applying substitution.

Now for the general case, let v be t_i for some $0 < i < n$. By induction we have $\mathcal{A}(E) \vdash t + v \approx v$ and $\mathcal{A}(E) \vdash v + u \approx u$. Applying once again A3 this yields $\mathcal{A}(E) \vdash t + u \approx t + v + u \approx v + u \approx u$. ■

Lemma 3.2.5 Let $\sqsubseteq$ be a precongruence containing $\precsim_{\mathrm{RS}}$ and $\equiv$ be its kernel. Let p, q be closed terms with $p \sqsubseteq q$ and $I_\tau(p) \subseteq I_\tau(q)$. Then $p + q \equiv q$.

Proof: As $p \sqsubseteq q$ and $\sqsubseteq$ is a precongruence for choice, we have $p + q \sqsubseteq q + q \sqsubseteq q$. To show that $q \sqsubseteq p + q$, let $p \leftrightarrows^2 \sum_{i \in I} \alpha_i p_i$ and $q \leftrightarrows \sum_{j \in J} \beta_j q_j$. It is well known [vG01] that $p \leftrightarrows q$ implies $I_\tau(p) = I_\tau(q)$ as well as $p \precsim_{\mathrm{RS}} q$ and hence $p \sqsubseteq q$. Writing $p|_\beta$ for $\sum_{\alpha_i = \beta} \alpha_i p_i$, the collection of β-summands of p, and likewise $q|_\beta = \sum_{\beta_j := \beta} \beta_j q_j$, we have $p \leftrightarrows \sum_{\beta \in I_\tau(p)} p|_\beta$ and $q \leftrightarrows \sum_{\beta \in I(q)} q|_\beta$. Using that $I_\tau(p) \subseteq I_\tau(q)$, and that $\sqsubseteq$ is a precongruence for the choice operator $+$, it suffices to show that $q|_\beta \sqsubseteq p|_\beta + q|_\beta$ for all $\beta \in A_\tau$. This is an immediate consequence of the axiom RS, which is sound for $\precsim_{\mathrm{RS}}$ and hence for $\sqsubseteq$. ■

Proof of Thm. 3.2.2: As $\sqsubseteq$ is a precongruence contained in the ready simulation preorder, all inequations in the BCCS-context closure $\overline{E}$ of E are sound w.r.t. $\sqsubseteq$. Considering that the soundness of an (in)equation is tantamount to the soundness of its closed substitution instances, the soundness of $\mathcal{A}(E)$ now follows from Lem. 3.2.5.

Ground-completeness and ω-completeness follow directly from Prop. 3.2.4: If $t \equiv u$, that is $t \sqsubseteq u$ and $u \sqsubseteq t$, we have $E \vdash t \preccurlyeq u$ and $E \vdash u \preccurlyeq t$ by the completeness of E. So Prop. 3.2.4 yields $\mathcal{A}(E) \vdash t \approx t + u \approx u$. ■

3.2.3 Applying the Algorithm to Weak Semantics

As said, the results of [AFI07, dFGP08b] were obtained for the language BCCSP. The main purpose of the present section is to apply the ideas from Section 3.2.2 to *weak* preorders: those that in some way abstract from internal activity, by treating τ differently from concrete actions.

When reading Thm. 3.2.2 in the context of weak process semantics, it helps to remember that $\precsim_{\mathrm{RS}}$ is the *concrete* ready simulation preorder, and "initials preserving" refers to preservation of the *strongly* initial actions, where τ is considered as simply another concrete action. Thm. 3.2.2 directly applies to the rooted variants of the η-simulation surveyed in [vG93b], for these preorders are coarser than the concrete ready simulation preorder and strong initials preserving. However, most weak semantics are not strong initials preserving (for

[2] Recall that this denotes the concrete bisimulation equivalence as in Def. 2.1.3.

instance, typically $\tau x \preccurlyeq x$ is sound), and consequently Thm. 3.2.2 fails to apply to them.

The precondition of being initials preserving is in fact nowhere used in the completeness proof in [dFGP08b], or its recreation in Section 3.2.2. Hence, this condition applies to the soundness claim only. Therefore, in order to apply the algorithm to weak semantics, all we need is to find another way of guaranteeing the soundness of the generated axioms.

Given that we deal with preorders containing the ready simulation preorder, the axiom $\mathrm{RS}_{\equiv}$ will always be sound. Moreover, the axioms generated by step (2) in the construction of $\mathcal{A}(E)$ are guaranteed to be sound by Lem. 3.2.5, for we have $\alpha(t+x) \sqsubseteq \alpha(u+x)$ and $I(\alpha(t+x)) = I(\alpha(u+x)) = \{\alpha\}$. One way to guarantee soundness of the remaining axioms, is to check this for each of them explicitly:

Theorem 3.2.6 Let $\sqsubseteq$ be a precongruence that contains the ready simulation preorder $\precsim_{\mathrm{RS}}$, and let E be a sound and ground-complete axiomatization of $\sqsubseteq$, such that for each axiom $t \preccurlyeq u$ in E the law $t + u \approx u$ is sound as well. Then $\mathcal{A}(E)$ is a sound and ground-complete axiomatization of the kernel of $\sqsubseteq$. Moreover, if E is ω-complete, then so is $\mathcal{A}(E)$. ■

Note that for the axioms stemming from $t \preccurlyeq u$ with $I_\tau(\sigma(t)) \subseteq I_\tau(\sigma(u))$ for any closed substitution σ, no check is needed, by Lem. 3.2.5. Next we present three other conditions that guarantee soundness of $\mathcal{A}(E)$.

Theorem 3.2.7 Let $\sqsubseteq$ be a precongruence that contains the concrete ready simulation preorder $\precsim_{\mathrm{RS}}$, such that $p \equiv \tau p$, with $\equiv$ the kernel of $\sqsubseteq$, for all processes p. Let E be a sound and ground-complete axiomatization of $\sqsubseteq$. Then $\mathcal{A}(E)$ is a sound and ground-complete axiomatization of $\equiv$. Moreover, if E is ω-complete, then so is $\mathcal{A}(E)$.

Proof: It suffices to show that $p \sqsubseteq q$ implies $p + q \equiv q$. So assume $p \sqsubseteq q$. Let $p' := \tau p$ and $q' := \tau q$. By assumption we have $p \equiv p'$ and $q \equiv q'$, and therefore $p' \sqsubseteq q'$. As $I_\tau(p') = I_\tau(q') = \{\tau\}$, Lem. 3.2.5 yields $p' + q' \equiv q'$, which implies $p + q \equiv q$. ■

Theorem 3.2.8 Let $\sqsubseteq$ be a precongruence that contains the concrete ready simulation preorder $\precsim_{\mathrm{RS}}$, such that $p \equiv \tau p$, with $\equiv$ the kernel of $\sqsubseteq$, for all processes p with $I_\tau(p) \neq \emptyset$, and such that $p \sqsubseteq q$ implies that if $I_\tau(p) \neq \emptyset$ then $I_\tau(q) \neq \emptyset$. Let E be a sound and ground-complete axiomatization of $\sqsubseteq$. Then $\mathcal{A}(E)$ is a sound and ground-complete axiomatization of $\equiv$. Moreover, if E is ω-complete, then so is $\mathcal{A}(E)$.

Proof: Again it suffices to show that $p \sqsubseteq q$ implies $p + q \equiv q$. So assume $p \sqsubseteq q$. If $I_\tau(p) = \emptyset$ then trivially $I_\tau(p) \subseteq I_\tau(q)$ and the result follows from Lem. 3.2.5. Otherwise, we have $p \equiv \tau p$ and $q \equiv \tau q$ and the result follows as in the previous

proof. ■

Let TAU2 be the second τ-law of CCS [HM85, Mil89a]:

$$\text{TAU2} \qquad \tau x \approx \tau x + x \ .$$

Theorem 3.2.9 Let $\sqsubseteq$ be a weak initials preserving precongruence that contains the concrete ready simulation preorder $\precsim_{\mathrm{RS}}$ and satisfies TAU2, and let E be a sound and ground-complete axiomatization of $\sqsubseteq$. Then $\mathcal{A}(E)$ is a sound and ground-complete axiomatization of the kernel of $\sqsubseteq$. Moreover, if E is ω-complete, then so is $\mathcal{A}(E)$.

Proof: A straightforward induction on the length of a path $p \Rightarrow p'$, using the soundness of TAU2, yields that if $p \Rightarrow p' \stackrel{\alpha}{\longrightarrow} p''$ then $p \equiv p + \alpha p''$, where $\equiv$ is the kernel of $\sqsubseteq$. Hence for any closed term p there is a closed term p' such that $p \equiv p'$ and $\mathcal{I}_\tau(p) = I_\tau(p')$. Using this, the soundness claim follows from Lem. 3.2.5, reasoning as in the proof of Thm. 3.2.7. ■

Note that $\mathcal{I}_\tau(p) = \mathcal{I}(p) \cup \{\tau \mid p \stackrel{\tau}{\longrightarrow}\}$ (see Def. 3.2.1). Thus, the precondition of Thm. 3.2.9 is that $p \sqsubseteq q$ implies that $\mathcal{I}(p) \subseteq \mathcal{I}(q)$ and that if $p \stackrel{\tau}{\longrightarrow}$ then $q \stackrel{\tau}{\longrightarrow}$.

So far, Thm. 3.2.6 applies to the widest selection of preorders, but it comes with the need to check the soundness of some of the generated axioms separately. We can go even further in this direction by observing that also the precondition of containing the ready simulation preorder is not used anywhere in the completeness proof:

Theorem 3.2.10 Let $\sqsubseteq$ be any precongruence, and let E be a ground-complete axiomatization of $\sqsubseteq$. Then $\mathcal{A}(E)$ is a ground-complete axiomatization of the kernel of $\sqsubseteq$. Moreover, if E is ω-complete, then so is $\mathcal{A}(E)$. ■

Note that this theorem makes no statement on the soundness of $\mathcal{A}(E)$. Hence an application of this theorem to achieve a sound and ground-complete axiomatization involves checking the soundness of all axioms generated by both step (1) and step (2) of the algorithm explicitly, as well as the soundness of the axioms A1-4 and $\mathrm{RS}_{\equiv}$. As A1-4 and $\mathrm{RS}_{\equiv}$ constitute a sound and ground-complete axiomatization of concrete ready simulation equivalence, checking the soundness of these axioms is naturally done by checking that the kernel of $\sqsubseteq$ contains concrete ready simulation equivalence. As we shall illustrate in the next section, this is a meaningful improvement over the precondition of Thm. 3.2.6 that $\sqsubseteq$ contains the concrete ready simulation preorder. The price to be paid for this improvement is that also the soundness of the axioms generated by step (2) of the algorithm has to be checked separately. This is because the proof of Lem. 3.2.5 uses that $\sqsubseteq$ contains the concrete ready simulation preorder.

In [vG93b] 155 weak preorders are reviewed. Most of them fail to be congruences for the choice operator of BCCS. Axiomatizations are typically proposed for the *congruence closures* of these preorders: the coarsest congruence

contained in them. All preorders $\sqsubseteq$ in [vG93b] and their congruence closures satisfy the property that if $p \sqsubseteq q$ then $\mathcal{I}(p) \subseteq \mathcal{I}(q)$.[3]

Of the 155 preorders surveyed in [vG93b], 87 contain the concrete ready simulation preorder. We can partition this collection into four classes.

6 preorders are variants of trace inclusion and the simulation preorder. They are precongruences for BCCS and satisfy the axiom $x \approx \tau x$. Consequently, they fall in the scope of Thm. 3.2.7.

16 preorders are variants of completed trace inclusion or the completed simulation preorder. Each of their congruence closures $\sqsubseteq$ has the property that $p \sqsubseteq q$ implies that if $I_\tau(p) \neq \emptyset$ then $I(q) \neq \emptyset$. Moreover, the kernels $\equiv$ of $\sqsubseteq$ have the property that $p \equiv \tau p$ for all processes p with $I_\tau(p) \neq \emptyset$. Consequently, these congruence closures fall in the scope of Thm. 3.2.8.

22 are variants of the η-simulation or the η-ready simulation. Their congruence closures are strong initials preserving, and hence fall under the scope of Thm. 3.2.2.

The congruence closures $\sqsubseteq$ of the remaining 43 preorders satisfy the property that $p \sqsubseteq q$ implies that if $p \xrightarrow{\tau}$ then $q \xrightarrow{\tau}$, and hence $\mathcal{I}_\tau(p) \subseteq \mathcal{I}_\tau(q)$. These precongruences therefore fall in the scope of Thm. 3.2.9.

Thus, the algorithm of [AFI07] applies to all congruence closures of preorders in [vG93b] coarser than the ready simulation preorder.

Applications

In De Nicola and Hennessy [NH84] three testing preorders are defined, and for each of them a sound and ground-complete axiomatization over BCCS is provided. In fact the axiomatizations apply to all of CCS, enriched with a special constant Ω, and the semantics of processes involves, besides A_τ-labeled transitions, a *convergence* predicate. However, the completeness proofs remain valid when restricting attention to the sublanguage BCCS, and there the convergence predicate plays no role (for all processes are convergent). The *combined may- and must-testing preorder* is axiomatized by the laws A1-4 together with the axioms

$$\begin{array}{lrcl}
\text{N1} & \alpha x + \alpha y & \approx & \alpha(\tau x + \tau y) \\
\text{N2} & x + \tau y & \preccurlyeq & \tau(x + y) \\
\text{N3} & \alpha x + \tau(\alpha y + z) & \approx & \tau(\alpha x + \alpha y + z) \\
\text{N4} & \tau x & \preccurlyeq & x
\end{array}$$

[3] In fact, most preorders in [vG93b] are actually pairs of preorders, as for every semantics a *may* and a *must* preorder are proposed. Inspired by [NH84], there are two differences between the may and the must preorders. One is a different treatment of *divergence* – this has no effect when restricting attention to BCCS processes. The other is that the preorders are oriented in opposite directions. This entire paper, as well as [AFI07, dFGP08b], has been written from the perspective of the may preorders. When dealing with must preorders $\sqsubseteq$ we have that if $p \sqsubseteq q$ then $\mathcal{I}(p) \supseteq \mathcal{I}(q)$. Moreover, none of these preorders contains $\precsim_{\text{RS}}$ – at best their inverses have this property. Consequently, for preorders oriented in the must direction, the algorithm is to be applied in the reverse direction, where an inequational axiom $t \preccurlyeq u$ gives rise to equational axioms like $t \approx t + u$.

where α ranges over A_τ. The *must preorder* has the additional axiom

$$\text{E1} \qquad \tau x + \tau y \preccurlyeq x$$

and the *may preorder* has the additional axiom

$$\text{F1} \qquad x \preccurlyeq \tau x + \tau y \ .$$

Note that TAU2 follows from N2 and N4. We will now apply the algorithm to obtain sound and ground-complete axiomatizations of the three associated testing equivalences.

Beforehand, we mention a trivial simplification in applying the algorithm: if the inequational axiomatization features an equation $t \approx u$, formally speaking this is an abbreviation for the two axioms $t \preccurlyeq u$ and $u \preccurlyeq t$. Thus, step (1) of the algorithm generates the equations $t + u \approx u$ and $u + t \approx t$. Together, these are equivalent to the original equation $t \approx u$. Moreover, in the presence of $t \approx u$ the two axioms generated by step (2) of the algorithm are redundant. Thus, we can simplify the algorithm by leaving equations untouched.

The may preorder. The may preorder of [NH84] coincides with weak trace inclusion, which is coarser than the ready simulation preorder. As remarked in [NH84], it is not hard to see that the axiomatization above can be simplified to A1-4 together with

$$\begin{array}{rcl} \tau x & \approx & x \\ \alpha x + \alpha y & \approx & \alpha(x+y) \\ x & \preccurlyeq & x + y \end{array}$$

Applying Thm. 3.2.7 yields a sound and ground-complete axiomatization of may -testing equivalence, which coincides with weak trace equivalence. It consists of A1-4, $\text{RS}_{\equiv}$ and

$$\begin{array}{rcl} \tau x & \approx & x \\ \alpha x + \alpha y & \approx & \alpha(x+y) \\ x + x + y & \approx & x + y \\ \alpha(x+z) + \alpha(x+y+z) & \approx & \alpha(x+y+z) \end{array}$$

As $\text{RS}_{\equiv}$ is an instance of the last axiom above, that last axiom follows from the second, and the third from A3, this axiomatization can be simplified to A1-4 together with

$$\begin{array}{rcl} \tau x & \approx & x \\ \alpha x + \alpha y & \approx & \alpha(x+y) \end{array}$$

which is a standard axiomatization for weak trace equivalences.

The must preorder. On BCCS, the must preorder of [NH84] coincides with the failures preorder of CSP [BHR84]. Its inverse contains the ready simulation preorder and is weak initials preserving. Hence we can apply Thm. 3.2.9 to obtain a sound and ground-complete axiomatization of must-testing equivalence.

First we note that N4 is a simple consequence of E1 and thus can be omitted. Now Thm. 3.2.9 yields the axioms A1-4, $\mathrm{RS}_{\equiv}$ and

$$\begin{array}{lrcl}
\mathrm{N1} & \alpha x + \alpha y & \approx & \alpha(\tau x + \tau y) \\
\mathrm{N2}_1 & x + \tau y & \approx & x + \tau y + \tau(x + y) \\
\mathrm{N2}_2 & \alpha(x+\tau y+z) & \approx & \alpha(x+\tau y+z) + \alpha(\tau(x+y)+z) \\
\mathrm{N3} & \alpha x + \tau(\alpha y + z) & \approx & \tau(\alpha x + \alpha y + z) \\
\mathrm{E1}_1 & \tau x + \tau y & \approx & \tau x + \tau y + x \\
\mathrm{E1}_2 & \alpha(\tau x + \tau y + z) & \approx & \alpha(\tau x + \tau y + z) + \alpha(x + z)
\end{array}$$

This axiomatization can be simplified to A1-4 together with

$$\begin{array}{lrcl}
\mathrm{N1} & \alpha x + \alpha y & \approx & \alpha(\tau x + \tau y) \\
\mathrm{N2}^* & x + \tau y & \approx & \tau y + \tau(x + y) \\
\mathrm{N3} & \alpha x + \tau(\alpha y + z) & \approx & \tau(\alpha x + \alpha y + z)
\end{array}$$

Namely, $\mathrm{E1}_1$ implies Tau2 which allows us to reformulate $\mathrm{N2}_1$ as $\mathrm{N2}^*$. The latter axiom implies Tau2 (by taking $y = x$) and hence also $\mathrm{N2}_1$ and $\mathrm{E1}_1$. It remains to derive $\mathrm{N2}_2$, $\mathrm{E2}_2$ and $\mathrm{RS}_{\equiv}$. In all three cases, by N1 it suffices to derive the instance where $\alpha = \tau$. Substituting τy for y in $\mathrm{N2}^*$ and applying $\tau\tau y \approx \tau y$ (which follows from N1) and Tau2 gives $\tau(x + \tau y) \approx x + \tau y$. Now it is straightforward to derive $\mathrm{N2}_2^{\tau}$, $\mathrm{E2}_2^{\tau}$ and $\mathrm{RS}_{\equiv}^{\tau}$.

This axiomatization has been mentioned in [vG97], just like the axiomatization of weak trace equivalence mentioned above. However, we have not found an actual proof of its ground-completeness (or the ground-completeness of any other axiomatization of must-testing equivalence over BCCS) in the literature.

The combined may and must preorder. The combined may- and must-testing preorder is the intersection of the may- and the must-testing preorders. On BCCS, it is contained in weak trace equivalence, and hence contains neither the concrete ready simulation preorder, nor its inverse. Therefore, Thm. 3.2.6 – 3.2.9 are not applicable to it. However, its kernel does contain concrete ready simulation equivalence, and with help of Thm. 3.2.10 we can obtain a sound and ground-complete axiomatization of it. The algorithm yields the axioms A1-4, $\mathrm{RS}_{\equiv}$ and

$$\begin{array}{lrcl}
\mathrm{N1} & \alpha x + \alpha y & \approx & \alpha(\tau x + \tau y) \\
\mathrm{N2}_1 & x + \tau y & \approx & x + \tau y + \tau(x + y) \\
\mathrm{N2}_2 & \alpha(x+\tau y+z) & \approx & \alpha(x+\tau y+z) + \alpha(\tau(x+y)+z) \\
\mathrm{N3} & \alpha x + \tau(\alpha y + z) & \approx & \tau(\alpha x + \alpha y + z) \\
\mathrm{N4}_1 & \tau x & \approx & \tau x + x \\
\mathrm{N4}_2 & \alpha(\tau x + z) & \approx & \alpha(\tau x + z) + \alpha(x + z)
\end{array}$$

The soundness of these axioms follows from the fact that they are derivable both from the axioms for the may preorder and from the axioms for the must preorder.

The axiomatization above is easily seen to be equivalent to the axiomatization of must-testing equivalence. This is no surprise, as it is known that on BCCS the must preorder and the combined preorder have the same kernel.

3.2.4 A Generalization to Infinite Processes

The results in [AFI07, dFGP08b] were obtained for finite processes only: processes that can be expressed in BCCSP. Hereby we extend these results to infinite processes that can be expressed by adding constants to BCCS. This is an easy way of dealing with recursion – an alternative to introducing recursion as a syntactic construct and requiring congruence properties for it. An infinite process can be defined by introducing one or more constants C together with axioms like $C \approx abC$; in this example, C represents a process that performs an infinite alternating sequence of a and b actions.

In order to obtain completeness of the axiomatizations $\mathcal{A}(E)$, any extension of BCCS with constants will do. Lem. 3.2.3, Prop. 3.2.4 and Thm. 3.2.10 remain valid in this setting. The only place where structural induction is used is in the proof of Lem. 3.2.3, and there constants do not bother us, as they cannot occur on a path from the root of a context, seen as a parse tree, to the hole.

In order to obtain soundness, we furthermore assume that for any constant C in the language there is a closed term $\sum_{i\in I} \alpha_i p_i$ in our extension of BCCS with constants – so the index set I is finite – such that $C \leftrightarroweq \sum_{i\in I} \alpha_i p_i$. It then follows that any closed term is bisimulation equivalent to a closed term of the form $\sum_{i\in I} \alpha_i p_i$. With this assumption, all our results generalize to BCCS augmented with constants.

The proof of Lem. 3.2.5 goes through unaltered. The only proof that needs to be adapted is the one of Thm. 3.2.9.

Lemma 3.2.11 Let $\equiv$ be a congruence containing $\leftrightarroweq$ that satisfies TAU2. If $p \Rightarrow p' \xrightarrow{\alpha} p''$ then $p \equiv p + \alpha p''$.

Proof: By induction on the length of the path $p \Rightarrow p'$.

In the base case $p = p' \leftrightarroweq \sum_{i\in I} \alpha_i p_i$, and by definition of $\leftrightarroweq$ there must be an $i \in I$ with $\alpha_i = \alpha$ and $p_i \leftrightarroweq p''$. It follows that $p \leftrightarroweq p + \alpha p''$ and hence $p \equiv p + \alpha p''$.

Now assume $p \xrightarrow{\tau} p' \Rightarrow \xrightarrow{\alpha} p''$. By induction, $p' \equiv p' + \alpha p''$. TAU2 yields $p \equiv p + \alpha p''$. ■

Proof of Thm. 3.2.9: Suppose $p \sqsubseteq q$. We have to show that $p + q \equiv q$, where $\equiv$ is the kernel of $\sqsubseteq$. By the assumption above, $p \leftrightarroweq \sum_{i\in I} \alpha_i p_i$ for a finite index set I and closed terms $\alpha_i p_i$ in our extension of BCCS with constants. We have $\{\alpha_i \mid i \in I\} = I_\tau(p) \subseteq \mathcal{I}_\tau(p) \subseteq \mathcal{I}_\tau(q)$, so for every $i \in I$ there is a term q_i such that $q \Rightarrow \xrightarrow{\alpha_i} q_i$. Let$q' := q + \sum_{i\in I} \alpha_i q_i$. Applying Lem. 3.2.11 once for every $i \in I$ we obtain $q \equiv q'$. Now $I_\tau(p) \subseteq I_\tau(q)$, so Lem. 3.2.5 yields $p + q' \equiv q'$, which implies $p + q \equiv q$. ■

Applications (continued)

Adding divergence. In [NH84] a special constant Ω denoting *divergence* is considered, and the three ground-complete axiomatizations of the preorders

mentioned in Section 3.2.3 extend to the presence of divergence by means of the extra axiom

$$\Omega \qquad \Omega \preccurlyeq x \ .$$

Although Ω is defined in terms of a convergence predicate, in all three testing preorders it is equivalent to a process engaging in an infinite τ-loop only. We could therefore equivalently think of Ω as the process generated by adding the transition rule $\Omega \xrightarrow{\tau} \Omega$ to BCCS. This way we obtain $\Omega \leftrightarrows \tau\Omega$, thereby fulfilling the soundness requirement of Section 3.2.4. Note that $I_\tau(\Omega) = \mathcal{I}_\tau(\Omega) = \{\tau\}$.

Invoking Thm. 3.2.7 we obtain a ground-complete axiomatization for may-testing equivalence by adding the extra axioms

$$\begin{array}{rcl} \Omega + x & \approx & x \\ \alpha(\Omega + z) + \alpha(x + z) & \approx & \alpha(x + z) \end{array}$$

to the ones mentioned in Section 3.2.3. The second one is derivable from the first and $\alpha x + \alpha y \approx \alpha(x + y)$. Using A4, the first one is equivalent to $\Omega \approx \mathbf{0}$.

As the must preorder $\sqsubseteq$ satisfies $\Omega \sqsubseteq a0$ for some $a \neq \tau$, it is not weak initials preserving (in either direction) and we may not apply Thm. 3.2.9, as we did in Section 3.2.3. In order to obtain a sound and ground-complete axiomatization of must-testing equivalence, we therefore resort to Thm. 3.2.6. Applying the algorithm to the ground-complete axiomatization of the must preorder yields the extra axioms

$$\begin{array}{lrcl} \Omega_1 & \Omega & \approx & \Omega + x \\ \Omega_2 & \alpha(\Omega + z) & \approx & \alpha(\Omega + z) + \alpha(x + z) \end{array}$$

Thm. 3.2.6 requires us to explicitly check the soundness of $\mathrm{N2}_1$, $\mathrm{E1}_1$ and Ω_1. We may not use the soundness of $\mathrm{N2}_1$ and $\mathrm{E1}_1$ obtained in Section 3.2.3, as it could have been invalidated by the addition of Ω to the language. The soundness of $\mathrm{N2}_1$ follows from Lem. 3.2.5, applying the remark right after Thm. 3.2.6. The soundness of $\mathrm{E1}_1$ follows because it is derivable from TAU2, which is derivable from N2 and N4. The soundness of Ω_1 follows because it is derivable from Ω, TAU2 and E1, as shown in [NH84].

E2 and TAU2 yield $\Omega \approx \Omega + \tau\Omega \approx \tau\Omega$. With N1 the axiom Ω_2 follows from its instance where $\alpha = \tau$, which follows from E2 and $\tau\Omega = \Omega$. Hence a sound and ground-complete axiomatization of must-testing equivalence, also known as the failures equivalence of CSP, consists of N1, $\mathrm{N2}^*$, N3 and Ω_1.

Applying the algorithm to the combined may and must preorder again yields the extra axioms Ω_1 and Ω_2, and using Thm. 3.2.10 we cannot assume soundness without establishing this separately. In the presence of Ω the kernels of the must preorder and the combined preorder do not coincide, and this time Ω_1 turns out not to be sound. This is an example where we cannot apply the algorithm to obtain a sound and ground-compete axiomatization. We conjecture that such an axiomatization exists nonetheless, namely consisting of N1, $\mathrm{N2}^*$, N3 and

$$\begin{array}{lrcl} \mathrm{D1} & \Omega + \tau x & \approx & \Omega + x \\ \mathrm{D3} & \Omega + \alpha x & \approx & \Omega + \alpha(\Omega + x) \ . \end{array}$$

In [NH84] the axioms D1 and D3 have been derived from N1-4, thereby establishing their soundness.

3.2.5 Concluding Remark

In [dFGP08b], de Frutos Escrig, Gregorio Rodríguez and Palomino also present a simplification of the algorithm of [AFI07] for a large class of applications. The simplification consists in skipping step (2) in favor of a *constrained similarity axiom*

$$\mathrm{NS}_{\equiv} \qquad N(x,y) \implies \alpha x + \alpha(x+y) \approx \alpha(x+y) \text{ for } \alpha \in A_\tau \ .$$

Here $N(x,y)$ is a congruence relation on processes such that $N(p,q)$ is implied by $I_\tau(p) = I_\tau(q)$. The constrained similarity axiom is a conditional equation, but it can in several cases be recast in equational terms. In the special case where $N(p,q)$ holds iff $I_\tau(p) = I_\tau(q)$, $\mathrm{NS}_{\equiv}$ is equivalent to $\mathrm{RS}_{\equiv}$. They show that the simplified algorithm applies to preorders $\sqsubseteq$ satisfying

$$\mathrm{NS} \qquad N(x,y) \implies x \preceq x + y$$

and such that $p \sqsubseteq q$ implies $N(p,q)$. In case $N(p,q) \Leftrightarrow I_\tau(p) = I_\tau(q)$ we have that NS is equivalent to RS.

In applying this algorithm to τ-free preorders in the linear time – branching time spectrum I, they use three different constraints N, whose ranges of application match those of our Thm. 3.2.7, 3.2.8 and 3.2.9. Yet, we have not been able to apply the simplified algorithm to weak preorders, due to the fact that we would need an asymmetric precongruence N, whereas symmetry is used crucially in the proofs in [dFGP08b]. The same applies to the generalizations of the constrained similarity approach investigated in [dFGP08a].

3.3 From Concrete to Weak Semantics

In this section, we establish a link between the axiomatizability of concrete and weak semantics. Namely, we present a general method to derive a ground- (resp. ω-)complete axiomatization for the weak semantics from its concrete counterpart. The method requires that for the semantics the following two axioms are sound:

$$\begin{array}{lrcl} \mathrm{W1} & \alpha x + \alpha y & \approx & \alpha(\tau x + \tau y) \\ \mathrm{W2} & \tau(x+y) + \tau x & \approx & \tau x + y \end{array}$$

In particular, this is the case for weak impossible futures, weak failures, weak completed trace and weak trace semantics; see Def. 2.1.6.

Before presenting the algorithm, we define what is "the weak counterpart" of a concrete semantics. We first look at the case of preorders, then adapt it to the case of equivalences.

Definition 3.3.1 Given any *concrete* preorder $\precsim$ for which A1-4 are sound, the corresponding *weak* preorder $\sqsubseteq$ is a precongruence satisfying:

1. The inequational theory of BCCS modulo $\sqsubseteq$ is a *conservative extension* of the inequational theory of BCCSP modulo $\precsim$. Namely, for any τ-free terms p and q, $p \precsim q$ iff $p \sqsubseteq q$;
2. W1-2 are sound modulo $\sqsubseteq$;
3. If $p \sqsubseteq q$ for some p and q with $p \xrightarrow{\tau}$ and $q \not\xrightarrow{\tau}$, then $\tau x \preccurlyeq x$ is sound modulo $\sqsubseteq$;
4. If $p \sqsubseteq q$ for some p and q with $p \not\xrightarrow{\tau}$ and $q \xrightarrow{\tau}$, then $x \preccurlyeq \tau x$ is sound modulo $\sqsubseteq$.

Remark 3.3.2
- Requirements (3) and (4) concern root conditions, which are usually indispensable to make sure that the preorder under consideration is a precongruence. A typical root condition says: $t \sqsubseteq u$ only if $t \xrightarrow{\tau}$ implies that $u \xrightarrow{\tau}$ (note that this is case for weak preorders, cf. Def. 2.1.6).
- Given any concrete preorder $\precsim$, we have defined a family of corresponding weak preorders. However, it will become clear later that, due to the constraint imposed by (2), the family of weak preorders gives rise to a unique one, except for the treatment of τ's at the root, which is determined by (3) and (4).

The case of an equivalence can be defined accordingly, as follows.

Definition 3.3.3 Given any concrete equivalence $\simeq$, the corresponding weak equivalence $\equiv$ is a congruence satisfying:

1. The equational theory of BCCS modulo $\equiv$ is a conservative extension of the equational theory of BCCSP modulo $\simeq$. Namely, for any τ-free p and q, $p \simeq q$ iff $p \equiv q$;
2. W1-2 are sound modulo $\equiv$;
3. If $p \equiv q$ for some p and q with $p \xrightarrow{\tau}$ and $q \not\xrightarrow{\tau}$, then $\tau x \approx x$ is sound modulo $\equiv$.

We now embark on presenting the algorithm. Again we first deal with the case of preorders. From any axiomatization E_A for a concrete preorder $\precsim$ for which A1-4 are sound, the following algorithm generates an axiomatization $\mathcal{A}(E_A)$ for the corresponding weak preorder $\sqsubseteq$, which contains the following (in)equations:

- For each $t \preccurlyeq u$ in E_A, the same inequation $t \preccurlyeq u$ with the action names in t and u ranging over $A \cup \{\tau\}$ (the resulting set of inequations is denoted by $E_{A\cup\{\tau\}}$);
- W1 and W2; and

- The following option:
  - If $p \sqsubseteq q$ for some p and q with $p \xrightarrow{\tau}$ and $q \not\xrightarrow{\tau}$, then $\tau x \preccurlyeq x$ is included;
  - If $p \sqsubseteq q$ for some p and q with $p \not\xrightarrow{\tau}$ and $q \xrightarrow{\tau}$, then $x \preccurlyeq \tau x$ is included.

Remark 3.3.4 [4] One might argue that the third bullet in the algorithm is difficult to verify since a naïve procedure might have to check infinitely many process terms, and thus it lacks the algorithmic aspect. However, if the axiomatization E for $\sqsubseteq$ is finite and ground-complete which is the usual case of applying the algorithm, this shortcoming can be remedied. As a matter of fact, for the former condition, one can verify whether there is some inequation $t \preccurlyeq u$ in E such that

- either $\tau \in \mathcal{I}(t) \setminus \mathcal{I}(u)$;
- or $var(t) \supsetneq var(u)$.

It is not difficult to see that a positive answer can be obtained iff for some p and q, $p \sqsubseteq q$, $p \xrightarrow{\tau}$ and $q \not\xrightarrow{\tau}$. And the procedure terminates since E is finite. The latter condition can be tackled in a similar way.

The algorithm for equivalences can be adapted accordingly. That is, $\mathcal{A}(E_A)$ contains the following equations for the weak equivalence $\equiv$, given the axiomatization E_A for the concrete equivalence $\simeq$:

- For each $t \approx u$ in E_A, the same equation $t \approx u$ with the action names in t and u ranging over $A \cup \{\tau\}$ (the resulting set of inequations is denoted by $E_{A\cup\{\tau\}}$);
- W1 and W2; and
- If $p \equiv q$ for some p and q with $p \xrightarrow{\tau}$ and $q \not\xrightarrow{\tau}$, then $\tau x \approx x$ is included.

Below we establish the correctness of the algorithm for the preorder (the equivalence case follows the same lines), namely,

Theorem 3.3.5 Let $\precsim$ be a concrete preorder for which A1-4 are sound and $\sqsubseteq$ a corresponding weak preorder. Let E be an axiomatization such that E_A is sound and ground-complete for BCCSP(A) modulo $\precsim$ and $E_{A\cup\{\kappa\}}$ is ground-complete for BCCSP($A\cup\{\kappa\}$) modulo $\precsim$ for some $\kappa \notin A$. Suppose that $\mathcal{A}(E)$ is sound for BCCS(A) modulo $\sqsubseteq$. Then $\mathcal{A}(E)$ is ground-complete axiomatization for BCCS(A) modulo $\sqsubseteq$. Moreover, if E_A is ω-complete for BCCSP(A) and $E_{A\cup\{\kappa\}}$ is ω-complete for BCCSP($A \cup \{\kappa\}$) for some $\kappa \notin A$, then $\mathcal{A}(E)$ is ω-complete for BCCS(A).

[4] This is pointed by Luca Aceto when he reviewed a manuscript of the dissertation.

We first present some lemmas. For a start, the following inequations can be derived from A1-4+W1-2:

$$\begin{array}{lrcl} \text{D1} & \tau(\tau x + y) & \approx & \tau x + y \\ \text{D2} & a(\sum_{i\in I} \tau x_i + y) & \approx & a(\sum_{i\in I} x_i + y) + \sum_{i\in I} a x_i \ . \end{array}$$

Lemma 3.3.6 D1-2 are derivable from A1-4+W1-2.

Proof: For D1,

$$\begin{array}{rcll} \tau(\tau x + y) & \approx & \tau(\tau(x+y) + \tau x) & \text{(W2)} \\ & \approx & \tau x + y \ . & \text{(W1, W2)} \end{array}$$

For D2, we apply induction on $|I|$. The base case $I = \emptyset$ is trivial. For $|I| \geq 1$, pick an $i_0 \in I$,

$$\begin{array}{rcll} a(\sum_{i\in I} \tau x_i + y) & = & a(\tau x_{i_0} + \sum_{i\in I\setminus\{i_0\}} \tau x_i + y) & \\ & \approx & a(\tau(x_{i_0} + \sum_{i\in I\setminus\{i_0\}} \tau x_i + y) + \tau x_{i_0}) & \text{(W2)} \\ & \approx & a(x_{i_0} + \sum_{i\in I\setminus\{i_0\}} \tau x_i + y) + a x_{i_0} & \text{(W1)} \\ & \approx & a(x_{i_0} + \sum_{i\in I\setminus\{i_0\}} x_i + y) + \sum_{i\in I\setminus\{i_0\}} a x_i + a x_{i_0} & \text{(induction)} \\ & = & a(\sum_{i\in I} x_i + y) + \sum_{i\in I} a x_i \ . & \end{array}$$

The proof is now complete. ■

Lemma 3.3.7 The following properties hold:

1. Given any BCCS term t such that $t \stackrel{\tau}{\not\longrightarrow}$, A1-4+W1-2 $\vdash t \approx t'$ for some τ-free term t'; and
2. Given any BCCS term t such that $t \stackrel{\tau}{\longrightarrow}$, A1-4+W1-2 $\vdash t \approx \sum_{i\in I} \tau t_i$ for some index set $I \neq \emptyset$ where for each $i \in I$, t_i is τ-free.

Proof:

1. We apply induction on $|t|$. Since $t \stackrel{\tau}{\not\longrightarrow}$, $t = \sum_{i\in I} a_i t_i + X$. For each $i \in I$,

$$t_i \approx \sum_{j\in J_i} \tau t'_j + \sum_{k\in K_i} b_k t'_k + Y_i \ .$$

Because of D1, we can guarantee that for each $j \in J_i$, $t'_j \stackrel{\tau}{\not\longrightarrow}$. By D2,

$$a_i t_i \approx a_i(\sum_{j\in J_i} \tau t'_j + \sum_{k\in K_i} b_k t'_k + Y_i) \approx \sum_{j\in J_i} a_i t'_j + a_i(\sum_{j\in J_i} t'_j + \sum_{k\in K_i} b_k t'_k + Y_i) \ .$$

Since $|t'_j| < |t|$ and $t'_j \stackrel{\tau}{\not\longrightarrow}$, by induction $\vdash t_j \approx t''_j$ such that t''_j is τ-free. Moreover, since $|\sum_{j\in J_i} t'_j + \sum_{k\in K_i} b_k t'_k + Y_i| < |t|$, by induction there exists some τ-free term t''' with $\vdash \sum_{j\in J_i} t'_j + \sum_{k\in K_i} b_k t'_k + Y_i \approx t'''$. It follows that

$$a_i t_i \approx \sum_{j\in J_i} a_i t''_j + a_i t''' \ .$$

Hence (1) is obtained.

2. Since $t \stackrel{\tau}{\longrightarrow}$, we have

$$t \approx \sum_{i\in I} \tau t_i + \sum_{j\in J} a_j t_j + X \ ,$$

where $I \neq \emptyset$. Because of D1 we can guarantee that for each $i \in I$, $t_i \stackrel{\tau}{\not\longrightarrow}$. Choosing arbitrary $i_0 \in I$, by W2 we have

$$t \approx \sum_{i\in I} \tau t_i + \sum_{j\in J} a_j t_j + X \approx \sum_{i\in I} \tau t_i + \sum_{j\in J} \tau(t_{i_0} + a_j t_j + X) \ .$$

Hence (2) follows from (1).

The proof is now complete. ■

Given any BCCS term t such that $t = \sum_{i\in I} \tau t_i$ and for each $i \in I$, t_i is τ-free. We define $\mathbb{I}(t)$ as $\sum_{i\in I} \kappa t_i$ where κ is a fresh action.

Lemma 3.3.8 Given two BCCS terms $t = \sum_{i\in I} \tau t_i$ and $u = \sum_{j\in J} \tau u_j$ where t_i, u_j are all τ-free for $i \in I$ and $j \in J$. Let $\precsim$ be a concrete semantics for which A1-4 are sound and $\sqsubseteq$ the corresponding weak semantics satisfying W1 and W2. Then $t \sqsubseteq u$ iff $\mathbb{I}(t) \precsim \mathbb{I}(u)$.

Proof:

$\Rightarrow$) Suppose that $t \sqsubseteq u$. Since $\sqsubseteq$ is a precongruence, $\kappa t \sqsubseteq \kappa u$. That is,

$$\kappa(\sum_{i\in I} \tau t_i) \sqsubseteq \kappa(\sum_{j\in J} \tau u_j) \ .$$

It follows from the soundness of W1 that

$$\sum_{i\in I} \kappa t_i \sqsubseteq \sum_{j\in J} \kappa u_j \ .$$

In other words, $\mathbb{I}(t) \sqsubseteq \mathbb{I}(u)$. Since $\mathbb{I}(t)$ and $\mathbb{I}(u)$ are both τ-free, $\mathbb{I}(t) \precsim \mathbb{I}(u)$.

$\Leftarrow$) If $\mathbb{I}(t) \precsim \mathbb{I}(u)$, then $\mathbb{I}(t) \sqsubseteq \mathbb{I}(u)$. Renaming κ into τ transforms $\mathbb{I}(t)$ and $\mathbb{I}(u)$ into t and u respectively, and so $\mathbb{I}(t) \sqsubseteq \mathbb{I}(u)$ implies $t \sqsubseteq u$.

■

Proof of Thm. 3.3.5: Let E be a sound and ground-complete axiomatization of $\precsim$. Suppose $t \sqsubseteq u$, where we assume that either t and u are closed terms, or E is ω-complete. It suffices to show $\mathcal{A}(E) \vdash t \preccurlyeq u$. We distinguish the following cases:

1. $t \xrightarrow{\tau}$ and $u \xrightarrow{\tau}$. Since $t \sqsubseteq u$, by Lem 3.3.7(2), $\mathcal{A}(E) \vdash t \approx \sum_{i\in I} \tau t_i$ and $\mathcal{A}(E) \vdash u \approx \sum_{j\in J} \tau u_j$ such that for each $i \in I$ and $j \in J$, t_i and u_j are τ-free. By Lem. 3.3.8, $\mathbb{I}(\sum_{i\in I} \tau t_i) \precsim \mathbb{I}(\sum_{j\in J} \tau u_j)$, i.e., $\sum_{i\in I} \kappa t_i \precsim \sum_{j\in J} \kappa u_j$ Since either t and u are closed terms or E is ω-complete, and $\sum_{i\in I} \kappa t_i$ and $\sum_{j\in J} \kappa u_j$ are terms over BCCSP$(A \cup \{\kappa\})$, it follows from the ground- (ω-)completeness of $E_{A\cup\{\kappa\}}$ that

$$E_{A\cup\{\kappa\}} \vdash \mathbb{I}(\sum_{i\in I} \tau t_i) \preccurlyeq \mathbb{I}(\sum_{j\in J} \tau u_j) \ .$$

   By a simple renaming of κ into τ, one can easily show that $E_{A\cup\{\tau\}} \vdash t \preccurlyeq u$. Hence $\mathcal{A}(E) \vdash t \preccurlyeq u$.

2. $t \not\xrightarrow{\tau}$ and $u \not\xrightarrow{\tau}$. Since $t \sqsubseteq u$, by Lem 3.3.7(1), $\mathcal{A}(E) \vdash t \approx t'$ and $\mathcal{A}(E) \vdash u \approx u'$ such that t' and u' are τ-free. Clearly $t' \precsim u'$ and since either t and u are closed terms or E is ω-complete, it follows from the ground-completeness of E that $E \vdash t' \preccurlyeq u'$. Hence $\mathcal{A}(E) \vdash t \preccurlyeq u$.

3. $t \xrightarrow{\tau}$ and $u \not\xrightarrow{\tau}$. Then the axiom $\tau x \preccurlyeq x$ must be included. Since $t \sqsubseteq u$, $\tau t \sqsubseteq \tau u$. The conclusion follows from $\mathcal{A}(E) \vdash t \approx \tau t \preccurlyeq \tau u \preccurlyeq u$. Note that the first step follows from D1 and the second one from Case (1).

4. $t \not\xrightarrow{\tau}$ and $u \xrightarrow{\tau}$. Then the axiom $x \preccurlyeq \tau x$ must be included. Since $t \sqsubseteq u$, $\tau t \sqsubseteq \tau u$. The conclusion follows from $\mathcal{A}(E) \vdash t \preccurlyeq \tau t \preccurlyeq \tau u \approx u$. Note that the second step follows from Case (1) and the last one from D1.

The proof is now complete. ■

3.3.1 Applications

In this section, we apply the above algorithm to produce complete axiomatizations for weak *failures*, weak *completed trace* and weak *trace* semantics from their concrete counterparts respectively.

Failures Semantics

The definitions for concrete failures preorder $\precsim_{\mathrm{F}}$ and weak failures preorder $\preceq_{\mathrm{WF}}$ are given in Def. 2.1.4 and Def. 2.1.6 respectively. One can verify that $\preceq_{\mathrm{WF}}$ is the corresponding weak preorder of $\precsim_{\mathrm{F}}$ in a straightforward way.

It is known (see e.g. Section 4.3) that A1-4 together with the following axiom

$$\text{F1} \qquad a(x+y) \preccurlyeq ax + a(y+z)$$

constitute a ground-complete axiomatization for BCCSP(A) modulo concrete failures preorder $\precsim_{\mathrm{F}}$. Moreover, if $|A| = \infty$, it is even ω-complete. In contrast, if $|A| < \infty$, it is *not* ω-complete anymore. To get a finite basis for the inequational theory of BCCSP(A) modulo $\precsim_{\mathrm{F}}$ in case $1 < |A| < \infty$, we need to add the following axiom:

$$\text{F2}_A \qquad \sum_{a\in A} ax_a \preccurlyeq \sum_{a\in A} ax_a + y \ .$$

Applying the algorithm described above, we can obtain a ground-complete axiomatization which consists of A1-4 together with

$$\begin{array}{lrcl}
\text{WF} & \alpha(x+y) & \preccurlyeq & \alpha x + \alpha(y+z) \\
\text{W1} & \alpha x + \alpha y & \approx & \alpha(\tau x + \tau y) \\
\text{W2} & \tau(x+y) + \tau x & \approx & \tau x + y \\
\text{W3} & x & \preccurlyeq & \tau x \ .
\end{array}$$

It is ground-complete in general, and even ω-complete if $|A| = \infty$ w.r.t. BCCS(A) modulo weak failures preorder $\preceq_{\mathrm{WF}}$. It turns out that we can simplify it a little, namely, the following axioms together with A1-4 suffice.

$$\begin{array}{lrcl}
\text{W1} & \alpha x + \alpha y & \approx & \alpha(\tau x + \tau y) \\
\text{W2}' & \tau(x+y) & \preccurlyeq & \tau x + y \\
\text{W3}' & x & \preccurlyeq & \tau x + y
\end{array}$$

As a matter of fact, by W3′, $x+y \preccurlyeq \tau(x+y)+y+z \preccurlyeq \tau x+y+z \preccurlyeq \tau x+\tau(y+z)$. It follows that $\alpha(x+y) \preccurlyeq \alpha x + \alpha(y+z)$.

In the case of a finite alphabet, we have to add one extra axiom

$$\sum_{a\in A} ax_a + \tau x_\tau \preccurlyeq \sum_{a\in A} ax_a + \tau x_\tau + y$$

which is obtained directly from the algorithm. And again we can simplify it. That is, F2_A suffices. In summary, we have

Theorem 3.3.9 For BCCS(A) modulo weak failures preorder $\preceq_{\mathrm{WF}}$, the axiomatization A1-4+W1+W2′+W3′ is ground-complete. Moreover, if $|A| = \infty$, it is also ω-complete. If $1 < |A| < \infty$, A1-4+W1+W2′+W3′+F2_A is ω-complete.

Completed Trace Semantics

The definitions for concrete completed trace preorder $\precsim_{\mathrm{CT}}$ and weak completed trace preorder $\preceq_{\mathrm{WCT}}$ are given in Def. 2.1.2 and Def. 2.1.6 respectively. It is known that A1-4 together with the following axioms

$$\begin{array}{lrcl}
\text{CT1} & ax & \preccurlyeq & ax + y \\
\text{CT2} & a(bw + cx + y + z) & \preccurlyeq & a(bw + y) + a(cx + z)
\end{array}$$

constitute a ground-complete axiomatization for complete traces preorder. Moreover, axiom F1 is needed to make the axiomatization ω-complete. By applying the algorithm, we obtain a ground-complete axiomatization which consists of A1-4, W1-3 together with

$$\begin{array}{lrcl} \text{WCT1} & \alpha x & \preccurlyeq & \alpha x + y \\ \text{WCT2} & \alpha(\beta w + \gamma x + y + z) & \preccurlyeq & \alpha(\beta w + y) + \alpha(\gamma x + z) \ . \end{array}$$

It turns out that W1+W2′+W3′+CT1 together A1-4 suffice. To see this, clearly W3 is an instance of W3′, and WCT2 can be derived as:

$$\alpha(\beta w + \gamma x + y + z) \preccurlyeq \alpha(\tau(\beta w + y) + \tau(\gamma x + z)) \approx \alpha(\beta w + y) + \alpha(\gamma x + z) \ .$$

(The first step follows from W3 and the second one follows from W1.) Moreover, the instance $\tau x \preccurlyeq \tau x + y$ of WCT1 can be derived as $\tau x \preccurlyeq \tau(\tau x + y) \preccurlyeq \tau x + y$. (The first step follows from W3′ and the second one follows from D1.) We now can conclude that A1-4+W1+W2′+W3′+CT1 is ground-complete for BCCS modulo the weak completed traces preorder. Interesting, it is *already* ω-complete since the axiom WF can be derived, as shown before. In summary,

Theorem 3.3.10 For BCCS(A) modulo weak completed trace preorder $\preceq_{\mathrm{WCT}}$, the axiomatization A1-4+W1+W2′+W3′+CT1 is ω-complete.

Trace Semantics

The definitions for concrete trace preorder $\precsim_{\mathrm{T}}$ and weak trace preorder $\preceq_{\mathrm{WT}}$ are given in Def. 2.1.2 and Def. 2.1.6 respectively. It is known that A1-4 together with the following axioms

$$\begin{array}{lrcl} \text{T1} & a(x + y) & \preccurlyeq & ax + ay \\ \text{T2} & x & \preccurlyeq & x + y \end{array}$$

constitute a ground-complete axiomatization for traces preorder. Moreover, if $|A| > 1$, it is also ω-complete. When $|A| = 1$, the following extra axiom is needed to make the axiomatization ω-complete:

$$\text{T3} \quad x \preccurlyeq ax \ .$$

By applying the algorithm, we can obtain a ground-complete axiomatization which consists of A1-4, W1-2 together with

$$\begin{array}{lrcl} \text{WT1} & \alpha(x + y) & \preccurlyeq & \alpha x + \alpha y \\ \text{T2} & x & \preccurlyeq & x + y \\ \text{WE} & x & \approx & \tau x \ . \end{array}$$

Clearly, due to WE, W1-2 become redundant and WT1 can be replaced simply by T1. In summary, we have

Theorem 3.3.11 For BCCS(A) modulo weak trace preorder $\preceq_{\mathrm{WT}}$, the axiomatization A1-4+T1-2+WE is ground-complete. Moreover, if $1 < |A| \leq \infty$, it is also ω-complete. If $|A| = 1$, A1-4+T1-2+WE+T3 is ω-complete.

3.4 Inverted Substitutions

Groote [Gro90] introduced the technique of *inverted substitutions* to prove that an *equational* axiomatization is ω-complete, which works as follows. Recall that $\mathrm{T}(\Sigma)$ and $\mathbb{T}(\Sigma)$ denote the sets of closed and open terms, respectively, over some signature Σ. Consider an axiomatization E. For each equation $t \approx u$ of which all closed instances can be derived from E, one must define a closed substitution ρ and a mapping $R : \mathrm{T}(\Sigma) \to \mathbb{T}(\Sigma)$ such that:

(1) $E \vdash R(\rho(t)) \approx t$ and $E \vdash R(\rho(u)) \approx u$;

(2) For each function symbol f (with arity n) in the signature, $E \cup \{p_i \approx q_i,\ R(p_i) \approx R(q_i) \mid i = 1, \ldots, n\} \vdash R(f(p_1, \ldots, p_n)) \approx R(f(q_1, \ldots, q_n))$ for all closed terms $p_1, \ldots, p_n, q_1, \ldots, q_n$; and

(3) $E \vdash R(\sigma(v)) \approx R(\sigma(w))$ for each $v \approx w \in E$ and closed substitution σ.

Then, as proved in [Gro90], E is ω-complete.

Here we adapt his technique to make it suitable for *inequational* axiomatizations.

Theorem 3.4.1 Consider an inequational axiomatization E over Σ. Suppose that for each inequation $t \preccurlyeq u$ of which all closed instances can be derived from E, there are a closed substitution ρ and a mapping $R : \mathrm{T}(\Sigma) \to \mathbb{T}(\Sigma)$ such that:

(1) $E \vdash t \preccurlyeq R(\rho(t))$ and $E \vdash R(\rho(u)) \preccurlyeq u$;

(2) $E \vdash R(\sigma(v)) \preccurlyeq R(\sigma(w))$ for each $v \preccurlyeq w \in E$ and closed substitution σ; and

(3) For each function symbol f (with arity n) in the signature, and all closed terms $p_1, \ldots, p_n, q_1, \ldots, q_n$:

$$E \ \cup\ \{p_i \preccurlyeq q_i,\ R(p_i) \preccurlyeq R(q_i) \mid i = 1, \ldots, n\} \ \vdash \\ R(f(p_1, \ldots, p_n)) \preccurlyeq R(f(q_1, \ldots, q_n)) \ .$$

Then E is ω-complete.

Proof: Let t, u be terms such that for each closed substitution σ,

$$\sigma(t) \ \preccurlyeq\ \sigma(u) \ .$$

By assumption, there are a closed substitution ρ and a mapping $R : \mathrm{T}(\Sigma) \to \mathbb{T}(\Sigma)$ such that properties (1)-(3) above are satisfied. We have to prove that $E \vdash t \preccurlyeq u$. This is an immediate corollary of the following claim:

Claim. For all closed terms p, q:

$$E \ \vdash\ p \ \preccurlyeq\ q \quad \Longrightarrow \quad E \ \vdash\ R(p) \ \preccurlyeq\ R(q) \ .$$

Namely, by assumption, $E \vdash \rho(t) \preccurlyeq \rho(u)$, and then the claim above implies that $E \vdash R(\rho(t)) \preccurlyeq R(\rho(u))$. So by property (1), $E \vdash t \preccurlyeq u$.

Proof of the claim: By induction on the proof of $E \vdash p \preccurlyeq q$. We have to check the four kinds of inference rules of inequational logic (cf. 2.1.3).

- $p = q$. Then $R(p) = R(q)$.
- $p \preccurlyeq q$ is an instance of some $v \preccurlyeq w \in E$ and a closed substitution σ. By property (2), $E \vdash R(p) \preccurlyeq R(q)$.
- $E \vdash p \preccurlyeq q$ has been proved by $E \vdash p \preccurlyeq r$ and $E \vdash r \preccurlyeq q$, for some r. By induction, $E \vdash R(p) \preccurlyeq R(r)$ and $E \vdash R(r) \preccurlyeq R(q)$. So $E \vdash R(p) \preccurlyeq R(q)$.
- $p = f(p_1, \ldots, p_n)$ and $q = f(q_1, \ldots, q_n)$, and $E \vdash p \preccurlyeq q$ has been proved by $E \vdash p_i \preccurlyeq q_i$ for $i = 1, \ldots, n$. By induction, $E \vdash R(p_i) \preccurlyeq R(q_i)$ for $i = 1, \ldots, n$. So by property (3), $E \vdash R(f(p_1, \ldots, p_n)) \preccurlyeq R(f(q_1, \ldots, q_n))$.

■

The whole proof is now complete. ■

3.5 Related and Future Work

Meta-theories for axiomatizability are an interesting topic. However, it is largely unexplored in the literature. Notable exception, which shares the same spirit of the results presented in this chapter, is a recent result due to Aceto, Fokkink, Ingolfsdottir and Mousavi [AFIM08] where they present a general technique for obtaining new results pertaining to the non-finite axiomatizability of behavioral semantics over process algebras from old ones. The proposed technique is based on a variation on the classic idea of reduction mappings which are basically translations between languages that preserve sound (in)equations and (in)equational proofs over the source language, and reflect families of (in)equations responsible for the non-finite axiomatizability of the target language. This technique is applied to obtain a number of new non-finite axiomatizability theorems in process algebra via reduction to Moller's celebrated non-finite axiomatizability result for CCS, for instance, discrete-time CCS modulo timed bisimilarity, temporal CCS modulo timed bisimilarity.

For the future work, on the one hand we expect more meta theorems; On the other hand, we believe that these theorems can be cast into more general models such as those given by Plotkin-style SOS rules.

Chapter 4

On Finite Alphabets and Infinite Bases

4.1 Introduction

To give further insight into the identifications made by the respective behavioral equivalences in the linear time – branching time spectrum I, van Glabbeek [vG90, vG01] studied them in the setting of the process algebra BCCSP. In particular, he associated with every behavioral equivalence a sound equational axiomatization. Most of the axiomatizations were also shown to be ground-complete. In this chapter, we shall consider the existence of finite bases for these semantics. Given a finite ground-complete axiomatization, to prove that it is a finite basis, it suffices to establish that it is ω-complete (see Section 2.1.3). Groote [Gro90] proposed a general "inverted substitution" technique to prove that an axiomatization is ω-complete (see also Section 3.4). He applied his technique to establish ω-completeness of several of van Glabbeek's ground-complete axiomatizations. In practice, Groote's technique only works in case of an infinite alphabet of actions.[1] On the other hand, in case of a singleton alphabet, most of the semantics in the linear time – branching time spectrum I collapse to either trace or completed trace semantics, in which case the equational theory of BCCSP is known to have a finite basis. However, in case of a finite alphabet with at least two actions, for most semantics in the linear time – branching time spectrum I it remained unknown whether the equational theory of BCCSP has a finite basis. In this chapter, we settle all remaining open questions.

We first give a brief summary of what was known up to now, and which open questions remained. Moller [Mol89] proved that the sound and ground-complete axiomatization for BCCSP modulo bisimulation equivalence is ω-complete, independent of the cardinality of the alphabet A. Groote [Gro90] presented ω-completeness proofs for completed trace equivalence (again independent of the cardinality of A), for trace equivalence (if $|A| > 1$), and for ready and failure equivalence (if $|A| = \infty$). Van Glabbeek [vG01, page 78] noted (without proof) that Groote's technique of *inverted substitutions* can also be used to

[1] In case of an infinite alphabet, occurrences of action names in axioms are interpreted as variables, as otherwise most of the axiomatizations mentioned in this introduction would be infinite.

prove that the ground-complete axiomatizations for BCCSP modulo simulation, ready simulation and failure trace equivalence are ω-complete if $|A| = \infty$. The same observation can be made regarding possible worlds semantics. Blom, Fokkink and Nain [BFN03] proved that BCCSP modulo ready trace equivalence does not have a finite sound and ground-complete axiomatization if $|A| = \infty$. Aceto, Fokkink, van Glabbeek and Ingolfsdottir [AFvGI04] proved such a negative result for 2-nested simulation and possible futures equivalence, for any A. If $|A| = 1$, then all semantics from completed traces up to ready simulation coincide with completed trace semantics, while simulation coincides with trace semantics. And there exists a finite basis for the equational theories of BCCSP modulo completed trace and trace equivalence if $|A| = 1$.

This chapter settles all the remaining questions: we prove that there is a finite basis for the equational theory of BCCSP modulo failure semantics, in case $1 < |A| < \infty$; for all the other cases standing open up to now (in Tab. 4.1 on page 88 with grey shadow), we prove that such a finite basis lacks.

Recall that the semantics considered in this chapter have a natural formulation as a *preorder* relation $\precsim$, where $p \precsim q$ if p is in some way simulated by q, or if the decorated traces of p are included in those of q. The corresponding *equivalence* relation $\simeq$ is defined as: $p \simeq q$ iff both $p \precsim q$ and $q \precsim p$. Recently, Aceto, Fokkink and Ingolfsdottir [AFI07] gave an algorithm that, given a sound and ground-complete axiomatization for BCCSP modulo a preorder no finer than ready simulation, produces a sound and ground-complete axiomatization for BCCSP modulo the corresponding equivalence. Moreover, if the original axiomatization for the preorder is ω-complete, then so is the resulting axiomatization for the equivalence (see also Section 3.2 for more details). So for the positive result regarding failure semantics, the stronger result is obtained by considering failure preorder. On the other hand, the negative results become more general if they are proved for the equivalence relations.

Structure of the chapter. Section 4.2 presents some basic facts. Section 4.3 contains a positive result for failure preorder. The remainder of the chapter presents negative results: Section 4.4 for failure trace equivalence, Section 4.5 for any equivalence from possible worlds up to ready pairs, Section 4.6 for simulation equivalence, Section 4.7 for completed simulation equivalence, and Section 4.8 for ready simulation equivalence. We conclude in Section 4.9 with an overview of the positive and negative results achieved in this chapter.

4.2 Basic Facts

Lemma 4.2.1 1. If $t \precsim_{\mathrm{T}} u$, then $\mathit{depth}(t) \leq \mathit{depth}(u)$.

2. If $t \precsim_{\mathrm{T}} u$, then $\mathit{act}_k(t) \subseteq \mathit{act}_k(u)$ for all $k \geq 0$. Moreover, if $t \precsim_{\mathrm{F}} u$, then also $\mathit{act}_0(u) \subseteq \mathit{act}_0(t)$, so $\mathcal{I}(t) = \mathcal{I}(u)$.

3. Suppose $|A| > 1$. If $t \precsim_{\mathrm{T}} u$, then, for all variables x, $t \xrightarrow{a_1 \cdots a_k} x + t'$ for some term t' implies $u \xrightarrow{a_1 \cdots a_k} x + u'$ for some term u'. Hence $\mathit{var}_k(t) \subseteq$

$var_k(u)$ for all $k \geq 0$.

Proof:

1. If $depth(t) = k$, then there exists a sequence of actions $a_1 \cdots a_k$ and a term t' such that $t \xrightarrow{a_1 \cdots a_k} t'$. Let ρ be the closed substitution defined by $\rho(x) = \mathbf{0}$ for all $x \in V$. Then $a_1 \cdots a_k$ is a trace of $\rho(t)$ and hence, since $t \precsim_{\mathrm{T}} u$, of $\rho(u)$. From the definition of ρ it is then clear that there exists a term u' such that $u \xrightarrow{a_1 \cdots a_k} u'$. It follows that $depth(t) = k \leq depth(u)$.

2. First suppose $t \precsim_{\mathrm{T}} u$ and let $a \in act_k(t)$ for some $k \geq 0$. Then there exists a sequence of actions $a_1 \cdots a_k$ and a term t' such that $t \xrightarrow{a_1 \cdots a_k} t'$ and $a \in \mathcal{I}(t')$. Now, let ρ be the closed substitution defined by $\rho(x) = \mathbf{0}$ for all $x \in V$. Then $a_1 \cdots a_k a$ is a trace of $\rho(t)$ and hence, since $t \precsim_{\mathrm{T}} u$, of $\rho(u)$. From the definition of ρ it is then clear that there exists a term u' such that $u \xrightarrow{a_1 \cdots a_k} u'$ with $a \in \mathcal{I}(u')$, so $a \in act_k(u)$.

   Next, suppose $t \precsim_{\mathrm{F}} u$ and let ρ be the closed substitution defined by $\rho(x) = \mathbf{0}$ for all $x \in V$. Then $(\lambda, A \setminus \mathcal{I}(t))$ (with λ denoting the empty sequence) is a failure pair of $\rho(t)$, and hence of $\rho(u)$, so $\mathcal{I}(u) \cap (A \setminus \mathcal{I}(t)) = \emptyset$; it follows that $act_0(u) \subseteq act_0(t)$. Since $t \precsim_{\mathrm{F}} u$ implies $t \precsim_{\mathrm{T}} u$, and hence $act_0(t) \subseteq act_0(u)$, it immediately follows that $\mathcal{I}(t) = act_0(t) = act_0(u) = \mathcal{I}(u)$.

3. Let x be a variable and suppose $t \xrightarrow{a_1 \cdots a_k} x + t'$ for some term t'. Let $m \geq depth(u)$, let a and b be two distinct elements of A, and let ρ be the closed substitution defined by $\rho(x) = a^m b\mathbf{0}$ and $\rho(y) = \mathbf{0}$ for any variable $y \neq x$. Then $\rho(t) \xrightarrow{a_1 \cdots a_{k+m} b} \mathbf{0}$ (with $a_{k+1} \cdots a_{k+m} = a^m$). Since $\rho(t) \precsim_{\mathrm{T}} \rho(u)$, $a_1 \cdots a_{k+m} b$ is also a trace of $\rho(u)$. Since $m \geq depth(u)$, clearly $u \xrightarrow{a_1 \cdots a_i} z + u'$ for some $i < m$, where $\rho(z) \xrightarrow{a_{i+1} \cdots a_{k+m} b} p$. By the definition of ρ, $z = x$ and $i = k$, so $u \xrightarrow{a_1 \cdots a_k} x + u'$ for some term u'. Clearly it follows that $x \in var_k(t)$ implies $x \in var_k(u)$ for all variables x, so $var_k(t) \subseteq var_k(u)$.

■

Note that Lem. 4.2.1(3) fails in case $|A| = 1$, for if $A = \{a\}$, then $x \precsim_{\mathrm{T}} ax$. In the remainder of this chapter we will assume that $|A| > 1$.

4.3 Failures

In this section we consider the failures preorder $\precsim_{\mathrm{F}}$. Van Glabbeek [vG01] presented a sound and ground-complete axiomatization of the failures preorder consisting of the axioms A1-4, the axiom

$$\text{F1} \qquad a(x + y) \preccurlyeq ax + a(y + z) \ ,$$

and the axiom $ax \preccurlyeq ax + az$. Note that the latter axiom is actually superfluous, since it can be obtained from F1 by substituting $\mathbf{0}$ for y and applying A3.

Below, we provide a basis for the equational theory of BCCSP modulo $\precsim_{\mathrm{F}}$. We shall prove that A1-4+F1 is a basis if $|A| = \infty$ (see Corollary 4.3.6). To get a basis for the case that $1 < |A| < \infty$, it will be necessary to add the following axiom:

$$\mathrm{F2}_A \qquad \sum_{a\in A} ax_a \preccurlyeq \sum_{a\in A} ax_a + y \ ,$$

where $\{x_a \mid a \in A\}$ is a family of distinct variables and $y \notin \{x_a \mid a \in A\}$. To see that $\mathrm{F2}_A$ is sound modulo $\precsim_{\mathrm{F}}$, let ρ be an arbitrary closed substitution and consider a failure pair $(a_1 \cdots a_k, B)$ of $\rho(\sum_{a\in A} ax_a)$. If $k > 0$, then clearly $(a_2 \cdots a_k, B)$ is a failure pair of $\rho(x_{a_1})$, so $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(\sum_{a\in A} ax_a + y)$. On the other hand, if $k = 0$, then note that $\mathcal{I}(\rho(\sum_{a\in A} ax_a)) = A$, so $B = \emptyset$, and hence $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(\sum_{a\in A} ax_a + y)$. To see that $\mathrm{F2}_{A'}$ is not sound modulo $\precsim_{\mathrm{F}}$ if A' is a proper subset of A, let ρ be the closed substitution such that $\rho(y) = b\mathbf{0}$ for some $b \notin A'$; then $\mathcal{I}(\rho(\sum_{a\in A'} ax_a)) = A' \neq A' \cup \{b\} = \mathcal{I}(\rho(\sum_{a\in A'} ax_a + y))$. Since A1-4+F1 are sound modulo $\precsim_{\mathrm{F}}$ independent of the alphabet, it also follows that $\mathrm{F2}_A$ cannot be derived from A1-4+F1.

Axiom $\mathrm{F2}_A$ expresses that additional variable summands may be added to a term t whenever $\mathcal{I}(t) = A$. The following lemma confirms that the proviso $\mathcal{I}(t) = A$ is necessary.

Lemma 4.3.1 If $t \precsim_{\mathrm{F}} u$, then $var_0(t) \subseteq var_0(u)$, and if moreover $\mathcal{I}(t) \neq A$, then $var_0(t) = var_0(u)$.

Proof: Suppose $t \precsim_{\mathrm{F}} u$. That $var_0(t) \subseteq var_0(u)$ follows immediately from Lem. 4.2.1(3). To prove that $\mathcal{I}(t) \neq A$ implies $var_0(t) = var_0(u)$, suppose, towards a contradiction, that $a \notin \mathcal{I}(t)$ for some $a \in A$ and that $x \in var_0(u) \setminus var_0(t)$ for some $x \in V$. Define a closed substitution ρ by $\rho(x) = a\mathbf{0}$ and $\rho(y) = \mathbf{0}$ for $y \neq x$. Since $a \notin \mathcal{I}(t)$ and $x \notin var_0(t)$, $(\lambda, \{a\})$ (with λ the empty trace) is a failure pair of $\rho(t)$. Since $x \in var_0(u)$, $(\lambda, \{a\})$ is not a failure pair of $\rho(u + Y)$. This contradicts the assumption that $t \precsim_{\mathrm{F}} u$. We conclude that $\mathcal{I}(t) \neq A$ implies $var_0(t) = var_0(u)$. ■

According to Lem. 4.3.1, all the variable summands of t are also summands of u. Moreover, if u has a variable summand x that t does not have, then $\mathcal{I}(t) = A$, so we can derive $t \preccurlyeq t + x$ with an application of $\mathrm{F2}_A$. We proceed to establish, for all $a \in A$, a relation between a prefix summand at' of t and the sum of all similar prefix summands au' of u. To conveniently express this relation, we first introduce some further notation.

Let t be a term, and let $A' \subseteq A$; we define the *restriction* $t{\restriction_{A'}}$ of t to A' by

$$t{\restriction_{A'}} = \sum\{at' \mid a \in A' \text{ and } at' \Subset t\} \ .$$

Recall that $t \precsim_{\mathrm{F}} u$ if, for all closed substitutions ρ, the failure pairs of $\rho(t)$ are included in $\rho(u)$. The preorder $\precsim_{\mathrm{F}}$ fails to have certain structural properties with respect to the operations of BCCSP; in particular, we cannot in general

conclude from $at \precsim_{\mathrm{F}} au$ that $t \precsim_{\mathrm{F}} u$. It will therefore be technically convenient to also have notation for a preorder that is slightly coarser than $\precsim_{\mathrm{F}}$. We define the *length* of a failure pair $(a_1 \cdots a_k, B)$ as the length of the sequence $a_1 \cdots a_k$, and we write $t \precsim^1_{\mathrm{F}} u$ if, for all closed substitutions ρ, the failure pairs of length ≥ 1 of $\rho(t)$ are included in those of $\rho(u)$. We leave it to the readers to verify that $t \precsim_{\mathrm{F}} u$ iff $t \precsim^1_{\mathrm{F}} u$ and $\mathcal{I}(u) \subseteq \mathcal{I}(t)$, and that $at \precsim^1_{\mathrm{F}} au$ implies $t \precsim^1_{\mathrm{F}} u$.

Lemma 4.3.2 If $t \precsim^1_{\mathrm{F}} u$, then, for every summand at' of t, $at' \precsim_{\mathrm{F}} u{\restriction}_{\{a\}}$.

Proof: Suppose $t \precsim^1_{\mathrm{F}} u$. Let $at' \Subset t$ and let ρ be a closed substitution.

We first prove that the failure pairs of length ≥ 1 of $\rho(at')$ are included in those of $\rho(u{\restriction}_{\{a\}})$, and then we will conclude that also the failure pairs of length 0 of $\rho(at')$ are included in those of $\rho(u{\restriction}_{\{a\}})$.

Consider a failure pair $(a_1 \cdots a_k, B)$ of $\rho(at')$ with $k \geq 1$. Then $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(t)$. By our assumption that $t \precsim^1_{\mathrm{F}} u$, it follows that $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(u)$. From this we cannot directly conclude that u has a summand au' such that $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(au')$, as u may have a variable summand x such that $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(x)$. To ascertain that u nevertheless also has the desired summand au', we define a modification ρ' of ρ such that for all $\ell < k$ and for all terms v, $\rho(v)$ and $\rho'(v)$ have the same failure pairs $(b_1 \cdots b_\ell, B)$, while $(a_1 \cdots a_k, B)$ is not a failure pair of $\rho'(x)$ for all $x \in V$.

We obtain $\rho'(x)$ from $\rho(x)$ by replacing subterms ap at depth $k-1$ by $\mathbf{0}$ if $a \notin B$ and by $aa\mathbf{0}$ if $a \in B$. That is,

$$\rho'(x) = \mathit{chop}_{k-1}(\rho(x))$$

with chop_m for all $m \geq 0$ inductively defined by

$$\begin{array}{lcl}
\mathit{chop}_m(\mathbf{0}) & = & \mathbf{0} \\
\mathit{chop}_m(p+q) & = & \mathit{chop}_m(p) + \mathit{chop}_m(q) \\
\mathit{chop}_0(ap) & = & \begin{cases} \mathbf{0} & \text{if } a \notin B \\ aa\mathbf{0} & \text{if } a \in B \end{cases} \\
\mathit{chop}_{m+1}(ap) & = & a\,\mathit{chop}_m(p) \ .
\end{array}$$

We first prove two properties concerning the failure pairs of $\mathit{chop}_m(p)$, for $m \geq 0$ and closed terms p.

I. For all $\ell \leq m$, the closed terms p and $\mathit{chop}_m(p)$ have the same failure pairs $(b_1 \cdots b_\ell, B)$. To show this, we apply induction on m.

Base case: Since the summands of $\mathit{chop}_0(p)$ are $aa\mathbf{0}$ for all $a \in \mathcal{I}(p) \cap B$, $\mathcal{I}(p) \cap B = \emptyset$ iff $\mathcal{I}(\mathit{chop}_0(p)) \cap B = \emptyset$.

Inductive case: Let $\ell \leq m+1$; we distinguish cases according to whether $\ell = 0$ or $\ell > 0$. If $\ell = 0$, then, since $\mathcal{I}(p) = \mathcal{I}(\mathit{chop}_{m+1}(p))$, it follows that $\mathcal{I}(p) \cap B = \emptyset$ iff $\mathcal{I}(\mathit{chop}_{m+1}(p)) \cap B = \emptyset$, so $(b_1 \cdots b_\ell, B)$ is a failure pair of p iff it is a failure pair of $\mathit{chop}_{m+1}(p)$. If $\ell > 0$, then, since $p \xrightarrow{b_1} p'$

iff $chop_{m+1}(p) \xrightarrow{b_1} chop_m(p')$ and, by the induction hypothesis, p' and $chop_m(p')$ have the same failure pairs $(b_2 \cdots b_\ell, B)$, $(b_1 \cdots b_\ell, B)$ is a failure pair of p iff it is a failure pair of $chop_{m+1}(p)$.

II. $chop_m(p)$ does not have any failure pair $(b_1 \cdots b_{m+1}, B)$. To show this, we apply induction on m.

Base case: Since the summands of $chop_0(p)$ are $aa\mathbf{0}$ with $a \in \mathcal{I}(p) \cap B$, $chop_0(p)$ does not have a failure pair (b_1, B).

Inductive case: By induction, for closed terms q, $chop_m(q)$ does not have failure pairs $(b_2 \cdots b_{m+2}, B)$. Since the transitions of $chop_{m+1}(p)$ are $chop_{m+1}(p) \xrightarrow{b_1} chop_m(p')$ for $p \xrightarrow{b_1} p'$, it follows that $chop_{m+1}(p)$ does not have failure pairs $(b_1 \cdots b_{m+2}, B)$.

We proceed to prove that ρ' has the desired properties mentioned above.

A. For all $\ell < k$ and for all terms v, $\rho(v)$ and $\rho'(v)$ have the same failure pairs $(b_1 \cdots b_\ell, B)$. To show this, we apply induction on ℓ.

Base case: From the definition of $chop_{k-1}$ it follows that $\mathcal{I}(\rho'(x)) \cap B = \mathcal{I}(\rho(x)) \cap B$ for all $x \in V$. Hence, $\mathcal{I}(\rho(v)) \cap B = \emptyset$ iff $\mathcal{I}(\rho'(v)) \cap B = \emptyset$.

Inductive case: Let $\ell + 1 < k$. We prove for each summand of v that applying ρ or ρ' gives rise to the same failure pairs $(b_1 \cdots b_{\ell+1}, B)$. By property (I), $\rho(x)$ and $\rho'(x) = chop_{k-1}(\rho(x))$ have the same failure pairs $(b_1 \cdots b_{\ell+1}, B)$. Furthermore, by induction, for each summand $b_1 v'$ of v, $\rho(v')$ and $\rho'(v')$ have the same failure pairs $(b_2 \cdots b_{\ell+1}, B)$; so $\rho(b_1 v')$ and $\rho'(b_1 v')$ have the same failure pairs $(b_1 \cdots b_{\ell+1}, B)$.

B. $(a_1 \cdots a_k, B)$ is not a failure pair of $\rho'(x)$ for all $x \in V$. This is immediate from property (II).

Now, since $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(at')$, $(a_2 \cdots a_k, B)$ is a failure pair of $\rho(t')$, and hence, by property (A), of $\rho'(t')$. It follows that $(a_1 \cdots a_k, B)$ is a failure pair of $\rho'(t)$, and hence, by our assumption that $t \precsim^1_{\mathrm{F}} u$, of $\rho'(u)$. Since, according to property (B), u does not have a variable summand x such that $(a_1 \cdots a_k, B)$ is a failure pair of $\rho'(x)$, and since $a_1 = a$, u must have a summand au' such that $(a_1 \cdots a_k, B)$ is a failure pair of $\rho'(au')$ of u. Then, again by property (A), $(a_1 \cdots a_k, B)$ is a failure pair of $\rho(au')$ and hence of $\rho(u\restriction_{\{a\}})$.

We have now established that the failure pairs of length ≥ 1 of $\rho(at')$ are included in those of $\rho(u\restriction_{\{a\}})$. In particular, since $\rho(at')$ has the failure pair $(a, \emptyset)$, so does $\rho(u\restriction_{\{a\}})$, and hence $\mathcal{I}(\rho(at')) = \{a\} = \mathcal{I}(\rho(u\restriction_{\{a\}}))$. As an immediate consequence we get that also the failure pairs of length 0 of $\rho(at')$ are included in those of $\rho(u\restriction_{\{a\}})$. We conclude that $at' \precsim_{\mathrm{F}} u\restriction_{\{a\}}$. ■

We now proceed to establish that if the inequation $at' \preccurlyeq \sum_{j \in J} au_j$ is sound modulo the failures preorder, then it can be derived from A1-4+F1+F2$_A$. For the case that $\mathcal{I}(t) \neq A$, we need the following lemma.

Lemma 4.3.3 If $at \precsim_{\mathrm{F}} \sum_{j\in J} au_j$ and $\mathcal{I}(t) \neq A$, then there exists $j \in J$ such that $\mathcal{I}(u_j) \subseteq \mathcal{I}(t)$ and $var_0(u_j) \subseteq var_0(t)$.

Proof: Suppose $at \precsim_{\mathrm{F}} \sum_{j\in J} u_j$ and $\mathcal{I}(t) \neq A$. Let $b \in A \setminus \mathcal{I}(t)$ and define the closed substitution ρ by $\rho(x) = \mathbf{0}$ if $x \in var_0(t)$ and $\rho(x) = b\mathbf{0}$ if $x \notin var_0(t)$. Then $(a, A \setminus \mathcal{I}(t))$ is a failure pair of $\rho(at)$, so there exists $j \in J$ such that $(a, A \setminus \mathcal{I}(t))$ is a failure pair of au_j. From $(A \setminus \mathcal{I}(t)) \cap \mathcal{I}(\rho(u_j)) = \emptyset$ it follows that $\mathcal{I}(u_j) \subseteq \mathcal{I}(t)$ and $var_0(u_j) \subseteq var_0(t)$. ■

The following lemma constitutes the crucial step in our completeness proof.

Lemma 4.3.4 If $at \precsim_{\mathrm{F}} \sum_{j\in J} au_j$, then A1-4+F1+F2$_A \vdash at \preccurlyeq \sum_{j\in J} au_j$.

Proof: We apply induction on the depth of t. Note that from $at \precsim_{\mathrm{F}} \sum_{j\in J} au_j$ it follows that $t \precsim^1_{\mathrm{F}} \sum_{j\in J} u_j$. Let $t\restriction_{\mathcal{I}(t)} = \sum_{i\in I} b_i t_i$. Then, for all $i \in I$, by Lem. 4.3.2 $b_i t_i \precsim_{\mathrm{F}} \sum_{j\in J} u_j\restriction_{\{b_i\}}$, and hence by the induction hypothesis A1-4+F1+F2$_A \vdash b_i t_i \preccurlyeq \sum_{j\in J} u_j\restriction_{\{b_i\}}$. It follows that

$$\text{A1-4+F1+F2}_A \vdash t\restriction_{\mathcal{I}(t)} = \sum_{i\in I} b_i t_i \preccurlyeq \sum_{i\in I}\sum_{j\in J} u_j\restriction_{\{b_i\}} = \sum_{j\in J} u_j\restriction_{\mathcal{I}(t)} \ . \tag{4.1}$$

We distinguish two cases.

Case 1: $\mathcal{I}(t) \neq A$.

According to Lem. 4.3.3 that there exists $j_0 \in J$ such that $\mathcal{I}(u_{j_0}) \subseteq \mathcal{I}(t)$ and $var_0(u_{j_0}) \subseteq var_0(t)$, and hence

$$u_{j_0}\restriction_{\mathcal{I}(t)} + var_0(t) = u_{j_0} + var_0(t) \ . \tag{4.2}$$

We get the following derivation:

$$\begin{aligned}
at &= a(t\restriction_{\mathcal{I}(t)} + var_0(t)) \\
&\preccurlyeq a(\sum_{j\in J} u_j\restriction_{\mathcal{I}(t)} + var_0(t)) && \text{(by (4.1))} \\
&= a(u_{j_0} + \sum_{j\in J} u_j\restriction_{\mathcal{I}(t)} + var_0(t)) && \text{(by (4.2))} \\
&\preccurlyeq au_{j_0} + a(\sum_{j\in J} u_j + var_0(t)) && \text{(by F1)} \\
&= au_{j_0} + a\sum_{j\in J} u_j && \text{(by Lem. 4.2.1(3))} \\
&\preccurlyeq au_{j_0} + \sum_{j\in J} au_j && \text{(by F1)} \\
&= \sum_{j\in J} au_j \ .
\end{aligned}$$

Case 2: $\mathcal{I}(t) = A$.

Then since $var_0(t) \subseteq \bigcup_{j\in J} var_0(u_j)$ by Lem. 4.2.1(3), with an application of $\mathrm{F2}_A$

$$t = t\restriction_{\mathcal{I}(t)} + var_0(t) \preccurlyeq t\restriction_{\mathcal{I}(t)} + \bigcup_{j\in J} var_0(u_j) \ . \tag{4.3}$$

We now get the following derivation:

$$\begin{aligned}
at &= a(t\restriction_{\mathcal{I}(t)} + var_0(t)) \\
&\preccurlyeq a(t\restriction_{\mathcal{I}(t)} + \bigcup_{j\in J} var_0(u_j)) && \text{(by (4.3))} \\
&\preccurlyeq a(\sum_{j\in J} u_j\restriction_{\mathcal{I}(t)} + \bigcup_{j\in J} var_0(u_j)) && \text{(by (4.1))} \\
&= a\sum_{j\in J} u_j && \text{(since } \mathcal{I}(t) = A\text{)} \\
&\preccurlyeq \sum_{j\in J} au_j && \text{(by F1)}
\end{aligned}$$

Concluding, we have proved that $\text{A1-4+F1+F2}_A \vdash at \preccurlyeq \sum_{j\in J} au_j$. ■

We are now in a position to establish that A1-4+F1+F2_A constitutes a complete axiomatization of the failures preorder.

Theorem 4.3.5 If $0 < |A| < \infty$, then A1-4+F1+F2_A is a complete axiomatization of BCCSP modulo failures preorder, i.e., for all terms t and u, if $t \precsim_{\mathrm{F}} u$, then $\text{A1-4+F1+F2}_A \vdash t \preccurlyeq u$.

Proof: Suppose $t \precsim_{\mathrm{F}} u$, and suppose $t = \sum_{i\in I} a_i t_i + var_0(t)$. Then, for all $i \in I$, by Lem. 4.3.2 $a_i t_i \precsim_{\mathrm{F}} u\restriction_{\{a_i\}}$, so by Lem. 4.3.4, $\text{A1-4+F1+F2}_A \vdash a_i t_i \preccurlyeq u\restriction_{\{a_i\}}$. Clearly, since $\mathcal{I}(t) = \mathcal{I}(u)$ by Lem. 4.2.1(2), it follows that

$$\text{A1-4+F1+F2}_A \vdash t\restriction_{\mathcal{I}(t)} \preccurlyeq u\restriction_{\mathcal{I}(u)} \ .$$

There are now two cases:

Case 1: $\mathcal{I}(t) \neq A$.

Then $var_0(t) = var_0(u)$ by Lem. 4.3.1, so clearly

$$\text{A1-4+F1+F2}_A \vdash t = t\restriction_{\mathcal{I}(t)} + var_0(t) \preccurlyeq u\restriction_{\mathcal{I}(u)} + var_0(u) = u \ .$$

Case 2: $\mathcal{I}(t) = A$.

Then $var_0(t) \subseteq var_0(u)$ by Lem. 4.3.1, so $t = t\restriction_{\mathcal{I}(t)} + var_0(t) \preccurlyeq t\restriction_{\mathcal{I}(t)} + var_0(u)$ by $\mathrm{F2}_A$, and hence

$$\text{A1-4+F1+F2}_A \vdash t = t\restriction_{\mathcal{I}(t)} + var_0(t) \preccurlyeq u\restriction_{\mathcal{I}(u)} + var_0(u) = u \ .$$

The proof is now complete. ■

Groote [Gro90] proved that in case $|A| = \infty$, BCCSP modulo failures *equivalence* has a finite basis. Here we can obtain the same result for failure *preorder*, by copying the proofs of Lem. 4.3.4 and Thm. 4.3.5, but omitting in both proofs "Case 2", which is only relevant for finite alphabets.

Corollary 4.3.6 If $|A| = \infty$, then A1-4+F1 is an ω-complete axiomatization for BCCSP modulo failures preorder.

4.4 Failure Traces

In this section we consider failure trace equivalence $\simeq_{\mathrm{FT}}$. (We mention in passing that for finitely branching processes, this is the same as *refusal semantics* [Phi87].) Blom, Fokkink and Nain [BFN03] gave a finite axiomatization that is sound and ground-complete for BCCSP modulo $\simeq_{\mathrm{FT}}$. It consists of axioms A1-4 together with

$$\begin{array}{lrcl} \mathrm{FT} & ax + ay & \approx & ax + ay + a(x + y) \\ \mathrm{RS} & a(bx + by + z) & \approx & a(bx + by + z) + a(bx + z) \ , \end{array}$$

where a, b range over A. Groote [Gro90] applied his technique of inverted substitutions to prove that this axiomatization is ω-complete in case A is infinite.

In this section we consider the case $1 < |A| < \infty$. We prove that then there does not exist a finite sound and ground-complete axiomatization for BCCSP modulo $\simeq_{\mathrm{FT}}$ that is ω-complete as well, and therefore failure trace equivalence is not finitely based over BCCSP. The cornerstone for this negative result is the following infinite family of equations e_n $(n \geq 1)$:

$$a^{n+1}x + a(a^n x + x) + a \sum_{b \in A \setminus \{a\}} a^n(b\mathbf{0} + x) \approx a(a^n x + x) + a \sum_{b \in A \setminus \{a\}} a^n(b\mathbf{0} + x) \ .$$

These equations are sound modulo $\simeq_{\mathrm{FT}}$. The idea is that, given a closed substitution ρ, either $\mathcal{I}(\rho(x)) \subseteq \{a\}$, in which case the failure traces of $\rho(a^{n+1}x)$ are included in those of $\rho(a(a^n x + x))$. Or $c \in \mathcal{I}(\rho(x))$ for some $c \neq a$, in which case the failure traces of $\rho(a^{n+1}x)$ are included in those of $\rho(a \sum_{b \in A \setminus \{a\}} a^n(b\mathbf{0} + x))$.

We shall use the proof-theoretic technique to show that $\simeq_{\mathrm{FT}}$ is not finitely based. The intuition behind our proof is that if the axioms in E have depth at most n, then the summand $a^{n+1}x$ at the left-hand side of e_n cannot be eliminated by means of a derivation from E. There is, however, one complication: the summand $a^{n+1}x$ may be "glued together" with other summands. For example, using the axioms FT and RS we can derive for $n \geq 1$:

$$a^{n+1}x + a \sum_{b \in A \setminus \{a\}} a^n(b\mathbf{0} + x) \approx a(a^n x + \sum_{b \in A \setminus \{a\}} a^n(b\mathbf{0} + x)) \ .$$

The right-hand side of the equation above does not have a summand $a^{n+1}x$, so the property of having a summand $a^{n+1}x$ is not preserved. Note that the right-hand side still does have a summand of the form av such that $a^n x \precsim_{\mathrm{FT}} v$ (take $v = (a^n x + \sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + x))$). We shall be able to show that if the equation $t \approx u$ is derivable from a collection of sound equations of terms with a depth $\leq n$, then it satisfies the following property P_n^{FT}:

> If $t, u \precsim_{\mathrm{FT}} a(a^n x+x)+a\sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0}+x)$, then t has a summand at' such that $a^n x \precsim_{\mathrm{FT}} t'$, then u has a summand au' such that $a^n x \precsim_{\mathrm{FT}} u'$.

In Lem. 4.4.1 we shall first establish that a substitution instance of a sound equation of terms with a depth $\leq n$ satisfies P_n^{FT}. Then, in Prop. 4.4.2, we prove that P_n^{FT} is preserved in derivations from a collection of sound equations of depth $\leq n$. Finally, we shall conclude that the family of equations e_n $(n \geq 1)$ obstructs a finite basis, because the left-hand side has the summand $a^{n+1}x$, while the right-hand side does *not* have a summand au' with $a^{n+1}x \precsim_{\mathrm{FT}} au'$.

Lemma 4.4.1 Suppose that $t \simeq_{\mathrm{FT}} u$, let $n \geq 1$ be a natural number greater than or equal to the depth of t and u, and suppose

$$\sigma(t), \sigma(u) \precsim_{\mathrm{FT}} a(a^n x + x) + a \sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + x) \ . \tag{4.4}$$

Then $\sigma(t)$ has a summand av such that $a^n x \precsim_{\mathrm{FT}} v$ iff $\sigma(u)$ has a summand aw such that $a^n x \precsim_{\mathrm{FT}} w$.

Proof: Clearly, by symmetry, it suffices to only consider the implication from left to right. So suppose that $\sigma(t)$ has a summand av such that $a^n x \precsim_{\mathrm{FT}} v$; then there are two cases:

Case 1: t has a variable summand z and $\sigma(z)$ has av as a summand.

> Since $t \simeq_{\mathrm{FT}} u$, by Lem. 4.2.1(3), u also has z as summand. Therefore, since $\sigma(z)$ has av as a summand, so does $\sigma(u)$.

Case 2: t has a summand at' such that $a^n x \precsim_{\mathrm{FT}} \sigma(t')$.

> First, we establish that
>
> $$\sigma(t') \xrightarrow{a^n} x \text{ and } var_m(\sigma(t')) = \emptyset \text{ for all } 0 \leq m < n. \tag{4.5}$$
>
> From the assumption (4.4) we conclude using Lem. 4.2.1 (2,3) that $\mathcal{I}(\sigma(t)) = \{a\}$, $var_0(\sigma(t'))$, $var_n(\sigma(t')) \subseteq \{x\}$ and $var_m(\sigma(t')) = \emptyset$ for all $0 < m < n$. It follows that $a\sigma(t') \precsim_{\mathrm{FT}} \sigma(t)$, and hence
>
> $$a\sigma(t') \precsim_{\mathrm{FT}} a(a^n x + x) + a \sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + x) \ . \tag{4.6}$$

Now, let ρ_1 be a closed substitution with $\rho_1(x) = \mathbf{0}$. Since $a^n\mathbf{0} \precsim_{\mathrm{FT}} \rho_1(\sigma(t'))$, we have $\rho_1(\sigma(t')) \xrightarrow{a^n} \mathbf{0}$. Since $var_n(\sigma(t')) \subseteq \{x\}$, it follows that either $\sigma(t') \xrightarrow{a^n} x$ or $\sigma(t') \xrightarrow{a^n} \mathbf{0}$.

Note that, to establish (4.5), it remains to prove $\sigma(t') \stackrel{a^n}{\nrightarrow} \mathbf{0}$ and $x \notin var_0(\sigma(t'))$. For this we consider $\sigma(t')$ under another closed substitution ρ_2 that satisfies $\rho_2(x) = c\mathbf{0}$ with c an action distinct from a. Then, according to (4.6), $a\rho_2(\sigma(t')) \precsim_{\mathrm{FT}} a(a^n c\mathbf{0} + c\mathbf{0}) + a\sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + c\mathbf{0})$, and since the closed term at the right-hand side does not exhibit the failure trace

$$\underbrace{\emptyset\, a \cdots \emptyset\, a}_{n+1 \text{ times}} A \ ,$$

we have $\rho_2(\sigma(t')) \stackrel{a^n}{\nrightarrow} \mathbf{0}$, so $\sigma(t') \stackrel{a^n}{\nrightarrow} \mathbf{0}$. Furthermore, since $a^n x \precsim_{\mathrm{FT}} \sigma(t')$, we have $a^n c\mathbf{0} \precsim_{\mathrm{FT}} \rho_2(\sigma(t'))$. So $c \notin \mathcal{I}(\rho_2(\sigma(t')))$, and hence $x \notin var_0(\sigma(t'))$. This completes the proof of (4.5).

We proceed to prove that u has a summand au' such that

$$\sigma(u') \xrightarrow{a^n} x \text{ and } var_0(\sigma(u')) = \emptyset \ . \tag{4.7}$$

From (4.5) and the assumption that $depth(\sigma(t)) \leq n$ it follows that there exist $\ell < n$, a variable y and a term t'' such that $t' \xrightarrow{a^\ell} y + t''$ and $\sigma(y) \xrightarrow{a^{n-\ell}} x$.

Define Z as the set of variables z such that $\sigma(z)$ has x as a summand, i.e.,

$$Z = \{z \in V \mid x \in var_0(\sigma(z))\} \ .$$

Since y has an occurrence in t' at depth $\ell < n$, it follows from (4.5) that $x \notin var_0(\sigma(y))$, so $y \notin Z$. Therefore, we can define a closed substitution ρ_3 by

$$\rho_3(z) = \begin{cases} a^{n+1}\mathbf{0} & \text{if } z = y \\ c\mathbf{0} & \text{if } z \in Z \\ \mathbf{0} & \text{otherwise} \ , \end{cases}$$

where c is again an action distinct from a.

Since $t \xrightarrow{a} t' \xrightarrow{a^\ell} y + t''$, $\rho_3(y) \xrightarrow{a^{n+1}} \mathbf{0}$, $c \notin \mathcal{I}(t')$, and $x \notin var_0(\sigma(t'))$ implies $var_0(t') \cap Z = \emptyset$, $\rho_3(t)$ admits the failure trace

$$\emptyset\, a\, \{c\}\, \underbrace{a\, \emptyset \cdots a\, \emptyset}_{\ell+n \text{ times}}\, a\, \{a\} \ ,$$

which by the assumption $t \simeq_{\mathrm{FT}} u$ is then also a failure trace of $\rho_3(u)$. Since $depth(u') < n$, and in view of the definition of ρ_3, this clearly means that u has a summand au' such that $c \notin \mathcal{I}(\rho_3(u'))$ and $u' \xrightarrow{a^\ell} y + u''$ for some term u''. Since $\sigma(y) \xrightarrow{a^{n-\ell}} x$, it follows that $\sigma(u') \xrightarrow{a^n} x$. Moreover, from

$c \notin \mathcal{I}(\rho_3(u'))$ it follows that $var_0(u') \cap Z = \emptyset$, and hence $x \notin var_0(\sigma(u'))$. So we have now established (4.7).

From the assumption (4.4) we can conclude, by Lem. 4.2.1 (2)(3), that $act_m(\sigma(u')) \subseteq \{a\}$ for all $0 \leq m < n$ and that $var_m(\sigma(u')) = \emptyset$ for all $0 < m < n$, and (4.7) adds that $\sigma(u') \xrightarrow{a^n} x$, and $var_0(\sigma(u')) = \emptyset$. These facts together easily imply $a^n x \precsim_{\mathrm{FT}} \sigma(u')$.

The proof is now complete. ■

We shall now prove that the property P_n^{FT} holds for every equation derivable from a collection of equations between terms of depth less than or equal to n. By the preceding lemma, it suffices to prove that the transitivity and congruence rules preserve P_n^{FT}.

Proposition 4.4.2 Let E be a finite axiomatization over BCCSP that is sound modulo $\simeq_{\mathrm{FT}}$, let $n \geq 1$ be a natural number greater than or equal to the depth of any term in E, and suppose $E \vdash t \approx u$ and

$$t, u \precsim_{\mathrm{FT}} a(a^n x + x) + a \sum_{b \in A \setminus \{a\}} a^n(b\mathbf{0} + x) \ .$$

Then t has a summand at' such that $a^n x \precsim_{\mathrm{FT}} t'$ iff u has a summand au' such that $a^n x \precsim_{\mathrm{FT}} u'$.

Proof: We prove the proposition by induction on the depth of a normalized derivation of the equation $t \approx u$ from E.

To establish the base case, note that if the derivation of $t \approx u$ consists of an application of the reflexivity rule, then the proposition is immediate, and if there exist terms v and w and a substitution σ such that $\sigma(v) = t$ and $\sigma(w) = u$ and $(v \approx w) \in E$ or $(w \approx v) \in E$, then $v \simeq_{\mathrm{FT}} w$ by the soundness of E, so the proposition follows by Lem. 4.4.1.

For the inductive case we distinguish cases according to the last rule applied.

Case 1: the last rule applied is the transitivity rule.

Then there exist a term v and normalized derivations of $t \approx v$ and $v \approx u$. By the soundness of E, $v \simeq_{\mathrm{FT}} u \precsim_{\mathrm{FT}} a(a^n x + x) + a \sum_{b \in A \setminus \{a\}} a^n(b\mathbf{0} + x)$. Hence, by the induction hypothesis, v has a summand av' such that $a^n x \precsim_{\mathrm{FT}} v'$, and therefore, again by induction, u has a summand au' such that $a^n x \precsim_{\mathrm{FT}} u'$.

Case 2: the last rule applied is the congruence rule for a.

Then $t = at'$ and $u = au'$ for some terms t' and u', and there exists a normal derivation of $t' \approx u'$. Since t consists of a single summand at', $a^n x \precsim_{\mathrm{FT}} t'$. So by the soundness of E, $a^n x \precsim_{\mathrm{FT}} u'$.

CASE 3: the last rule applied is the congruence rule for $+$. Then $t = t_1 + t_2$ and $u = u_1 + u_2$ for some terms t_1, t_2, u_1 and u_2, and there exist normal derivations of $t_1 \approx u_1$ and $t_2 \approx u_2$. Since t has a summand at' with $a^n x \precsim_{\mathrm{FT}} t'$, so does either t_1 or t_2. Assume, without loss of generality, that t_1 has a summand at' such that $a^n x \precsim_{\mathrm{FT}} t'$. Since $\mathcal{I}(u) = \{a\}$, clearly $u_1 \precsim_{\mathrm{FT}} u \precsim_{\mathrm{FT}} a(a^n x + x) + a\sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + x)$. So by the induction hypothesis u_1, and hence u, has a summand au' with $a^n x \precsim_{\mathrm{FT}} u'$.

The proof is now complete. ■

Now we are in a position to prove the main theorem of this section.

Theorem 4.4.3 Let $1 < |A| < \infty$. Then the equational theory of BCCSP modulo $\simeq_{\mathrm{FT}}$ is not finitely based.

Proof: Let E be a finite axiomatization over BCCSP that is sound modulo $\simeq_{\mathrm{FT}}$. Let $n \geq 1$ be greater than or equal to the depth of any term in E.

Note that $a(a^n x + x) + a\sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + x)$ does not contain a summand au' such that $a^n x \precsim_{\mathrm{FT}} u'$. So according to Prop. 4.4.2, the equation

$$a^{n+1}x + a(a^n x + x) + a\sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + x) \approx a(a^n x + x) + a\sum_{b\in A\setminus\{a\}} a^n(b\mathbf{0} + x) \ ,$$

which is sound modulo $\simeq_{\mathrm{FT}}$, cannot be derived from E. It follows that every finite collection of equations that are sound modulo $\simeq_{\mathrm{FT}}$ is necessarily incomplete, and hence the equational theory of BCCSP modulo $\simeq_{\mathrm{FT}}$ is not finitely based. ■

4.5 From Ready Pairs to Possible Worlds

In this section we consider all congruences $\simeq$ that finer than or as fine as ready equivalence and coarser than or coarse as possible worlds equivalence (i.e., $\simeq_{\mathrm{PW}} \subseteq \simeq \subseteq \simeq_{\mathrm{R}}$). We prove that if $1 < |A| < \infty$, then no finite sound and ground-complete axiomatization for BCCSP modulo $\simeq$ is ω-complete.

In [vG90, vG01], van Glabbeek gave a finite axiomatization that is sound and ground-complete for BCCSP modulo $\simeq_{\mathrm{R}}$. It consists of axioms A1-4 together with

$$\mathrm{R} \qquad a(bx + z_1) + a(by + z_2) \approx a(bx + by + z_1) + a(by + z_2) \ ,$$

where a, b range over A. In case A is infinite, Groote [Gro90] proved with his technique of inverted substitutions that this axiomatization is ω-complete. So in that case, ready equivalence is finitely based over BCCSP.

Note that $\simeq_{\mathrm{PW}} \subseteq \simeq_{\mathrm{RT}} \subseteq \simeq_{\mathrm{R}}$. Blom, Fokkink and Nain [BFN03] proved that if $|A| = \infty$, then no finite axiomatization is sound and ground-complete for BCCSP modulo $\simeq_{\mathrm{RT}}$. They also proved that if $|A| < \infty$, then a finite sound and ground-complete axiomatization for BCCSP modulo $\simeq_{\mathrm{RT}}$ is obtained by extending axioms A1-4 with

$$\mathrm{RT} \qquad a(\sum_{i=1}^{|A|}(b_i x_i + b_i y_i) + z) \approx a(\sum_{i=1}^{|A|} b_i x_i + z) + a(\sum_{i=1}^{|A|} b_i y_i + z) \ ,$$

where $a, b_1, \ldots, b_{|A|}$ range over A.

In [vG90, vG01], van Glabbeek gave a finite axiomatization that is sound and ground-complete for BCCSP modulo $\simeq_{\mathrm{PW}}$. It consists of axioms A1-4 together with

$$\mathrm{PW} \qquad a(bx + by + z) \approx a(bx + z) + a(by + z) \ ,$$

where a, b range over A. If A is infinite, then Groote's technique of inverted substitutions can be applied in a straightforward fashion to prove that this axiomatization is ω-complete. So in that case, possible worlds equivalence is finitely based over BCCSP.

To prove the result mentioned above, originally we started out with the following infinite family of equations e_n for $n > |A|$:

$$\begin{aligned} & a(x_1 + \cdots + x_n) + \sum_{i=1}^{n} a(x_1 + \cdots + x_{i-1} + x_{i+1} + \cdots + x_n) \\ \approx \ & \sum_{i=1}^{n} a(x_1 + \cdots + x_{i-1} + x_{i+1} + \cdots + x_n) \ . \end{aligned}$$

These equations are sound modulo $\simeq_{\mathrm{PW}}$. Namely, it is not hard to see that for each closed substitution ρ, the possible worlds of the summand $\rho(a(x_1+\cdots+x_n))$ at the left-hand side of $\rho(e_n)$ are included in the possible worlds of the right-hand side of $\rho(e_n)$.

However, our expectation that the equations e_n for $n > |A|$ would obstruct a finite ω-complete axiomatization turned out to be false. Namely, e_n can be obtained by (1) applying to e_{n-1} a substitution ρ with $\rho(x_i) = x_i + x_n$ for $i = 1, \ldots, n-1$, and (2) adding the summand $a(x_1 + \cdots + x_{n-1})$ at the left- and right-hand side of the resulting equation. Hence, from $e_{|A|+1}$ (together with A1-3) we can derive the e_n for $n > |A|$.

Therefore we then moved to a more complicated family of equations (see Def. 4.5.8), similar in spirit to the equations e_n. However, while cancellation of the summand $a(x_1 + \cdots + x_{n-1})$ from e_n for $n > |A| + 1$ leads to an equation that is again sound modulo $\simeq_{\mathrm{PW}}$, such a cancellation is not possible for the new family of equations (see Lem. 4.5.10). We prove that they do obstruct a finite ω-complete axiomatization (see Thm. 4.5.14).

4.5.1 Cover Equations

We introduce a class of *cover equations* (cf. Section 2.1.5), and show that they are sound modulo $\simeq_{\mathrm{PW}}$. We prove that each equation that involves terms of depth ≤ 1 and that is sound modulo $\simeq_{\mathrm{R}}$ can be derived from the cover equations. Moreover, if such an equation contains no more than k summands at its left- and right-hand side, then it can be derived from cover equations containing no more than k summands at their left- and right-hand sides (see Prop. 4.5.7).

Definition 4.5.1 A term $\sum_{i\in I} aY_i$ is a *cover* of a term aX if:

1. $\forall Z \subseteq X$ with $|Z| \leq |A| - 1$, $\exists i{\in}I$ $(Z \subseteq Y_i \subseteq X)$; and
2. $\forall Z \subseteq X$ with $|Z| = |A|$, $\exists i{\in}I$ $(Z \subseteq Y_i)$.

This is denoted by $\sum_{i\in I} aY_i \trianglerighteq aX$. We say that $aX + \sum_{i\in I} aY_i \approx \sum_{i\in I} aY_i$ is a *cover equation*.

Example 4.5.2 $\sum_{i=1}^{n} a(x_1 + \cdots + x_{i-1} + x_{i+1} + \cdots + x_n) \trianglerighteq a(x_1 + \cdots + x_n)$ for $n > |A|$. Hence the equations that were given at the start of this section are cover equations.

If $|X| \leq |A| - 1$, then by Def. 4.5.1(1), $t \trianglerighteq aX$ implies that aX is a summand of t. So the only interesting cover equations are the ones where $|X| \geq |A|$ (cf. Def. 4.5.8).

We proceed to prove that the cover equations are sound modulo $\simeq_{\mathrm{PW}}$.

Lemma 4.5.3 If $\sum_{i\in I} aY_i \trianglerighteq aX$, then $aX + \sum_{i\in I} aY_i \simeq_{\mathrm{PW}} \sum_{i\in I} aY_i$.

Proof: Let ρ be an arbitrary closed substitution. It suffices to show that the possible worlds of $\rho(aX)$ are also possible worlds of $\rho(\sum_{i\in I} aY_i)$. Let ap be a possible world of $\rho(aX)$. Then p is a possible world of $\rho(X)$. By the definition of possible worlds equivalence, p has exactly $|\mathcal{I}(\rho(X))|$ summands, one summand bp_b for each $b \in \mathcal{I}(\rho(X))$; and for each $b \in \mathcal{I}(\rho(X))$ there is an $x_b \in X$ such that $\rho(x_b) \xrightarrow{b} q_b$ and p_b is a possible world of q_b. Let $Z = \{x_b \mid b \in \mathcal{I}(\rho(X))\}$. Then $\mathcal{I}(\rho(Z)) = \mathcal{I}(\rho(X))$. Clearly p is a possible world of $\rho(Z)$. Note that $|Z| \leq |\mathcal{I}(\rho(X))|$. We distinguish two cases.

Case 1: $|\mathcal{I}(\rho(X))| \leq |A| - 1$.

By Def. 4.5.1(1), $Z \subseteq Y_{i_0} \subseteq X$ for some $i_0 \in I$. Then clearly p is a possible world of $\rho(Y_{i_0})$. Thus ap is a possible world of $\rho(\sum_{i\in I} aY_i)$.

Case 2: $|\mathcal{I}(\rho(X))| = |A|$.

By Def. 4.5.1(2), $Z \subseteq Y_{i_0}$ for some $i_0 \in I$. Then $\mathcal{I}(\rho(Z)) \subseteq \mathcal{I}(\rho(Y_{i_0}))$, and hence, since $\mathcal{I}(\rho(Z)) = A$, it follows that $\mathcal{I}(\rho(Y_{i_0})) = \mathcal{I}(\rho(Z))$. From $Z \subseteq Y_{i_0}$ and $\mathcal{I}(\rho(Y_{i_0})) = \mathcal{I}(\rho(Z))$ we conclude that every possible world of Z is a possible word of Y_{i_0}. Since p is a possible world of $\rho(Z)$, it follows that p is a possible world of $\rho(Y_{i_0})$. Thus ap is a possible world of $\rho(\sum_{i\in I} aY_i)$.

The proof is now complete. ■

We proceed to prove that each sound equation $t \approx u$ modulo $\simeq_{\mathrm{R}}$ where t and u have depth 1 and contain no more than k summands, can be derived from the cover equations with $|I| \leq k$ (see Prop. 4.5.7). First we present some notations.

Definition 4.5.4 $C^k = \{aX + \sum_{i\in I} aY_i \approx \sum_{i\in I} aY_i \mid \sum_{i\in I} aY_i \trianglerighteq aX \wedge |I| \leq k\}$ for $k \geq 0$.

Definition 4.5.5 R_1 denotes the set of equations $t \approx u$ with $\mathit{depth}(t) = \mathit{depth}(u) \leq 1$ that are sound modulo $\simeq_{\mathrm{R}}$.

Let $S(t)$ denote the number of distinct summands (modulo A1-4) unequal to $\mathbf{0}$ of term t. For $k \geq 0$,

$$R_1^k = \{t \approx u \in R_1 \mid S(t) \leq k \wedge S(u) \leq k\} \ .$$

In the remainder of this section we assume that $A = \{a_1, \ldots, a_{|A|}\}$.

We present part of the proof of Prop. 4.5.7 as a separate lemma, as this lemma will be reused in the proof of Lem. 4.5.11.

Lemma 4.5.6 If $t \approx u \in R_1$, then t and u contain exactly the same summands aX with $|X| \leq |A| - 1$.

Proof: Let aX be a summand of t where $X = \{x_1, \ldots, x_k\}$ with $k \leq |A| - 1$. We define $\rho(x_i) = a_i\mathbf{0}$ for $i = 1, \ldots, k$ and $\rho(y) = a_{k+1}\mathbf{0}$ for $y \notin X$. Then $(a, \{a_1, \ldots, a_k\})$ is a ready pair of $\rho(t)$, so it must be a ready pair of $\rho(u)$. Since $\mathit{depth}(u) \leq 1$, this implies that aX is a summand of u.

By symmetry, each summand aX with $|X| \leq |A| - 1$ of u is also a summand of t. ■

Proposition 4.5.7 $C^k \vdash R_1^k$ for $k \geq 0$.

Proof: Let $t \approx u \in R_1^k$. Consider a summand aX of t with $|X| \geq |A|$. We prove that a subset of the summands of u form a cover of aX.

Case 1: $Z = \{z_1, \ldots, z_k\} \subseteq X$ with $k \leq |A| - 1$.

We define $\rho(z_i) = a_i\mathbf{0}$ for $i = 1, \ldots, k$, $\rho(x) = \mathbf{0}$ for $x \in X \setminus Z$, and $\rho(y) = a_{|A|}\mathbf{0}$ for $y \notin X$. The ready pair $(a, \{a_1, \ldots, a_k\})$ of $\rho(aX)$ must also be a ready pair of $\rho(u)$. Since $\mathit{depth}(u) \leq 1$, this implies that there is a summand aY of u with $Z \subseteq Y \subseteq X$.

Case 2: $Z = \{z_1, \ldots, z_{|A|}\} \subseteq X$.

We define $\rho(z_i) = a_i\mathbf{0}$ for $i = 1, \ldots, |A|$ and $\rho(y) = \mathbf{0}$ for $y \notin Z$. The ready pair (a, A) of $\rho(aX)$ must also be a ready pair of $\rho(u)$. Since $\mathit{depth}(u) \leq 1$, this implies that there is a summand aY of u with $Z \subseteq Y$.

Concluding, in view of Def. 4.5.1, $u = u_1 + u_2$ with $u_1 \trianglerighteq aX$. Since $S(u_1) \leq S(u) \leq k$, we have $aX + u_1 \approx u_1 \in C^k$. So $C^k \vdash aX + u \approx u$.

By Lem. 4.2.1(3) and 4.5.6, each summand $x \in V$ and aX with $|X| \leq |A|-1$ of t is a summand of u. Moreover, $C^k \vdash aX + u \approx u$ for each summand aX of t with $|X| \geq |A|$. Hence, $C^k \vdash t + u \approx u$.

By symmetry, also $C^k \vdash t + u \approx t$. So $C^k \vdash t \approx t + u \approx u$. ■

4.5.2 Cover Equations $a_1X_n + \Theta_n \approx \Theta_n$ for $n \geq |A|$

We now turn our attention to a special kind of cover equation $a_1X_n + \Theta_n \approx \Theta_n$ for $n \geq |A|$, where Θ_n contains $n+1$ summands (see Def. 4.5.8 and Lem. 4.5.9). If a term u is obtained by eliminating one or more summands from Θ_n, then $a_1X_n + u \not\trianglerighteq_{\mathrm{R}} u$ (see Lem. 4.5.10); moreover, if a summand of a term u is not a summand of $a_1X_n + \Theta_n$, then $\Theta_n \not\trianglerighteq_{\mathrm{R}} u$ (see Lem. 4.5.11). These two facts together imply that $a_1X_n + \Theta_n \approx \Theta_n$ cannot be derived from C^n (see Prop. 4.5.13). Prop. 4.5.7 and 4.5.13 form the cornerstones of the proof of Thm. 4.5.14, which contains the main result of this section.

Definition 4.5.8 Let $n \geq |A|$. Let $x_1, \ldots, x_n, \hat{x}_{|A|}, \ldots, \hat{x}_n$ be distinct variables. Let $X_{|A|-1}$ and X_n denote $\{x_1, \ldots, x_{|A|-1}\}$ and $\{x_1, \ldots, x_n\}$, respectively. We define that Θ_n denotes the term

$$a_1X_{|A|-1} + \sum_{i=1}^{|A|-1} a_1(X_n \setminus \{x_i\}) + \sum_{i=|A|}^{n} a_1(X_{|A|-1} \cup \{x_i, \hat{x}_i\}) \ .$$

Lemma 4.5.9 $\Theta_n \trianglerighteq a_1X_n$ for $n \geq |A|$.

Proof: Let $Z \subseteq X_n$ with $|Z| \leq |A| - 1$. We need to find a summand a_1Y of Θ_n with $Z \subseteq Y \subseteq X_n$. We distinguish two cases. On the one hand, if $Z \subseteq X_{|A|-1}$, then $Z \subseteq X_{|A|-1} \subseteq X_n$. On the other hand, if $Z \not\subseteq X_{|A|-1}$, then $Z \subseteq X_n \setminus \{x_i\} \subseteq X_n$ for some $1 \leq i \leq |A| - 1$.

Let $Z \subseteq X_n$ with $|Z| = |A|$. We need to find a summand a_1Y of Θ_n with $Z \subseteq Y$. Again there are two cases. On the one hand, if $X_{|A|-1} \subset Z$, then $Z \subseteq X_{|A|-1} \cup \{x_i, \hat{x}_i\}$ for some $|A| \leq i \leq n$. On the other hand, if $X_{|A|-1} \not\subset Z$, then then $Z \subseteq X_n \setminus \{x_i\}$ for some $1 \leq i \leq |A| - 1$. ■

Lemma 4.5.10 Let $n \geq |A|$. If the summands of u are a proper subset of the summands of Θ_n, then $a_1X_n + u \not\trianglerighteq_{\mathrm{R}} u$.

Proof: Suppose that all summands of u are summands of Θ_n, but that some summand a_1Y of Θ_n is not a summand of u. We consider the three possible forms of Y, and for each case give a closed substitution ρ such that some ready pair of $\rho(a_1X_n)$ is not a ready pair of $\rho(u)$.

Case 1: $Y = X_{|A|-1}$.

We define $\rho(x_i) = a_i\mathbf{0}$ for $i = 1, \ldots, |A|-1$, $\rho(x_i) = \mathbf{0}$ for $i = |A|, \ldots, n$, and $\rho(y) = a_{|A|}\mathbf{0}$ for $y \notin X_n$. Then the ready pair $(a_1, \{a_1, \ldots, a_{|A|-1}\})$ of $\rho(a_1X_n)$ is not a ready pair of $\rho(u)$.

Case 2: $Y = X_n \setminus \{x_{i_0}\}$ for some $1 \leq i_0 \leq |A|-1$.

We define $\rho(x_i) = a_i\mathbf{0}$ for $i = 1, \ldots, i_0-1, i_0+1, \ldots, |A|$, $\rho(x_i) = \mathbf{0}$ for $i = i_0$ and $i = |A|+1, \ldots, n$, and $\rho(y) = a_{i_0}\mathbf{0}$ for $y \notin X_n$. Then the ready pair $(a_1, \{a_1, \ldots, a_{i_0-1}, a_{i_0+1}, \ldots, a_{|A|}\})$ of $\rho(a_1X_n)$ is not a ready pair of $\rho(u)$.

Case 3: $Y = X_{|A|-1} \cup \{x_{i_0}, \hat{x}_{i_0}\}$ for some $|A| \leq i_0 \leq n$.

We define $\rho(x_i) = a_i\mathbf{0}$ for $i = 1, \ldots, |A|-1$, $\rho(x_{i_0}) = a_{|A|}\mathbf{0}$, and $\rho(y) = \mathbf{0}$ for $y \notin X_{|A|-1} \cup \{x_{i_0}\}$. Then the ready pair $(a_1, \{a_1, \ldots, a_{|A|}\})$ of $\rho(a_1X_n)$ is not a ready pair of $\rho(u)$.

The proof is now complete. ■

Lemma 4.5.11 Let $n \geq |A|$. If $\Theta_n \simeq_{\mathrm{R}} u$, then each summand of u is a summand of $a_1X_n + \Theta_n$.

Proof: Let $\Theta_n \simeq_{\mathrm{R}} u$. By Lem. 4.2.1(1), $depth(u) = 1$. By Lem. 4.2.1(3), u does not have summands $x \in V$, so clearly each summand of u is of the form a_1Y. If $|Y| \leq |A|-1$, then by Lem. 4.5.6, a_1Y is a summand of Θ_n. Let $|Y| \geq |A|$; we prove that a_1Y is a summand of $a_1X_n + \Theta_n$.

By Lem. 4.2.1(3), $Y \subseteq X_n \cup \{\hat{x}_i \mid i{=}|A|, \ldots, n\}$. We distinguish two cases.

Case 1: $\hat{x}_i \in Y$ for some $|A| \leq i \leq n$.

Suppose, towards a contradiction, that there is a $y \in Y \setminus (X_{|A|-1} \cup \{x_i, \hat{x}_i\})$. We define $\rho(y) = a_1\mathbf{0}$, $\rho(\hat{x}_i) = a_2\mathbf{0}$, and $\rho(z) = \mathbf{0}$ for $z \notin \{y, \hat{x}_i\}$. The ready pair $(a_1, \{a_1, a_2\})$ of $\rho(a_1Y)$ is not a ready pair of $\rho(\Theta_n)$, contradicting $\Theta_n \simeq_{\mathrm{R}} u$.

Suppose, towards a contradiction, that there is an $x \in (X_{|A|-1} \cup \{x_i, \hat{x}_i\}) \setminus Y$. Note that $\hat{x}_i \in Y$ implies $x \neq \hat{x}_i$. We define $\rho(x) = a_1\mathbf{0}$, $\rho(\hat{x}_i) = a_2\mathbf{0}$, and $\rho(z) = \mathbf{0}$ for $z \notin \{x, \hat{x}_i\}$. The ready pair $(a_1, \{a_2\})$ of $\rho(a_1Y)$ is not a ready pair of $\rho(\Theta_n)$, contradicting $\Theta_n \simeq_{\mathrm{R}} u$.

Hence, $Y = X_{|A|-1} \cup \{x_i, \hat{x}_i\}$.

Case 2: $Y \subseteq X_n$.

Since $|Y| \geq |A|$, there is a $Z = \{z_1, \ldots, z_{|A|-1}\} \subseteq Y$ with $Z \not\subseteq X_{|A|-1}$. We define $\rho(z_i) = a_i\mathbf{0}$ for $i = 1, \ldots, |A|-1$, $\rho(y) = \mathbf{0}$ for $y \in Y \setminus Z$, and $\rho(z) = a_{|A|}\mathbf{0}$ for $z \notin Y$. The ready pair $(a_1, \{a_1, \ldots, a_{|A|-1}\})$ of $\rho(a_1Y)$ must be a ready pair of $\rho(\Theta_n)$, which implies that there is a summand a_1Y' of Θ_n with $Z \subseteq Y' \subseteq Y$. Since $Z \not\subseteq X_{|A|-1}$ and $Y \subseteq X_n$, it follows that $Y' = X_n \setminus \{x_{i_0}\}$ for some $1 \leq i_0 \leq |A|-1$. Hence, either $Y = X_n$ or $Y = X_n \setminus \{x_{i_0}\}$.

Concluding, each summand of u is a summand of $a_1X_n + \Theta_n$. ■

The following example shows that Lem. 4.5.11 would fail if $|A| = 1$.

Example 4.5.12 Let $|A| = 1$ and $n = 1$. Note that $\Theta_1 = a_1\mathbf{0} + a_1(x_1 + \hat{x}_1)$ and $a_1X_1 = a_1x_1$. Since $|A| = 1$, $a_1\mathbf{0}+a_1(x_1+\hat{x}_1) \simeq_{\mathrm{R}} a_1\hat{x}_1+a_1\mathbf{0}+a_1(x_1+\hat{x}_1)$. However, $a_1\hat{x}_1$ is not a summand of $a_1x_1 + a_1\mathbf{0} + a_1(x_1 + \hat{x}_1)$.

Proposition 4.5.13 $C^n \nvdash a_1X_n + \Theta_n \approx \Theta_n$ for $n \geq |A|$.

Proof: Suppose, towards a contradiction, that there is a derivation of $a_1X_n + \Theta_n \approx \Theta_n$ using only equations in C^n: $a_1X_n+\Theta_n = u_0 \approx u_1 \approx \cdots \approx u_j = \Theta_n$ for some $j \geq 1$. By Lem. 4.2.1(1), $u_1, \ldots, u_j$ have depth 1. Since $u_0 = a_1X_n + \Theta_n$, $u_j = \Theta_n$, and the equations in C^n are of the form $aY + v \approx v$, there must be a $1 \leq i \leq j$ such that $u_{i-1} = a_1X_n + u_i$ and a_1X_n is not a summand of u_i. Since $\Theta_n \simeq_{\mathrm{R}} u_i$, Lem. 4.5.11 implies that all summands of u_i are summands of Θ_n. Since $a_1X_n + u_i \simeq_{\mathrm{R}} u_i$, Lem. 4.5.10 implies that $u_i = \Theta_n$. Hence, $a_1X_n + \Theta_n \approx \Theta_n$ can be derived using a single application of an equation $a_1Y + v \approx v \in C^n$. Then $\sigma(Y) = X_n$ and $\sigma(v) + w = \Theta_n$ for some substitution σ and term w. Since $a_1X_n + \sigma(v) \simeq_{\mathrm{R}} \sigma(v)$ and $\sigma(v) + w = \Theta_n$, Lem. 4.5.10 implies that $\sigma(v) = \Theta_n$. However, $a_1Y + v \approx v \in C^n$ implies $S(v) \leq n$, and v does not contain summands from V, so clearly $S(\sigma(v)) \leq n$. This contradicts the fact that $S(\sigma(v)) = S(\Theta_n) = n+1$. Concluding, $C^n \nvdash a_1X_n + \Theta_n \approx \Theta_n$. ■

Theorem 4.5.14 Let $1 < |A| < \infty$. Let $\simeq$ be a congruence that is included in ready equivalence and includes possible worlds equivalence. Then the equational theory of BCCSP modulo $\simeq$ is not finitely based.

Proof: Let E be a finite axiomatization that is sound and ground-complete for BCCSP modulo a congruence $\simeq$ that is included in ready equivalence and includes possible worlds equivalence. Suppose, towards a contradiction, that E is ω-complete. By Lem. 4.5.9 and 4.5.3, $a_1X_n + \Theta_n \approx \Theta_n$ for $n \geq |A|$ is sound modulo $\simeq_{\mathrm{PW}}$, so also modulo $\simeq$. Then these equations can be derived from E. Let E_1 denote the equations in E of depth ≤ 1. By Lem. 4.2.1(1), $E_1 \vdash a_1X_n + \Theta_n \approx \Theta_n$ for $n \geq |A|$.

Choose an $n \geq |A|$ such that $S(t) \leq n$ and $S(u) \leq n$ for each $t \approx u \in E_1$. Since E_1 is sound modulo $\simeq$, so also modulo $\simeq_{\mathrm{R}}$, it follows that $E_1 \subseteq R_1^n$. By Prop. 4.5.7, $C^n \vdash E_1$. This implies that $C^n \vdash a_1X_n + \Theta_n \approx \Theta_n$, which contradicts Prop. 4.5.13.

Concluding, E is not ω-complete. ■

4.6 Simulation

In this section we consider simulation equivalence $\simeq_{\mathrm{S}}$. In [vG90, vG01], van Glabbeek gave a finite axiomatization that is sound and ground-complete for

BCCSP modulo $\simeq_{\mathrm{S}}$. It consists of axioms A1-4 together with

$$\mathrm{S} \qquad a(x+y) \approx a(x+y) + ax \ ,$$

where a ranges over A. In case A is infinite, Groote's technique of inverted substitutions from [Gro90] can be applied in a straightforward fashion to prove that van Glabbeek's axiomatization is ω-complete; see [CF06].

An infinite supply of actions is crucial in this particular application of the inverted substitutions technique, for we shall prove below that the equational theory of BCCSP modulo $\simeq_{\mathrm{S}}$ does not have a finite basis if $1 < |A| < \infty$. The cornerstone for this negative result is the following infinite family of equations:

$$a(x+\Psi_n) + \sum_{\theta\in A^n} a(x+\Psi_n^\theta) + a\Phi_n \approx \sum_{\theta\in A^n} a(x+\Psi_n^\theta) + a\Phi_n \qquad (n \geq 0) \ .$$

Here, the Φ_n are defined inductively as follows:

$$\left\{ \begin{array}{lcl} \Phi_0 & = & \mathbf{0} \\ \Phi_{n+1} & = & \sum_{b\in A} b\Phi_n \end{array} \right. .$$

Moreover, the Ψ_n and Ψ_n^θ are defined by:

$$\begin{array}{lcll} \Psi_n & = & \sum_{b_1\cdots b_n\in A^n} b_1\cdots b_n\mathbf{0} & \\ \Psi_n^\theta & = & \sum_{b_1\cdots b_n\in A^n\setminus\{\theta\}} b_1\cdots b_n\mathbf{0} & \text{for } \theta\in A^n \ . \end{array}$$

For any closed term p with $\mathit{depth}(p) \leq n$, clearly $p \precsim_{\mathrm{S}} \Phi_n$. So in particular, $\Psi_n \precsim_{\mathrm{S}} \Phi_n$.

It is not hard to see that the equations above are sound modulo $\simeq_{\mathrm{S}}$. The idea is that, given a closed substitution ρ, either $\mathit{depth}(\rho(x)) < n$, in which case $a(\rho(x)+\Psi_n)$ is simulated by $a\Phi_n$. Or some $b_1\cdots b_n \in A^n$ is a trace of $\rho(x)$, in which case $a(\rho(x)+\Psi_n)$ is simulated by $a(\rho(x)+\Psi_n^{b_1\cdots b_n})$.

We shall prove below that $\simeq_{\mathrm{S}}$ is not finitely based, using the proof-theoretic technique, by showing that whenever an equation $t \approx u$ is derivable from a set of sound axioms of depth $\leq n$, then it satisfies the following property P_n^{S}:

> If $t, u \precsim_{\mathrm{S}} \sum_{\theta\in A^n} a(x+\Psi_n^\theta) + a\Phi_n$, then t has a summand similar to $a(x+\Psi_n)$ iff u has a summand similar to $a(x+\Psi_n)$.

We shall first establish in Lem. 4.6.2 that an equation satisfies P_n^{S} if it is a substitution instance of a sound equation of terms with a depth $\leq n$. Then, in Prop. 4.6.3, we prove, using Lem. 4.6.2, that P_n^{S} holds for every equation derivable from a collection of sound equations E, provided that the depth of the terms in E does not exceed n. From the proposition we can directly infer that the infinite family of equations above obstructs a finite basis, because the left-hand side contains a summand similar to $a(x+\Psi_n)$, while the right-hand side does not.

The following lemma constitutes an important step in the proof that P_n^{S} is preserved by substitution instances of sound equations of terms with a depth $\leq n$.

Lemma 4.6.1 If $a(x+\Psi_n) \precsim_{\mathrm{S}} at \precsim_{\mathrm{S}} \sum_{\theta\in A^n} a(x+\Psi_n^\theta) + a\Phi_n$, then $at \simeq_{\mathrm{S}} a(x+\Psi_n)$.

Proof: Since $x+\Psi_n \precsim_{\mathrm{S}} t$, by Lem. 4.2.1(3), x is a summand of t. Clearly, there exists a term t' that does not have x as a summand such that $t = x+t'$ (modulo A3). Since $a(x+t') \precsim_{\mathrm{S}} \sum_{\theta\in A^n} a(x+\Psi_n^\theta)+a\Phi_n$, by Lem. 4.2.1(3), t' is a closed term.

We prove that $t' \precsim_{\mathrm{S}} \Psi_n$. Consider a closed substitution ρ with $\rho(x) = a^{n+1}\mathbf{0}$. Since $a(\rho(x)+t') \precsim_{\mathrm{S}} \sum_{\theta\in A^n} a(\rho(x)+\Psi_n^\theta)+a\Phi_n$ and clearly $\rho(x)+t' \not\precsim_{\mathrm{S}} \Phi_n$, it follows that $\rho(x)+t' \precsim_{\mathrm{S}} \rho(x)+\Psi_n^\theta$ for some $\theta\in A^n$. Hence $t' \precsim_{\mathrm{S}} a^{n+1}\mathbf{0}+\Psi_n^\theta$. Since $at \precsim_{\mathrm{S}} \sum_{\theta\in A^n} a(x+\Psi_n^\theta)+a\Phi_n$, by Lem. 4.2.1(1), $\mathit{depth}(t') \le \mathit{depth}(t) \le n$. So $t' \precsim_{\mathrm{S}} a^n\mathbf{0}+\Psi_n^\theta \precsim_{\mathrm{S}} \Psi_n$.

Then $at = a(x+t') \precsim_{\mathrm{S}} a(x+\Psi_n)$, and, by assumption, $a(x+\Psi_n) \precsim_{\mathrm{S}} at$, so $at \simeq_{\mathrm{S}} a(x+\Psi_n)$. ■

We shall now establish that substitution instances of sound equations of depth $\le n$ satisfy P_n^{S}.

Lemma 4.6.2 Suppose $t \simeq_{\mathrm{S}} u$, let $n>1$ be a natural number greater than or equal to the depth of t and u, and suppose $\sigma(t), \sigma(u) \precsim_{\mathrm{S}} \sum_{\theta\in A^n} a(x+\Psi_n^\theta)+a\Phi_n$. Then $\sigma(t)$ has a summand similar to $a(x+\Psi_n)$ if and only if $\sigma(u)$ has a summand similar to $a(x+\Psi_n)$.

Proof: Clearly, by symmetry, it suffices to only consider the implication from left to right. So suppose that $\sigma(t)$ has a summand similar to $a(x+\Psi_n)$; then there are two cases:

Case 1: t has a variable summand z and $\sigma(z)$ has a summand similar to $a(x+\Psi_n)$.

Since $t \simeq_{\mathrm{S}} u$, by Lem. 4.2.1(3), u also has z as summand. Since $\sigma(z)$ has a summand similar to $a(x+\Psi_n)$, the same holds for $\sigma(u)$.

Case 2: t has a summand at' and $\sigma(at') \simeq_{\mathrm{S}} a(x+\Psi_n)$.

Note that from $\sigma(t') \simeq_{\mathrm{S}} x+\Psi_n$ it follows by Lem. 4.2.1(3) that x is a summand of $\sigma(t')$, and this means that t' has a variable summand y with x a summand of $\sigma(y)$.

The following claim constitutes a crucial step in the remainder of the proof for this case.

Claim. The term u has a summand au' such that, for every $m \ge 0$ and for every variable z, if $t' \xrightarrow{a_1\cdots a_m} z+v$ for some term v, then $u' \xrightarrow{a_1\cdots a_m} z+w$ for some term w.

Proof of the claim: We consider the terms t and u under a special closed substitution ρ, that we now proceed to define. Let a and b be distinct

actions, and let $\ulcorner . \urcorner : V \to \mathbb{Z}_{>0}$ be an injection (which exists since V is countable); then ρ is defined by

$$\rho(z) = a^{\ulcorner z \urcorner \cdot n} b\mathbf{0} \quad \text{for all } z \in V.$$

From the assumption that $t \simeq_{\mathrm{S}} u$, it follows that $\rho(t) \simeq_{\mathrm{S}} \rho(u)$. Since $\rho(t) \xrightarrow{a} \rho(t')$, there exists a closed term p such that $\rho(u) \xrightarrow{a} p$ and $\rho(t') \precsim_{\mathrm{S}} p$.

To establish that u has a summand au' such that $\rho(t') \precsim_{\mathrm{S}} \rho(u')$, we argue that u cannot have a variable summand z such that $\rho(z) \xrightarrow{a} p$. Recall that t' has a variable summand y; since $\rho(y) = a^{\ulcorner y \urcorner \cdot n} b\mathbf{0}$ and $\rho(t') \precsim_{\mathrm{S}} p$, it follows that b has an occurrence at depth $\ulcorner y \urcorner \cdot n$ in p. Now assume towards a contradiction that z is a variable summand of u such that $\rho(z) \xrightarrow{a} p$. Then $p = a^{\ulcorner z \urcorner \cdot n - 1} b\mathbf{0}$, which, since clearly $\ulcorner y \urcorner \cdot n \neq \ulcorner z \urcorner \cdot n - 1$, contradicts that b occurs in p at depth $\ulcorner y \urcorner \cdot n$ in p. So u has a summand au' such that $\rho(t') \precsim_{\mathrm{S}} \rho(u')$.

Now suppose that $t' \xrightarrow{a_1 \cdots a_m} z + v$ for some term v. Then, since $\rho(t') \precsim_{\mathrm{S}} \rho(u')$, there exists a closed term q such that $\rho(u') \xrightarrow{a_1 \cdots a_m} q$ and $\rho(z + v) \precsim_{\mathrm{S}} q$.

We shall now first prove that there exists u'' such that $u' \xrightarrow{a_1 \cdots a_m} u''$ and $\rho(u'') = q$. Assume towards a contradiction that there is no such u''. Then clearly there exist $\ell < m$, a variable z', and a term u''' such that $u' \xrightarrow{a_1 \cdots a_\ell} z' + u'''$ and $\rho(z') \xrightarrow{a_{\ell+1} \cdots a_m} q$. Since $\rho(z') = a^{\ulcorner z' \urcorner \cdot n} b\mathbf{0}$, it follows that $q = a^{\ulcorner z' \urcorner \cdot n - (m-\ell)} b\mathbf{0}$, and hence the single occurrence of b in p is at depth $\ulcorner z' \urcorner \cdot n - (m - \ell)$. Since $0 < m - \ell < n$, it follows that b does not occur at depth $\ulcorner z \urcorner \cdot n$ in q; this contradicts $\rho(z + v) \precsim_{\mathrm{S}} q$.

So there exists u'' such that $u \xrightarrow{a_1 \cdots a_m} u''$ and $\rho(u'') = q$. Since $\rho(z + v) \precsim_{\mathrm{S}} q = \rho(u'')$ and $\rho(z) = a^{\ulcorner z \urcorner \cdot n} b\mathbf{0}$, $\rho(u'') \xrightarrow{a^{\ulcorner z \urcorner \cdot n}} b\mathbf{0}$. Hence, since $\mathit{depth}(u'') < n$ and $\ulcorner z \urcorner > 0$, there exists a variable z', a term w, and $\ell < n$ such that $u'' \xrightarrow{a^\ell} z' + w$ and $\rho(z') \xrightarrow{a^{\ulcorner z \urcorner \cdot n - \ell}} b\mathbf{0}$. From the definition of ρ it is clear that $\ulcorner z \urcorner \cdot n - \ell = \ulcorner z' \urcorner \cdot n$. Since $\ell \leq \mathit{depth}(u'') < n$, it follows that $\ell = 0$, so $\ulcorner z' \urcorner = \ulcorner z \urcorner$, and hence, since $\ulcorner . \urcorner$ is an injection, $z' = z$. We have established that $u'' = z + w$, and thereby the proof of the claim is complete. ■

Now consider any $a_1 \cdots a_n \in A^n$. Since $\Psi_n \precsim_{\mathrm{S}} \sigma(t')$ and $\mathit{depth}(t') < n$, there exist $0 \leq m < n$, a variable z and a term v such that $t' \xrightarrow{a_1 \cdots a_m} z + v$ and $a_{m+1} \cdots a_n$ a trace of $\sigma(z)$. By our claim above, $u' \xrightarrow{a_1 \cdots a_m} z + w$ for some term w. Since $a_{m+1} \cdots a_n$ is a trace of $\sigma(z)$, it follows that $a_1 \cdots a_n$ is a trace of $\sigma(u')$. This holds for all $a_1 \cdots a_n \in A^n$, so $\Psi_n \precsim_{\mathrm{S}} \sigma(u')$.

Furthermore, recall that y is a summand of t', and that x is a summand of $\sigma(y)$. Since $t' \xrightarrow{\lambda} t'$ (with λ the empty sequence of actions), by our claim it follows that $u' \xrightarrow{\lambda} u + w$ for some term w. So y is a summand of u', and hence x is a summand of $\sigma(u')$.

We conclude that $x + \Psi_n \precsim_S \sigma(u')$, and hence $a(x + \Psi_n) \precsim_S a\sigma(u')$. From the assumption of the lemma that $\sigma(u) \precsim_S \sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n$ it follows that $a\sigma(u') \precsim_S \sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n$. So, by Lem. 4.6.1, $a\sigma(u') \simeq_S a(x + \Psi_n)$.

The proof is now complete. ■

We shall now prove that P_n^S holds for every equation derivable from a collection of equations between terms of depth less than or equal to n. By the preceding lemma, it only remains to prove that the transitivity and congruence rules preserve P_n^S.

Proposition 4.6.3 Let E be a finite axiomatization over BCCSP that is sound modulo $\simeq_S$, let n be a natural number greater than the depth of any term in E, and suppose $E \vdash t \approx u$ and $t, u \precsim_S \sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n$. Then t has a summand similar to $a(x + \Psi_n)$ iff u has a summand similar to $a(x + \Psi_n)$.

Proof: We prove the proposition by induction on the depth of a normalized derivation of the equation $t \approx u$ from E.

To establish the base case, note that if the derivation of $t \approx u$ consists of an application of the reflexivity rule, then the proposition is immediate, and if there exist terms v and w and a substitution σ such that $\sigma(v) = t$, $\sigma(w) = u$, and $(v \approx w) \in E$ or $(w \approx v) \in E$, then $v \simeq_S w$ by the soundness of E, so the proposition follows from Lem. 4.6.2.

For the inductive case we distinguish cases according to the last rule applied.

Case 1: the last rule applied is the transitivity rule.

Then there exist a term v and normalized derivations of $t \approx v$ and $v \approx u$. By the soundness of E, $v \simeq_S u \precsim_S \sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n$. So, by the induction hypothesis, v has a summand similar to $a(x + \Psi_n)$, and hence, again by the induction hypothesis, u has a summand similar to $a(x + \Psi_n)$.

Case 2: the last rule applied is the congruence rule for a.

Then $t = at'$ and $u = au'$ for some terms t' and u', and there exists a normal derivation of $t' \approx u'$. Since t consists of a single summand, $at' \simeq_S a(x + \Psi_n)$. So, by the soundness of E, $u = au' \simeq_S a(x + \Psi_n)$.

Case 3: the last rule applied is the congruence rule for $+$.

Then $t = t_1 + t_2$ and $u = u_1 + u_2$ for some terms t_1, t_2, u_1 and u_2, and there exist normal derivations of $t_1 \approx u_1$ and $t_2 \approx u_2$. Since t has a summand similar to $a(x + \Psi_n)$, so does either t_1 or t_2. Assume, without loss of generality, that t_1 has a summand completed similar to $a(x + \Psi_n)$. Then clearly $\mathcal{I}(t_1) = \mathcal{I}(u_1) = \{a\}$, so $t_1, u_1 \precsim_S t, u \precsim_S \sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n$. By the induction hypothesis, it follows that u_1, and hence u, has a summand similar to $a(x + \Psi_n)$.

The proof is now complete. ■

Now we are in a position to prove the main theorem of this section.

Theorem 4.6.4 Let $1 < |A| < \infty$. Then the equational theory of BCCSP modulo $\simeq_{\mathrm{S}}$ is not finitely based.

Proof: Let E be a finite axiomatization over BCCSP that is sound modulo $\simeq_{\mathrm{S}}$. Let $n > 1$ be greater than or equal to the depth of any term in E.

Note that $\sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n$ does not contain a summand similar to $a(x + \Psi_n)$. So according to Prop. 4.6.3, the equation

$$a(x + \Psi_n) + \sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n \approx \sum_{\theta \in A^n} a(x + \Psi_n^\theta) + a\Phi_n \ ,$$

which is sound modulo $\simeq_{\mathrm{S}}$, cannot be derived from E. It follows that every finite collection of equations that are sound modulo $\simeq_{\mathrm{S}}$ is necessarily incomplete, and hence the equational theory of BCCSP modulo $\simeq_{\mathrm{S}}$ is not finitely based. ■

4.7 Completed Simulation

In this section we consider completed simulation equivalence $\simeq_{\mathrm{CS}}$. In [vG90, vG01], van Glabbeek gave a finite axiomatization that is sound and ground-complete for BCCSP modulo $\simeq_{\mathrm{CS}}$. It consists of axioms A1-4 together with

$$\mathrm{CS} \qquad a(bx + y + z) \approx a(bx + y + z) + a(bx + z) \ ,$$

where a, b range over A. We prove that the equational theory of BCCSP modulo $\simeq_{\mathrm{CS}}$ does not have a finite basis if $|A| > 1$. (Note that our proof in this section also works in case $|A| = \infty$, whereas all the other proofs of negative results assume $|A| < \infty$). The cornerstone for this negative result is the following infinite family of equations:

$$a^n x + a^n \mathbf{0} + a^n(x + y) \approx a^n \mathbf{0} + a^n(x + y) \qquad (n \geq 1) \ .$$

It is not hard to see that these equations are sound modulo $\simeq_{\mathrm{CS}}$. The idea is that, given a closed substitution ρ, either $\rho(x)$ cannot perform any action, in which case $\rho(a^n x)$ is completed simulated by $a^n \mathbf{0}$, or x can perform some action, in which case $\rho(a^n x)$ is completed simulated by $\rho(a^n(x + y))$.

We shall prove that there cannot be a finite sound axiomatization E for BCCSP modulo $\simeq_{\mathrm{CS}}$ from which the equations above can all be derived. We apply the proof-theoretic technique, showing that if the axioms in E have depth smaller than n and the equation $t \approx u$ is derivable from E, then it satisfies the following property P_n^{CS}:

> If $t, u \precsim_{\mathrm{CS}} a^n \mathbf{0} + a^n(x + y)$, then t has a summand completed similar to $a^n x$ iff u has a summand completed similar to $a^n x$.

The crucial step is to prove that P_n^{CS} holds for all substitution instances of sound equations of depth $\leq n$ (see Lem. 4.7.1). The proof that the transitivity and congruence rules preserve P_n^{CS}, in Prop. 4.7.3, will then be analogous to our proof in the previous section that they preserve P_n^{S}. We infer that the infinite family of equations above obstructs a finite basis, by noting that the left-hand sides of the equations have a summand $a^n x$, while the right-hand sides do not.

The following lemma constitutes an crucial step in the proof that substitution instance of sound equations of depth $\leq n$ satisfy P_n^{CS}.

Lemma 4.7.1 If $at \precsim_{\mathrm{CS}} a^n\mathbf{0} + a^n(x+y)$ and $at \xrightarrow{a^n} t'$ with $t' = x$, then $at = a^n x$.

Proof: We first prove by induction on n that if $at \precsim_{\mathrm{CS}} a^n\mathbf{0} + a^n(x+y)$, then $at = a^n\mathbf{0}$ or $at = a^n x$ or $at = a^n y$ or $at = a^n(x+y)$.

Suppose $n = 1$. Then $\mathcal{I}(t) = \emptyset$ by Lem. 4.2.1(2) and $var_0(t) \subseteq \{x, y\}$ by Lem. 4.2.1(3), so $t = \mathbf{0}$ or $t = x$ or $t = y$ or $t = x + y$.

Suppose $n > 1$. Then by Lem. 4.2.1(2) $\mathcal{I}(t) = \{a\}$ and by Lem. 4.2.1(3) $var_0(t) = \emptyset$, so $t = \sum_{i\in I} at_i$ with $I \neq \emptyset$. Clearly, $at_i \precsim_{\mathrm{CS}} a^{n-1}\mathbf{0} + a^{n-1}(x+y)$, so by the induction hypothesis $at_i = a^{n-1}\mathbf{0}$ or $at_i = a^{n-1}x$ or $at_i = a^{n-1}y$ or $at_i = a^{n-1}(x+y)$, for all $i \in I$.

It remains to establish that $at_i = at_j$ for all $i, j \in I$. Suppose, towards a contradiction, that $at_i \neq at_j$ for some $i, j \in I$. Then clearly there exist t_i' and t_j' such that $at_i \xrightarrow{a^{n-1}} t_i'$, $at_j \xrightarrow{a^{n-1}} t_j'$ and $t_i' \neq t_j'$. Modulo symmetry we can distinguish six cases, and in each of them it suffices to provide a closed substitution ρ such that $\rho(at) \not\precsim_{\mathrm{CS}} \rho(a^n\mathbf{0} + a^n(x+y))$.

Cases 1,2,3: $t_i' = \mathbf{0}$ and $t_j' = x$ or $t_j' = y$ or $t_j' = x + y$.

Define ρ such that $\rho(x) \not\simeq_{\mathrm{CS}} \mathbf{0}$ and $\rho(y) \not\simeq_{\mathrm{CS}} \mathbf{0}$. Then $\rho(t) \not\precsim_{\mathrm{CS}} a^{n-1}\mathbf{0}$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_j') \not\simeq_{\mathrm{CS}} \mathbf{0}$), and $\rho(t) \not\precsim_{\mathrm{CS}} a^{n-1}\rho(x+y)$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_i) \simeq_{\mathrm{CS}} \mathbf{0}$ whereas $\rho(x+y) \not\simeq_{\mathrm{CS}} \mathbf{0}$). So $\rho(at) \not\precsim_{\mathrm{CS}} \rho(a^n\mathbf{0} + a^n(x+y))$.

Cases 4,5: $t_i' = x$ and $t_j' = y$ or $t_j' = x + y$.

Define ρ such that $\rho(x) = \mathbf{0}$ and $\rho(y) \not\simeq_{\mathrm{CS}} \mathbf{0}$. Then $\rho(t) \not\precsim_{\mathrm{CS}} a^{n-1}\mathbf{0}$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_j') \not\simeq_{\mathrm{CS}} \mathbf{0}$) and $\rho(t) \not\precsim_{\mathrm{CS}} a^{n-1}\rho(x+y)$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_i') \simeq_{\mathrm{CS}} \mathbf{0}$ and $\rho(x+y) \not\simeq_{\mathrm{CS}} \mathbf{0}$). So $\rho(at) \not\precsim_{\mathrm{CS}} \rho(a^n\mathbf{0} + a^n(x+y))$.

Case 6: $t_i' = y$ and $t_j' = x + y$.

Define ρ such that $\rho(x) \not\simeq_{\mathrm{CS}} \mathbf{0}$ and $\rho(y) = \mathbf{0}$. Then $\rho(t) \not\precsim_{\mathrm{CS}} a^{n-1}\mathbf{0}$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_j') \not\simeq_{\mathrm{CS}} \mathbf{0}$) and $\rho(t) \not\precsim_{\mathrm{CS}} a^{n-1}\rho(x+y)$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_i') \simeq_{\mathrm{CS}} \mathbf{0}$ and $\rho(x+y) \not\simeq_{\mathrm{CS}} \mathbf{0}$). So $\rho(at) \not\precsim_{\mathrm{CS}} \rho(a^n\mathbf{0} + a^n(x+y))$.

We have established that $at_i = at_j$ for all $i, j \in I$, so we may conclude that if $at \precsim_{\mathrm{CS}} a^n\mathbf{0}+a^n(x+y)$, then $at = a^n\mathbf{0}$ or $at = a^n x$ or $at = a^n y$ or $at = a^n(x+y)$. If, moreover, $at \xrightarrow{a^n} t'$ with $t' = x$, then it is easy to define closed substitutions showing that $at \neq a^n\mathbf{0}$, $at \neq a^n y$ and $at \neq a^n(x+y)$, so the proof of the lemma is complete. ■

In the following lemma we establish that substitution instances of sound equations of depth $< n$ satisfy P_n^{CS}.

Lemma 4.7.2 Suppose $t \simeq_{\mathrm{CS}} u$, let $n \geq 1$ be a natural number greater than the depth of t and u, and suppose $\sigma(t), \sigma(u) \precsim_{\mathrm{CS}} a^n\mathbf{0} + a^n(x+y)$. Then $\sigma(t)$ has a summand $a^n x$ iff $\sigma(u)$ has a summand $a^n x$.

Proof: Clearly, by symmetry, it suffices to establish the direction from left to right. So suppose $\sigma(t)$ has a summand $a^n x$; then there are two cases:

Case 1: t has a variable summand z and $\sigma(z)$ has a summand $a^n x$.

Then, since $t \simeq_{\mathrm{CS}} u$, by Lem. 4.2.1(3) u also has z as a summand, so clearly $\sigma(u)$ also has a summand $a^n x$.

Case 2: t has a summand at' and $\sigma(at') = a^n x$.

Then, since $\mathit{depth}(at') < n$, from $\sigma(at') = a^n x$ it follows that there exist a variable z and a term t'' such that $at' \xrightarrow{a^m} z + t''$ and $\sigma(z) = a^{n-m}x$ for some $1 \leq m < n$. Since $t \simeq_{\mathrm{CS}} u$, by Lem. 4.2.1(3), u has a summand au' such that $au' \xrightarrow{a^m} z+u''$ for some term u'', and consequently $a\sigma(u') \xrightarrow{a^n} u'''$ with $u''' = x$. Since also $a\sigma(u') \precsim_{\mathrm{CS}} \sigma(u) \precsim_{\mathrm{CS}} a^n\mathbf{0} + a^n(x+y)$, it follows by Lem. 4.7.1 that $a\sigma(u') = a^n x$. So $\sigma(u)$ has a summand $a^n x$.

The proof is now complete. ■

We shall now prove that if an equation derivable from a collection of equations of depth $< n$, then it satisfies P_n^{CS}.

Proposition 4.7.3 Let E be a finite axiomatization over BCCSP that is sound modulo $\simeq_{\mathrm{CS}}$, let n be a natural number greater than the depth of any term in E, and suppose $E \vdash t \approx u$ and $t, u \precsim_{\mathrm{CS}} a^n\mathbf{0}+a^n(x+y)$. Then t has a summand completed similar to $a^n x$ iff u has a summand completed similar to $a^n x$.

Proof: A straightforward adaptation of the proof of Prop. 4.6.3, using Lem. 4.7.2 instead of Lem. 4.6.2, replacing $\simeq_{\mathrm{S}}$ by $\simeq_{\mathrm{CS}}$, $\precsim_{\mathrm{S}}$ by $\precsim_{\mathrm{CS}}$, "similar" by "completed similar" and $\sum_{\theta\in A^n} a(x+\Psi_n^\theta) + a\Phi_n$ by $a^n\mathbf{0} + a^n(x+y)$. ■

Now we are in a position to prove the main theorem of this section.

Theorem 4.7.4 Let $|A| > 1$. Then the equational theory of BCCSP modulo $\simeq_{\mathrm{CS}}$ is not finitely based.

Proof: Let E be any finite axiomatization over BCCSP that is sound modulo $\simeq_{\mathrm{CS}}$ and let $n \geq 1$ greater than the depth of any term in E. Since $a^n\mathbf{0}+a^n(x+y)$ does not have a summand completed similar to $a^n x$, by Prop. 4.7.3 the equation

$$a^n x + a^n\mathbf{0} + a^n(x+y) \approx a^n\mathbf{0} + a^n(x+y) \ ,$$

which is sound modulo $\simeq_{\mathrm{CS}}$, cannot be derived from E. It follows that every finite collection of equations that are sound modulo $\simeq_{\mathrm{CS}}$ is necessarily incomplete, and hence the equational theory of BCCSP modulo $\simeq_{\mathrm{CS}}$ is not finitely based. ■

4.8 Ready Simulation

In this section we consider ready simulation equivalence $\simeq_{\mathrm{RS}}$. Blom, Fokkink and Nain [BFN03] gave a finite axiomatization that is sound and ground-complete for BCCSP modulo $\simeq_{\mathrm{RS}}$. It consists of axioms A1-4 together with the axiom RS presented at the start of Section 4.4.

Note that the equations in the infinite family presented in the previous section to show that $\simeq_{\mathrm{CS}}$ is not finitely based if $|A| > 1$, are not sound modulo $\simeq_{\mathrm{RS}}$. To see this, let a and b be distinct actions, and let ρ be a closed substitution such that $\rho(x) = a\mathbf{0}$ and $\rho(y) = b\mathbf{0}$. Then $\rho(a^n x)$ is not ready simulated by $\rho(a^n\mathbf{0})$ because $\mathcal{I}(\rho(x)) = \{a\} \neq \emptyset = \mathcal{I}(\mathbf{0})$, and $\rho(a^n x)$ is not ready simulated by $\rho(a^n(x+y)$, because $\mathcal{I}(\rho(x)) = \{a\} \neq \{a,b\} = \rho(x+y)$.

To obtain a negative result for $\simeq_{\mathrm{RS}}$, we proceed to consider below the following adaptation of the infinite family of equations of the previous section:

$$a^n x + a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0}) \approx a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0}) \qquad (n \geq 1) \ .$$

These equations are sound modulo $\simeq_{\mathrm{RS}}$. The idea is that, given a closed substitution ρ, either $\rho(x)$ cannot perform any action, in which case $\rho(a^n x)$ is ready simulated by $\rho(a^n\mathbf{0})$, or $\rho(x)$ can perform some action b, in which case $\rho(a^n x)$ is ready simulated by $\rho(a^n(x + b\mathbf{0}))$. Note, however, that the summations in the above equations only abbreviate BCCSP terms if $|A| < \infty$. So we assume $1 < |A| < \infty$ in the remainder of this section.

The condition $|A| < \infty$ is, in fact, necessary for the negative result that we are about to prove, for if $A = \infty$, then Groote's technique of inverted substitutions from [Gro90] can be applied in a straightforward fashion to prove that the axiomatization of Blom, Fokkink and Nain [BFN03] is ω-complete; see [CFN06].

The proof that there cannot be a finite sound axiomatization E for BCCSP modulo $\simeq_{\mathrm{RS}}$ from which the equations above can all be derived, is again with an application of the proof-theoretic technique. Let P_n^{RS} be the property:

> If $t, u \precsim_{\mathrm{RS}} a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$, then t has a summand ready similar to $a^n x$ iff u has a summand ready similar to $a^n x$.

Note that this is essentially the same property as P_n^{CS} of the previous section. Also the proof that P_n^{RS} is satisfied by every equation $t \approx u$ derivable from a collection of sound equations of depth $< n$ is analogous to the proof in the previous section. We only need to reconsider Lem. 4.7.1 in the light of the new family of equations.

Lemma 4.8.1 If $at \precsim_{\mathrm{RS}} a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$ and $at \xrightarrow{a^n} t'$ with $t' = x$, then $at = a^n x$.

Proof: We first prove by induction on n that if $at \precsim_{\mathrm{CS}} a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$, then $at = a^n\mathbf{0}$ or $at = a^n x$ or $at = a^n(x + b\mathbf{0})$ for some $b \in A$.

Suppose $n = 1$. Note that $var_0(t) \subseteq \{x\}$ by Lem. 4.2.1(3). Next, we establish that $\mathcal{I}(t) \subseteq \{b\}$ for some $b \in A$. To this end, let ρ be a closed substitution such that $\rho(x) = \mathbf{0}$. Then $\mathcal{I}(\rho(t)) = \mathcal{I}(\rho(\mathbf{0})) = \emptyset$ or $\mathcal{I}(\rho(t)) = \mathcal{I}(\rho(x + b\mathbf{0})) = \{b\}$ for some $b \in A$, and hence $\mathcal{I} \subseteq \{b\}$ for some $b \in A$. Now it has been shown that $t = \mathbf{0}$ or $t = x$ or $t = b\mathbf{0}$ or $t = x + b\mathbf{0}$. To exclude the case that $t = b\mathbf{0}$, suppose that $\mathcal{I}(t) = \{b\}$, and consider a substitution ρ such that $\rho(x) = c\mathbf{0}$ for some $c \neq b$. Since $\mathcal{I}(\rho(t)) \neq \mathcal{I}(\rho(\mathbf{0}))$ and $\mathcal{I}(\rho(t)) \neq \mathcal{I}(\rho(x + b'\mathbf{0}))$ for $b' \neq b$, it follows that $\mathcal{I}(\rho(t)) = \mathcal{I}(\rho(x + b\mathbf{0})) = \{b, c\}$. So $x \in var_0(t)$, and hence $t = x + b\mathbf{0}$.

Suppose $n > 1$. Then $\mathcal{I}(t) = \{a\}$ by Lem. 4.2.1(2) and $var_0(t) = \emptyset$ by Lem. 4.2.1(3), so $t = \sum_{i\in I} at_i$ with $I \neq \emptyset$. Clearly, $at_i \precsim_{\mathrm{RS}} a^{n-1}\mathbf{0} + \sum_{b\in A} a^{n-1}(x + b\mathbf{0})$, so by the induction hypothesis, for all $i \in I$, $at_i = a^{n-1}\mathbf{0}$ or $at_i = a^{n-1}x$ or $at_i = a^{n-1}(x + b_i\mathbf{0})$ for some $b_i \in A$.

It remains to establish that $at_i = at_j$ for all $i, j \in I$. Suppose, towards a contradiction, that $at_i \neq at_j$ for some $i, j \in I$. Then clearly there exist t'_i and t'_j such that $at_i \xrightarrow{a^{n-1}} t'_i$, $at_j \xrightarrow{a^{n-1}} t'_j$ and $t'_i \neq t'_j$. Modulo symmetry we can distinguish four cases, and in each of them it suffices to provide a closed substitution ρ such that $\rho(at) \not\precsim_{\mathrm{RS}} \rho(a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0}))$.

Cases 1,2: $t'_i = \mathbf{0}$ and $t'_j = x$ or $t'_j = x + b_j\mathbf{0}$.

> Define ρ such that $\rho(x) \not\simeq_{\mathrm{RS}} \mathbf{0}$. Then $\rho(t) \not\precsim_{\mathrm{RS}} a^{n-1}\mathbf{0}$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t'_j) \not\simeq_{\mathrm{RS}} \mathbf{0}$) and $\rho(t) \not\precsim_{\mathrm{RS}} a^{n-1}\rho(x + b_j\mathbf{0})$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t'_i) \simeq_{\mathrm{RS}} \mathbf{0}$).

Case 3: $t'_i = x$ and $t'_j = x + b_j\mathbf{0}$.

> Define ρ such that $\rho(x) = \mathbf{0}$. Then $\rho(t) \not\precsim_{\mathrm{RS}} a^{n-1}\mathbf{0}$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t'_j) \not\simeq_{\mathrm{RS}} \mathbf{0}$) and $\rho(t) \not\precsim_{\mathrm{RS}} a^{n-1}\rho(x + b_j\mathbf{0})$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t'_i) = \mathbf{0}$).

Case 4: $t'_i = x + b_i\mathbf{0}$ and $t'_j = x + b_j\mathbf{0}$ for some $b_i, b_j \in A$ with $b_i \neq b_j$.

> Define ρ such that $\rho(x) = \mathbf{0}$. Then $\rho(t) \not\precsim_{\mathrm{RS}} a^{n-1}\mathbf{0}$ (because $\rho(t) \xrightarrow{a^{n-1}} \rho(t'_i) \not\simeq_{\mathrm{RS}} \mathbf{0}$) and $\rho(t) \not\precsim_{\mathrm{RS}} a^{n-1}\rho(x + b\mathbf{0})$ for all $b \in A$ (because $b \neq b_k$ for $k = i$ or $k = j$, so that $\rho(t) \xrightarrow{a^{n-1}} \rho(t'_k) \simeq_{\mathrm{RS}} b_k\mathbf{0}$ and $\rho(x + b\mathbf{0}) \not\simeq_{\mathrm{RS}} b_k\mathbf{0}$).

We have established that $at_i = at_j$ for all $i, j \in I$, so we may conclude that if $at \precsim_{\mathrm{RS}} a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$, then $at = a^n\mathbf{0}$ or $at = a^n x$ or $at = a^n(x + b\mathbf{0})$ for some $b \in A$. If, moreover, $at \xrightarrow{a^n} t'$ with $t' = x$, then it is easy to define closed substitutions showing that $at \neq a^n\mathbf{0}$ and $at \neq a^n(x + b\mathbf{0})$, so the proof of the lemma is complete. ■

The following lemma corresponds to Lem. 4.7.2 of the previous section.

Lemma 4.8.2 Suppose $t \simeq_{\mathrm{RS}} u$, let $n \geq 1$ be a natural number greater than the depth of t and u, and suppose $\sigma(t), \sigma(u) \precsim_{\mathrm{RS}} a^n\mathbf{0} + \sum_{b\in A} a^n(x + b)$. Then $\sigma(t)$ has a summand $a^n x$ iff $\sigma(u)$ has a summand $a^n x$.

Proof: A straightforward adaptation of the proof of Lem. 4.7.2, using Lem. 4.8.1 instead of Lem. 4.7.1, replacing $\simeq_{\mathrm{CS}}$ by $\simeq_{\mathrm{RS}}$, $\precsim_{\mathrm{CS}}$ by $\precsim_{\mathrm{RS}}$, and $a^n\mathbf{0} + a^n(x + y)$ by $a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$. ■

The following proposition corresponds to Proposition 4.7.3 from the previous section.

Proposition 4.8.3 Let E be a finite axiomatization over BCCSP that is sound modulo $\simeq_{\mathrm{RS}}$, let n be a natural number greater than the depth of any term in E, and suppose $E \vdash t \approx u$ and $t, u \precsim_{\mathrm{RS}} a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$. Then t has a summand ready similar to $a^n x$ iff u has a summand ready similar to $a^n x$.

Proof: A straightforward adaptation of the proof of Prop. 4.6.3, using Lem. 4.8.2 instead of Lem. 4.6.2, replacing $\simeq_{\mathrm{S}}$ by $\simeq_{\mathrm{RS}}$, $\precsim_{\mathrm{S}}$ by $\precsim_{\mathrm{RS}}$, "similar" by "ready similar" and $\sum_{\theta\in A^n} a(x + \Psi_n^\theta) + a\Phi_n$ by $a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$. ■

Now we are in a position to prove the main theorem of this section.

Theorem 4.8.4 Let $1 < |A| < \infty$. Then the equational theory of BCCSP modulo $\simeq_{\mathrm{RS}}$ is not finitely based.

Proof: Let E be a finite axiomatization over BCCSP that is sound modulo $\simeq_{\mathrm{RS}}$. Let n be greater than the depth of any term in E.

Note that $a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0})$ does not contain a summand ready similar to $a^n x$. So according to Prop. 4.8.3, the equation

$$a^n x + a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0}) \approx a^n\mathbf{0} + \sum_{b\in A} a^n(x + b\mathbf{0}) \ ,$$

which is sound modulo $\simeq_{\mathrm{RS}}$, cannot be derived from E. It follows that every finite collection of equations that are sound modulo $\simeq_{\mathrm{RS}}$ is necessarily incomplete, and hence the equational theory of BCCSP modulo $\simeq_{\mathrm{RS}}$ is not finitely based. ■

4.9 Conclusion

For every equivalence in van Glabbeek's linear time – branching time spectrum I, it has now been determined whether it is finitely based or not. Tab. 4.1 presents an overview, with a + indicating that a finite basis exists and a − indicating that a finite basis does not exist. (The grey shadow indicates that it was open prior to the results described in this chapter.) We distinguish three categories, according to the cardinality of the alphabet A: singleton, finite with at least two actions, and infinite.

	$\|A\| = 1$	$1 < \|A\| < \infty$	$\|A\| = \infty$
bisimulation	+	+	+
2-nested simulation	−	−	−
possible futures	−	−	−
ready simulation	+	−	+
completed simulation	+	−	−
simulation	+	−	+
possible worlds	+	−	+
ready traces	+	−	−
failure traces	+	−	+
readies	+	−	+
failures	+	+	+
completed traces	+	+	+
traces	+	+	+

Table 4.1: The existence of finite bases for BCCSP in the linear time – branching time spectrum I

Chapter 5

Impossible Futures

5.1 Introduction

In this chapter, we study *impossible futures* semantics [Vog92, VM01]. This semantics is missing in van Glabbeek's original spectrum I, because it was only studied seriously from 2001 on, the year that [vG01] appeared. Impossible futures semantics is a natural variant of possible futures semantics [RB81] (see also Def. 2.1.4). It is also closely related to fair testing semantics [RV07]. In [vGV06] it was shown that *weak* impossible futures equivalence is the coarsest congruence with respect to choice and parallel composition operators containing weak bisimilarity with explicit divergence that respects deadlock/livelock traces and assigns unique solutions to recursive equations. For the definitions of impossible futures semantics, see Def. 2.1.4 and Def. 2.1.6.

In Chapter 4, a complete categorization of the (in)equational theories for the process algebra BCCSP modulo the semantics in the linear time – branching time spectrum I [vG01] has been given. For each preorder and equivalence it is studied whether a finite, sound, ground-complete axiomatization exists. And if so, whether there exists a finite basis for the equational theory.

So all questions on these matters regarding concrete semantics have been resolved? No, as for impossible futures semantics, the (in)equational theory remained unexplored, with the only exception that the inequational theory of BCCS modulo *weak* impossible futures preorder was studied in [VM01]. In that paper, Voorhoeve and Mauw offer a finite, sound, ground-complete axiomatization; their ground-completeness proof relies heavily on the presence of τ. They also prove that their axiomatization is ω-complete (they do not refer to ω-completeness explicitly, but they work on open terms, see [VM01, Thm. 5]). They implicitly assume an infinite alphabet (at [VM01, page 7] they require a different action for each variable).

As for weak semantics in general, much less is known, compared to concrete semantics. For several of the semantics in the linear time – branching time spectrum II [vG93b], a sound and *ground*-complete axiomatization has been given, in the setting of BCCS, see, e.g., [vG97]. Moreover a finite basis has been given for *weak*, *delay*, η- and *branching bisimulation* semantics [Mil89b, vG93a].

In this chapter, we focus on the axiomatizability of concrete and weak impossible futures preorders and equivalences in the setting of BCCSP and BCCS. In summary, we obtain the following results:

1. We prove that there exists a finite, sound, ground-complete axiomatization for BCCSP modulo concrete impossible futures *preorder* $\precsim_{\mathrm{IF}}$. By contrast, in [AFvGI04] it was shown that such an axiomatization does not exist modulo *possible* futures preorder. Thanks to the result established in Section 3.3, a finite, sound, ground-complete axiomatization for *weak* impossible futures preorder $\precsim_{\mathrm{WIF}}$ can be obtained for free.

2. We show that BCCS modulo weak impossible futures *equivalence* $\simeq_{\mathrm{WIF}}$ does *not* have a finite, sound, ground-complete axiomatization. This negative result is based on the following infinite family of equations, for $m \geq 0$:

$$\tau a^{2m}\mathbf{0} + \tau(a^m\mathbf{0} + a^{2m}\mathbf{0}) \;\approx\; \tau(a^m\mathbf{0} + a^{2m}\mathbf{0}) \;.$$

   Again thanks to the result established in Section 3.3, this negative result carries over to concrete impossible futures equivalence $\simeq_{\mathrm{IF}}$. Moreover, in light of this, one can easily establish the nonderivability of the equations $a^{2m+1}\mathbf{0} + a(a^m\mathbf{0} + a^{2m}\mathbf{0}) \;\approx\; a(a^m\mathbf{0} + a^{2m}\mathbf{0})$ from any given finite equational axiomatization sound for $\simeq_{\mathrm{WIF}}$. As these equations are valid modulo (concrete) 2-nested simulation equivalence, this negative result applies to all BCCS-congruences that are at least as fine as weak impossible futures equivalence and at least as coarse as concrete 2-nested simulation equivalence. Note that the corresponding result of [AFvGI04] can be inferred.

3. We investigate ω-completeness for impossible futures semantics.

   First, we prove that if the alphabet of actions is *infinite*, then the aforementioned ground-complete axiomatization for BCCSP modulo $\precsim_{\mathrm{IF}}$ is ω-complete. To prove this result, we apply the technique of inverted substitutions (see Section 3.4). The result established in Section 3.3 allows this result to carry over to $\precsim_{\mathrm{WIF}}$.

   Second, we prove that in case of a *finite* alphabet of actions, the inequational theory of BCCS modulo $\precsim_{\mathrm{WIF}}$ does *not* have a finite basis. In case of a singleton alphabet, this negative result is based on the following infinite family of equations, for $m \geq 0$:

$$a^m x \;\preccurlyeq\; a^m x + x \;.$$

   And for finite alphabets with at least two actions, we use the family

$$\tau(a^m x)+\tau(a^m x+x)+\sum_{b\in A}\tau(a^m x+a^m b\mathbf{0}) \preccurlyeq \tau(a^m x+x)+\sum_{b\in A}\tau(a^m x+a^m b\mathbf{0}) \;.$$

   The result established in Section 3.3 allows these negative results to carry over to $\precsim_{\mathrm{IF}}$.

4. *n-Nested* concrete impossible futures semantics, for $n \geq 0$, form a natural hierarchy (cf. [AFvGI04]), which coincides with the universal relation for $n = 0$, trace semantics for $n = 1$, and impossible futures semantics for $n = 2$. Using a proof strategy from [AFvGI04], we show that the negative result regarding concrete impossible futures equivalence extends to all n-nested impossible futures equivalences for $n \geq 2$, and to all n-nested impossible futures preorders for $n \geq 3$. Apparently, (2-nested) impossible futures preorder is the only positive exception.

To achieve these *negative* results, we mainly exploit the proof-theoretic technique (cf. Section 2.1.5). On top of this, a *saturation* principle is introduced, to transform a single summand into a large collection of (semi-)saturated summands, which plays a pivotal role in obtaining positive results.

To the best of our knowledge, impossible futures semantics is the first example that affords a finite, ground-complete axiomatization for BCCSP (or BCCS) modulo the *preorder*, while missing a finite, ground-complete axiomatization for BCCSP (or BCCS) modulo the *equivalence*. This surprising fact suggests that, for instance, if one wants to show $p \simeq_{\mathrm{IF}} q$ in general, one has to resort to deriving $p \precsim_{\mathrm{IF}} q$ and $q \precsim_{\mathrm{IF}} p$ separately, instead of proving it directly.

In [AFI07, dFG07] an algorithm is presented which produces, from an axiomatization for BCCSP modulo a preorder, an axiomatization for BCCSP modulo the corresponding equivalence. If the original axiomatization for the preorder is ground-complete or ω-complete, then so is the resulting axiomatization for the equivalence. In Section 3.2, we show that the same algorithm applies equally well to weak semantics. However, that algorithm only applies to semantics that are at least as coarse as *ready simulation semantics*. Since impossible futures semantics is incomparable to ready simulation semantics, it falls outside the scope of [AFI07, dFG07] and Section 3.2. Interestingly, our results yield that no such algorithm exists for certain semantics incomparable with (or finer than) ready simulation.

Structure of the chapter. Section 5.2 offers sound, finite, ground-complete axiomatizations for $\precsim_{\mathrm{IF}}$ and $\preceq_{\mathrm{WIF}}$; it also contains the proof of the negative result for $\simeq_{\mathrm{IF}}$ and $\simeq_{\mathrm{WIF}}$. Section 5.3 is devoted to the proofs of the negative and positive results regarding ω-completeness. Section 5.4 contains the negative results regarding n-nested (concrete) impossible futures semantics. Section 5.5 concludes the chapter with an overview of the positive and negative results achieved in this chapter.

5.2 Ground-Completeness

5.2.1 Concrete Impossible Futures Preorder

In this section, we provide a ground-complete axiomatization for impossible futures preorder. It consists of the core axioms A1-4 together with two extra axioms:

$$\begin{array}{lrcl} \text{IF1} & a(x+y) & \preccurlyeq & ax+ay \\ \text{IF2} & a(x+y)+ax+a(y+z) & \approx & ax+a(y+z) \ . \end{array}$$

Recall that here, $t \approx u$ denotes that both $t \preccurlyeq u$ and $u \preccurlyeq t$ are present in the inequational axiomatization. It is not hard to see that IF1-2 are sound modulo $\precsim_{\mathrm{IF}}$. The rest of this section is devoted to proving the following theorem.

Theorem 5.2.1 A1-4+IF1-2 is ground-complete for BCCSP(A) modulo $\precsim_{\mathrm{IF}}$.

To give some intuition on the ground-completeness proof, we first present an example.

Example 5.2.2 Let $p = a(a\mathbf{0} + a^2\mathbf{0}) + a^4\mathbf{0}$ and $q = a(a\mathbf{0} + a^3\mathbf{0}) + a^3\mathbf{0}$. It is not hard to see that $p \precsim_{\mathrm{IF}} q$. However, neither $a(a\mathbf{0}+a^2\mathbf{0}) \precsim_{\mathrm{IF}} a(a\mathbf{0}+a^3\mathbf{0})$ nor $a(a\mathbf{0} + a^2\mathbf{0}) \precsim_{\mathrm{IF}} a^3\mathbf{0}$ holds. In order to derive $p \preccurlyeq q$, we therefore first derive with IF2 that $q \approx p+q$. And $p \preccurlyeq p+q$ can be derived with IF1.

In general, to derive a sound closed inequation $p \preccurlyeq q$, first we derive $q \approx \mathbb{S}(q)$ (see Lem. 5.2.5), where $\mathbb{S}(q)$ contains for every $a \in \mathcal{I}(q)$ a "saturated" a-summand (see Def. 5.2.3). (In Ex. 5.2.2, this saturated a-summand would have the form $a(a\mathbf{0} + a^2\mathbf{0} + a^3\mathbf{0} + a(a\mathbf{0} + a^2\mathbf{0}))$.) Then, in the proof of Thm. 5.2.1, we derive $\Psi+\mathbb{S}(q) \approx \mathbb{S}(q)$ (equation (5.1)), $p \preccurlyeq \Psi$ (equation (5.2)) and $p \preccurlyeq p+q$ (equation (5.3)), where the closed term Ψ is built from many "semi-saturated" summands (like, in Ex. 5.2.2, p). These results together provide the desired proof (see the last line of the proof of Thm. 5.2.1).

Definition 5.2.3 For each closed term q, the closed term $\mathbb{S}(q)$ is defined inductively on the depth of q as follows:

$$\mathbb{S}(q) \ = \ q \ + \ \sum_{a \in \mathcal{I}(q)} a(\mathbb{S}(\sum_{aq' \Subset q} q')) \ .$$

Example 5.2.4 If $q = a(b(c\mathbf{0} + d\mathbf{0}) + be\mathbf{0}) + af\mathbf{0}$, then $\mathbb{S}(q) = a(b(c\mathbf{0} + d\mathbf{0}) + be\mathbf{0}) + af\mathbf{0} + a(b(c\mathbf{0} + d\mathbf{0}) + be\mathbf{0} + f\mathbf{0} + b(c\mathbf{0} + d\mathbf{0} + e\mathbf{0}))$.

Lemma 5.2.5 For each closed term q, A1-4+IF1-2 $\vdash q \approx \mathbb{S}(q)$.

Proof: By induction on $depth(q)$. For any $a \in \mathcal{I}(q)$,

$$q \ \approx \ q + a(\sum_{aq' \Subset q} q') \ \approx \ q + a(\mathbb{S}(\sum_{aq' \Subset q} q')) \ .$$

The first derivation step uses IF2, and the second induction. Hence, summing up over all $a \in \mathcal{I}(q)$,

$$q \ \approx \ q + \sum_{a \in \mathcal{I}(q)} a(\mathbb{S}(\sum_{aq' \Subset q} q')) \ = \ \mathbb{S}(q) \ .$$

■

For closed terms q and $\gamma \in \mathcal{T}(q)$, the closed term q_γ is obtained by summing over all closed terms q' such that $q \xrightarrow{\gamma} q'$, and then applying the saturation from Def. 5.2.3. The auxiliary terms q_γ will only be used in the derivation of equation (5.1) within the proof of Thm. 5.2.1.

Definition 5.2.6 Given a closed term q, and a completed trace $a_1 \cdots a_d$ of q. For $0 \leq \ell \leq d$ we define

$$Q_{a_1 \cdots a_\ell} \;=\; \{q_\ell \mid q \xrightarrow{a_1} q_1 \cdots \xrightarrow{a_\ell} q_\ell\} \;,$$

and

$$q_{a_1 \cdots a_\ell} \;=\; \mathbb{S}(\sum_{q_\ell \in Q_{a_1 \cdots a_\ell}} q_\ell) \;.$$

Note that $q_\varepsilon = \mathbb{S}(q)$. We prove some basic properties for the terms q_γ.

Lemma 5.2.7 Given a closed term q, and a completed trace $a_1 \cdots a_d$ of q. Then, for $0 \leq \ell < d$,

- $q_{a_1 \cdots a_\ell} \xrightarrow{a_{\ell+1}} q_{a_1 \cdots a_{\ell+1}}$; and
- $q_{a_1 \cdots a_\ell} \xrightarrow{a_{\ell+1}} q_{\ell+1}$ for all $q_{\ell+1} \in Q_{a_1 \cdots a_{\ell+1}}$.

Proof: Clearly, $q_{\ell+1} \in Q_{a_1 \cdots a_{\ell+1}}$ iff there exists some $q_\ell \in Q_{a_1 \cdots a_\ell}$ such that $q_\ell \xrightarrow{a_{\ell+1}} q_{\ell+1}$. And since $a_1 \cdots a_{\ell+1}$ is a trace of q, $a_{\ell+1} \in \mathcal{I}(q_\ell)$ for some $q_\ell \in Q_{a_1 \cdots a_\ell}$. So by Def. 5.2.3,

$$q_{a_1 \cdots a_\ell} \;=\; \mathbb{S}(\sum_{q_\ell \in Q_{a_1 \cdots a_\ell}} q_\ell) \xrightarrow{a_{\ell+1}} \mathbb{S}(\sum_{q_{\ell+1} \in Q_{a_1 \cdots a_{\ell+1}}} q_{\ell+1}) \;=\; q_{a_1 \cdots a_{\ell+1}} \;.$$

Moreover, for all $q_{\ell+1} \in Q_{a_1 \cdots a_{\ell+1}}$, we have $\sum_{q_\ell \in Q_{a_1 \cdots a_\ell}} q_\ell \xrightarrow{a_{\ell+1}} q_{\ell+1}$. Hence, by Def. 5.2.3,

$$q_{a_1 \cdots a_\ell} \;=\; \mathbb{S}(\sum_{q_\ell \in Q_{a_1 \cdots a_\ell}} q_\ell) \xrightarrow{a_{\ell+1}} q_{\ell+1} \;.$$

■

We now embark on proving the promised ground-completeness result.

Proof of Thm. 5.2.1: Suppose $p \precsim_{\mathrm{IF}} q$. We derive $p \preccurlyeq q$ using induction on $\mathit{depth}(p)$. If $p = \mathbf{0}$, then clearly $q = \mathbf{0}$, and we are done. So assume $p \neq \mathbf{0}$, and consider any completed path[1] $\pi = a_1 p_1 \cdots a_d p_d$ of p (with $d \geq 1$); that is, $p \xrightarrow{a_1} p_1 \cdots \xrightarrow{a_d} p_d = \mathbf{0}$. We inductively construct closed terms ψ_ℓ^π, for

[1] A sequence $a_1 p_1 \cdots a_k p_k$ is a *completed path* of a term p_0 if $p_0 \xrightarrow{a_1} p_1 \cdots \xrightarrow{a_k} p_k$ with $\mathcal{I}(p_k) = \emptyset$. We write $\mathcal{CP}(p)$ for the set of completed paths of term p, which is ranged over by π.

ℓ from d down to 1. For the base case, $\psi_d^\pi = \mathbf{0}$. Now let $1 \leq \ell < d$. Since $p \xrightarrow{a_1 \cdots a_\ell} p_\ell$ and $p \precsim_{\mathrm{IF}} q$, there exists a sequence of transitions $q \xrightarrow{a_1 \cdots a_\ell} q_\ell$ such that $\mathcal{T}(q_\ell) \subseteq \mathcal{T}(p_\ell)$. We define

$$\psi_\ell^\pi \;=\; q_\ell + a_{\ell+1}\psi_{\ell+1}^\pi \;.$$

We prove, by induction on $d - \ell$, that for $1 \leq \ell \leq d$,

$$\mathcal{T}(\psi_\ell^\pi) \;\subseteq\; \mathcal{T}(p_\ell) \;.$$

The base case is trivial, since $\mathcal{T}(\psi_d^\pi) = \emptyset$. Now let $1 \leq \ell < d$. By induction, $\mathcal{T}(\psi_{\ell+1}^\pi) \subseteq \mathcal{T}(p_{\ell+1})$. Moreover, $p_\ell \xrightarrow{a_{\ell+1}} p_{\ell+1}$, so $\mathcal{T}(a_{\ell+1}\psi_{\ell+1}^\pi) \subseteq \mathcal{T}(p_\ell)$. Hence, $\mathcal{T}(\psi_\ell^\pi) = \mathcal{T}(q_\ell + a_{\ell+1}\psi_{\ell+1}^\pi) = \mathcal{T}(q_\ell) \cup \mathcal{T}(a_{\ell+1}\psi_{\ell+1}^\pi) \subseteq \mathcal{T}(p_\ell)$.

Next, we prove, by induction on $d - \ell$, that for $1 \leq \ell \leq d$,

$$E \;\vdash\; a_\ell\psi_\ell^\pi + q_{a_1 \cdots a_{\ell-1}} \;\approx\; q_{a_1 \cdots a_{\ell-1}} \;.$$

In the base case, since $\psi_d^\pi = \mathbf{0} \in Q_{a_1 \cdots a_d}$ (see Def. 5.2.6), this is a direct consequence of the second item in Lem. 5.2.7. Now let $1 \leq \ell < d$.

$$\begin{aligned}
& a_\ell\psi_\ell^\pi + q_{a_1 \cdots a_{\ell-1}} \\
=\;& a_\ell(q_\ell + a_{\ell+1}\psi_{\ell+1}^\pi) + q_{a_1 \cdots a_{\ell-1}} + \; a_\ell q_\ell + a_\ell q_{a_1 \cdots a_\ell} && \text{(Lem. 5.2.7)} \\
\approx\;& a_\ell(q_\ell + a_{\ell+1}\psi_{\ell+1}^\pi) + q_{a_1 \cdots a_{\ell-1}} + \; a_\ell q_\ell + a_\ell(a_{\ell+1}\psi_{\ell+1}^\pi + q_{a_1 \cdots a_\ell}) && \text{(induction)} \\
\approx\;& q_{a_1 \cdots a_{\ell-1}} + a_\ell q_\ell + \; a_\ell(q_{a_1 \cdots a_\ell} + a_{\ell+1}\psi_{\ell+1}^\pi) && \text{(IF2)} \\
\approx\;& q_{a_1 \cdots a_{\ell-1}} + a_\ell q_\ell + \; a_\ell q_{a_1 \cdots a_\ell} && \text{(induction)} \\
=\;& q_{a_1 \cdots a_{\ell-1}} \;. && \text{(Lem. 5.2.7)}
\end{aligned}$$

In the end, for $\ell = 1$, we derive $a_1\psi_1^\pi + q_\varepsilon \approx q_\varepsilon$. In other words,

$$E \;\vdash\; a_1\psi_1^\pi + \mathbb{S}(q) \;\approx\; \mathbb{S}(q) \;.$$

Since this holds for all completed paths π of p, it follows that

$$E \vdash \sum_{a \in \mathcal{I}(p)} \sum_{ap' \in p} \sum_{\pi \in \mathcal{CP}(ap')} a\psi_1^\pi + \mathbb{S}(q) \;\approx\; \mathbb{S}(q) \;, \tag{5.1}$$

where $\mathcal{CP}(ap')$ denotes the set of completed paths of the summand ap'.

On the other hand, for every summand ap' of p,

$$p' \;\precsim_{\mathrm{IF}} \sum_{\pi \in \mathcal{CP}(ap')} \psi_1^\pi \;.$$

Namely, consider any path $\pi_0 = a_1 p_1 \cdots a_h p_h$ of ap'. Extend π_0 to some completed path π of ap'. By the definition of the ψ_ℓ^π, clearly, $\psi_\ell^\pi \xrightarrow{a_{\ell+1}} \psi_{\ell+1}^\pi$ for $1 \leq \ell < h$. So $\psi_1^\pi \xrightarrow{a_2 \cdots a_h} \psi_h^\pi$. Moreover, we proved that $\mathcal{T}(\psi_h^\pi) \subseteq \mathcal{T}(p_h)$.

So by induction on depth, for every summand ap' of p, we derive

$$p' \preccurlyeq \sum_{\pi\in\mathcal{CP}(ap')} \psi_1^\pi \ .$$

And thus, by IF1,

$$ap' \preccurlyeq a(\sum_{\pi\in\mathcal{CP}(ap')} \psi_1^\pi) \preccurlyeq \sum_{\pi\in\mathcal{CP}(ap')} a\psi_1^\pi \ .$$

Hence, summing over all summands ap' of p,

$$E \vdash p \preccurlyeq \sum_{a\in\mathcal{I}(p)} \sum_{ap'\Subset p} \sum_{\pi\in\mathcal{CP}(ap')} a\psi_1^\pi \ . \tag{5.2}$$

Finally, since $p \precsim_{\mathrm{IF}} q$, clearly, for each $a \in \mathcal{I}(p)$,

$$\sum_{ap'\Subset p} p' \precsim_{\mathrm{IF}} \sum_{aq'\Subset q} q' \ .$$

So by induction on depth, for each $a \in \mathcal{I}(p)$,

$$E \vdash \sum_{ap'\Subset p} p' \preccurlyeq \sum_{aq'\Subset q} q' \ .$$

So by IF2 and IF1, and since $\mathcal{I}(p) = \mathcal{I}(q)$,

$$p \approx p + \sum_{a\in\mathcal{I}(p)} a(\sum_{ap'\Subset p} p') \preccurlyeq p + \sum_{a\in\mathcal{I}(q)} a(\sum_{aq'\Subset q} q') \preccurlyeq p + \sum_{a\in\mathcal{I}(q)} \sum_{aq'\Subset q} aq' \ .$$

That is,

$$E \vdash p \preccurlyeq p + q \ . \tag{5.3}$$

In the end, (5.3), (5.2) and (5.1), together with Lem. 5.2.5, yield

$$E \vdash p \preccurlyeq p + q \approx p + \mathbb{S}(q) \preccurlyeq \sum_{a\in\mathcal{I}(p)} \sum_{ap'\Subset p} \sum_{\pi\in\mathcal{CP}(ap')} a\psi_1^\pi + \mathbb{S}(q) \approx \mathbb{S}(q) \approx q \ .$$

■

5.2.2 Weak Impossible Futures Preorder

We now apply the link established in Section 3.3 to obtain a ground-complete axiomatization for weak impossible future preorder $\precsim_{\mathrm{WIF}}$ for free. It consists of A1-4, IF1-2, W1-2 (see page 48), together with $x \preccurlyeq \tau x$ (W3). Again, this axiomatization can be simplified. It turns out that IF1 and IF2 are redundant. (To see this, we only need to observe that A1-4+W1-3 $\vdash a(x+y) \preccurlyeq a(\tau x+\tau y) \approx$

W1	$\alpha x + \alpha y$	$\approx$	$\alpha(\tau x + \tau y)$
W2′	$\tau(x + y)$	$\preccurlyeq$	$\tau x + y$
W3	x	$\preccurlyeq$	τx

Table 5.1: Axioms for weak impossible futures preorder

$ax + ay$ and A1-4+W1-3 $\vdash \tau(x + y) + \tau x + \tau(y + z) \approx \tau x + y + \tau(y + z) \approx \tau(y + z) + \tau x$. It follows that A1-4+W1-3 $\vdash$ IF2.) Furthermore, W2 can be replaced by W2′. (To see this, by W2′, $\tau(x + y) + \tau x \preccurlyeq \tau x + y$; by W3, $\tau x + y \approx \tau x + x + y \preccurlyeq \tau x + \tau(x + y)$.) In summary, the crucial axiomas are presented in Tab. 5.1. Clearly, W1-3 are sound for BCCS modulo $\preceq_{\mathrm{WIF}}$, hence,

Corollary 5.2.8 A1-4+W1+W2′+W3 is ground-complete for BCCS(A) modulo $\preceq_{\mathrm{WIF}}$.

Remark 5.2.9 A slightly more complicated axiomatization has been obtained in [VM01], where, instead of W2′, the axiom $\tau(\tau x+y) \approx \tau x+y$ is present. Here, it becomes a direct corollary from the axiomatization for concrete semantics. It is also very interesting to see the difference between failures and impossible futures semantics. It seems that the nuance lies in only one axiom, namely, for the weak impossible future preorder, $x \preccurlyeq \tau x$ suffices; while for the failures preorder, a stronger axiom $x \preccurlyeq \tau x + y$ has to be included. However, as revealed by the algorithm, we argue that the essential difference actually arises in the concrete case. Namely, between IF1-2 and F1.

5.2.3 Weak Impossible Futures Equivalence

We now prove that for any (nonempty) A there does *not* exist any finite, sound, ground-complete axiomatization for BCCS(A) modulo $\simeq_{\mathrm{WIF}}$. The cornerstone for this negative result is the following infinite family of closed equations, for $m \geq 0$:

$$\tau a^{2m}\mathbf{0} + \tau(a^m\mathbf{0} + a^{2m}\mathbf{0}) \;\approx\; \tau(a^m\mathbf{0} + a^{2m}\mathbf{0}) \;.$$

It is not hard to see that they are sound modulo $\simeq_{\mathrm{WIF}}$. We start with a few lemmas.

Lemma 5.2.10 If $p \preceq_{\mathrm{WIF}} q$ then $\mathcal{WCT}(p) \subseteq \mathcal{WCT}(q)$.

Proof: A process p has a weak completed trace $a_1 \cdots a_k$ iff it has a weak impossible future $(a_1 \cdots a_k, A)$. ■

Lemma 5.2.11 Suppose $t \preceq_{\mathrm{WIF}} u$. Then for any t' with $t \Rightarrow \xrightarrow{\tau} t'$ there is some u' with $u \Rightarrow \xrightarrow{\tau} u'$ such that $var(u') \subseteq var(t')$.

Proof: Let $t \Rightarrow\!\!\xrightarrow{\tau} t'$. Fix some $m > \mathit{depth}(t)$, and consider the closed substitution ρ defined by $\rho(x) = \mathbf{0}$ if $x \in \mathit{var}(t')$ and $\rho(x) = a^m\mathbf{0}$ if $x \notin \mathit{var}(t')$. Since $\rho(t) \Rightarrow \rho(t')$ with $\mathit{depth}(\rho(t')) = \mathit{depth}(t') < m$, and $\rho(t) \preceq_{\mathrm{WIF}} \rho(u)$, clearly $\rho(u) \Rightarrow q$ for some q with $\mathit{depth}(q) < m$. From the definition of ρ it then follows that there must exist $u \Rightarrow u'$ with $\mathit{var}(u') \subseteq \mathit{var}(t')$. In case $u \Rightarrow\!\!\xrightarrow{\tau} u'$ we are done, so assume $u' = u$. Let σ be the substitution with $\sigma(x) = \mathbf{0}$ for all $x \in V$. Since $\sigma(t) \xrightarrow{\tau}$ and $t \preceq_{\mathrm{WIF}} u$ we have $\sigma(u) \xrightarrow{\tau}$, so $u \xrightarrow{\tau} u''$ for some u''. Now $\mathit{var}(u'') \subseteq \mathit{var}(u) = \mathit{var}(u') \subseteq \mathit{var}(t')$. ■

Lemma 5.2.12 Assume that, for terms t, u, closed substitution σ, $a \in A$ and integer m:

1. $t \simeq_{\mathrm{WIF}} u$;
2. $m > \mathit{depth}(u)$;
3. $\mathcal{WCT}(\sigma(u)) \subseteq \{a^m, a^{2m}\}$; and
4. there is a closed term p' such that $\sigma(t) \Rightarrow\!\!\xrightarrow{\tau} p'$ and $\mathcal{WCT}(p') = \{a^{2m}\}$.

Then there is a closed term q' such that $\sigma(u) \Rightarrow\!\!\xrightarrow{\tau} q'$ and $\mathcal{WCT}(q') = \{a^{2m}\}$.

Proof: According to proviso (4) of the lemma, we can distinguish two cases.

- There exists some $x \in V$ such that $t \Rightarrow t'$ with $t' = t''+x$ and $\sigma(x) \Rightarrow\!\!\xrightarrow{\tau} p'$ where $\mathcal{WCT}(p') = \{a^{2m}\}$. Consider the closed substitution ρ defined by $\rho(x) = a^m\mathbf{0}$ and $\rho(y) = \mathbf{0}$ for any $y \neq x$. Then $a^m \in \mathcal{WCT}(\rho(t)) = \mathcal{WCT}(\rho(u))$, using Lem. 5.2.10, and this is only possible if $u \Rightarrow u'$ for some $u' = u'' + x$. Hence $\sigma(u) \Rightarrow\!\!\xrightarrow{\tau} p'$.
- $t \Rightarrow\!\!\xrightarrow{\tau} t'$ with $\mathcal{WCT}(\sigma(t')) = \{a^{2m}\}$. Since $\mathit{depth}(t') \leq \mathit{depth}(t) = \mathit{depth}(u) < m$, clearly, for any $x \in \mathit{var}(t')$, either $\mathit{depth}(\sigma(x)) = 0$ or $\mathit{norm}(\sigma(x)) > m$, where $\mathit{norm}(p)$ denotes the length of the shortest weak completed trace of p. Since $t \simeq_{\mathrm{WIF}} u$, by Lem. 5.2.11, $u \Rightarrow\!\!\xrightarrow{u} u'$ with $\mathit{var}(u') \subseteq \mathit{var}(t')$. Hence, for any $x \in \mathit{var}(u')$, either $\mathit{depth}(\sigma(x)) = 0$ or $\mathit{norm}(\sigma(x)) > m$. Since $\mathit{depth}(u') < m$, $a^m \notin \mathcal{WCT}(\sigma(u'))$. It follows from $\mathcal{WCT}(\sigma(u)) \subseteq \{a^m, a^{2m}\}$ that $\mathcal{WCT}(\sigma(u')) = \{a^{2m}\}$. And $u \Rightarrow\!\!\xrightarrow{\tau} u'$ implies $\sigma(u) \Rightarrow\!\!\xrightarrow{\tau} \sigma(u')$.

The proof is now complete. ■

Lemma 5.2.13 Assume that, for E an axiomatization sound for $\preceq_{\mathrm{WIF}}$, closed terms p, q, closed substitution σ, $a \in A$ and integer m:

1. $E \vdash p \approx q$;
2. $m > \max\{\mathit{depth}(u) \mid t \approx u \in E\}$;

3. $\mathcal{WCT}(q) \subseteq \{a^m, a^{2m}\}$; and
4. there is a closed term p' such that $p \Rightarrow\!\xrightarrow{\tau} p'$ and $\mathcal{WCT}(p') = \{a^{2m}\}$.

Then there is a closed term q' such that $q \Rightarrow\!\xrightarrow{\tau} q'$ and $\mathcal{WCT}(q') = \{a^{2m}\}$.

Proof: By induction on the derivation of $E \vdash p \approx q$.

- Suppose $E \vdash p \approx q$ because $\sigma(t) = p$ and $\sigma(u) = q$ for some $t \approx u \in E$ or $u \approx t \in E$ and closed substitution σ. The claim then follows by Lem. 6.4.2.
- Suppose $E \vdash p \approx q$ because $E \vdash p \approx r$ and $E \vdash r \approx q$ for some r. Since $r \simeq_{\mathrm{WIF}} q$, by proviso (3) of the lemma and Lem. 5.2.10, $\mathcal{WCT}(r) \subseteq \{a^m, a^{2m}\}$. Since there is a p' such that $p \Rightarrow\!\xrightarrow{\tau} p'$ with $\mathcal{WCT}(p') = \{a^{2m}\}$, by induction, there is an r' such that $r \Rightarrow\!\xrightarrow{\tau} r'$ and $\mathcal{WCT}(r') = \{a^{2m}\}$. Hence, again by induction, there is a q' such that $q \Rightarrow\!\xrightarrow{\tau} q'$ and $\mathcal{WCT}(q') = \{a^{2m}\}$.
- Suppose $E \vdash p \approx q$ because $p = p_1 + p_2$ and $q = q_1 + q_2$ with $E \vdash p_1 \approx q_1$ and $E \vdash p_2 \approx q_2$. Since there is a p' such that $p \Rightarrow\!\xrightarrow{\tau} p'$ and $\mathcal{WCT}(p') = \{a^{2m}\}$, either $p_1 \Rightarrow\!\xrightarrow{\tau} p'$ or $p_2 \Rightarrow\!\xrightarrow{\tau} p'$. Assume, without loss of generality, that $p_1 \Rightarrow\!\xrightarrow{\tau} p'$. By induction, there is a q' such that $q_1 \Rightarrow\!\xrightarrow{\tau} q'$ and $\mathcal{WCT}(q') = \{a^{2m}\}$. Now $q \Rightarrow\!\xrightarrow{\tau} q'$.
- Suppose $E \vdash p \approx q$ because $p = cp_1$ and $q = cq_1$ with $c \in A$ and $E \vdash p_1 \approx q_1$. In this case, proviso (4) of the lemma can not be met.
- Suppose $E \vdash p \approx q$ because $p = \tau p_1$ and $q = \tau q_1$ with $E \vdash p_1 \approx q_1$. By proviso (4) of the lemma, either $\mathcal{WCT}(p_1) = \{a^{2m}\}$ or there is a p' such that $p_1 \Rightarrow\!\xrightarrow{\tau} p'$ and $\mathcal{WCT}(p') = \{a^{2m}\}$. In the first case, $q \Rightarrow\!\xrightarrow{\tau} q_1$ and $\mathcal{WCT}(q_1) = \{a^{2m}\}$ by Lem. 5.2.10. In the second, by induction, there is a q' such that $q_1 \Rightarrow\!\xrightarrow{\tau} q'$ and $\mathcal{WCT}(q') = \{a^{2m}\}$. Again $q \Rightarrow\!\xrightarrow{\tau} q'$.

The proof is now complete. ■

Theorem 5.2.14 There is no finite, sound, ground-complete axiomatization for BCCS(A) modulo $\simeq_{\mathrm{WIF}}$.

Proof: Let E be a finite axiomatization over BCCS(A) that is sound modulo $\simeq_{\mathrm{WIF}}$. Let m be greater than the depth of any term in E. Clearly, there is no term r such that $\tau(a^m\mathbf{0} + a^{2m}\mathbf{0}) \Rightarrow\!\xrightarrow{\tau} r$ and $\mathcal{WCT}(r) = \{a^{2m}\}$. So according to Lem. 5.2.13, the closed equation $\tau a^{2m}\mathbf{0} + \tau(a^m\mathbf{0} + a^{2m}\mathbf{0}) \approx \tau(a^m\mathbf{0} + a^{2m}\mathbf{0})$ cannot be derived from E. Nevertheless, it is valid modulo $\simeq_{\mathrm{WIF}}$. ■

Remark 5.2.15 Lem. 5.2.13 does *not* hold if its first requirement is changed into $E \vdash p \preccurlyeq q$. Note that the proof regarding the congruence rule for $\tau_{\_}$ in

Lem. 5.2.13 fails for $\preceq_{\mathrm{WIF}}$. For example, consider the following closed inequations, for $m \geq 0$:

$$\tau a^{2m}\mathbf{0} \;\preccurlyeq\; \tau(a^{2m}\mathbf{0} + a^m\mathbf{0}) \ .$$

They are sound modulo $\preceq_{\mathrm{WIF}}$, and satisfy the third and fourth requirement of Lem. 5.2.13. However, they can all be derived by means of IF1:

$$\begin{aligned}\tau a^{2m}\mathbf{0} = \tau(a^m(a^m+\mathbf{0})) &\preccurlyeq \tau(a^{m-1}(a^{m+1}\mathbf{0} + a\mathbf{0}))\\ &\preccurlyeq \tau(a^{m-2}(a^{m+2}\mathbf{0} + a^2\mathbf{0})) \preccurlyeq \cdots \preccurlyeq \tau(a^{2m}\mathbf{0} + a^m\mathbf{0}) \ .\end{aligned}$$

5.2.4 Concrete Impossible Futures Equivalence

The link established in Section 3.3 together with Thm. 5.2.14 reveals the nonexistence of a finite, ground-complete axiomatization for *concrete* impossible future equivalences for BCCSP. That is,

Corollary 5.2.16 There is no finite, sound, ground-complete axiomatization for BCCSP(A) modulo $\simeq_{\mathrm{IF}}$.

5.3 ω-Completeness

5.3.1 Infinite Alphabet

In this section, we show that the axiomatization consisting of A1-4+IF1-2 is ω-complete, provided the alphabet is infinite. The proof is based on the *inverted substitution* technique, which is adapted from [Gro90] (see Section 3.4).

Theorem 5.3.1 If $|A| = \infty$, then A1-4+IF1-2 is ω-complete.

Proof: Consider any pair of BCCSP(A) terms t, u. Define the closed substitution ρ by $\rho(y) = a_y\mathbf{0}$, where a_y is a unique action for $y \in V$ that occurs in neither t nor u. Such actions exist because A is infinite. We define the mapping R from closed to open BCCSP(A) terms as follows:

$$\begin{cases} R(\mathbf{0}) & = \mathbf{0} \\ R(at) & = y \text{ if } a = a_y \text{ for some } y \in V \\ R(at) & = aR(t) \text{ if } a \neq a_y \text{ for all } y \in V \\ R(t+u) & = R(t) + R(u) \end{cases}$$

We check the three aforementioned properties.

(1) Since t and u do not contain actions of the form a_y, clearly $R(\rho(t)) = t$ and $R(\rho(u)) = u$.

(2) For A1-4, the proof is trivial. We check the remaining cases IF1 and IF2. Let σ be a closed substitution. For IF1, we distinguish two cases.

- $a = a_y$ for some $y \in V$. Then $R(a_y(\sigma(x_1) + \sigma(x_2))) = y \preccurlyeq y + y = R(a_y(\sigma(x_1)) + a_y(\sigma(x_2)))$.

– $a \neq a_y$ for all $y \in V$. Then $R(a(\sigma(x_1) + \sigma(x_2))) = a(R(\sigma(x_1)) + R(\sigma(x_2))) \preccurlyeq aR(\sigma(x_1)) + aR(\sigma(x_2)) = R(a\sigma(x_1) + a\sigma(x_2))$.

We now turn to IF2. We distinguish two cases as well.

– $a = a_y$ for some $y \in V$. Then $R(a_y(\sigma(x_1) + \sigma(x_2)) + a_y\sigma(x_1) + a_y(\sigma(x_2) + \sigma(x_3))) = y + y + y \approx y + y = R(a_y\sigma(x_1) + a_y(\sigma(x_2) + \sigma(x_3)))$.

– $a \neq a_y$ for all $y \in V$. Then $R(a(\sigma(x_1) + \sigma(x_2)) + a\sigma(x_1) + a(\sigma(x_2) + \sigma(x_3))) = a(R(\sigma(x_1)) + R(\sigma(x_2))) + aR(\sigma(x_1)) + a(R(\sigma(x_2)) + R(\sigma(x_3)))$ $\approx$ $aR(\sigma(x_1)) + a(R(\sigma(x_2)) + R(\sigma(x_3)))$ $=$ $R(a\sigma(x_1) + a(\sigma(x_2) + \sigma(x_3)))$.

(3) Consider the operator $\_ + \_$. From $R(p_1) \preccurlyeq R(q_1)$ and $R(p_2) \preccurlyeq R(q_2)$ we derive $R(p_1 + p_2) = R(p_1) + R(p_2) \preccurlyeq R(q_1) + R(q_2) = R(q_1 + q_2)$.

Consider the prefix operator $a\_$. We distinguish two cases.

– $a = a_y$ for some $y \in V$. Then $R(a_y p_1) = y = R(a_y q_1)$.

– $a \neq a_y$ for all $y \in V$. Then from $R(p_1) \preccurlyeq R(q_1)$ we derive $R(ap_1) = aR(p_1) \preccurlyeq aR(q_1) = R(aq_1)$.

The proof is now complete. ■

This result, together with the link established in Section 3.3, yields

Corollary 5.3.2 If $|A| = \infty$, then A1-4+W1+W2′+W3 is ω-complete .

5.3.2 Finite Alphabet

$1 < |A| < \infty$. We prove that the inequational theory of BCCS(A) modulo $\preceq_{\mathrm{WIF}}$ does *not* have a finite basis in case of a finite alphabet with at least two elements. The cornerstone for this negative result is the following infinite family of inequations, for $m \geq 0$:

$$\tau(a^m x) + \Phi_m \;\preccurlyeq\; \Phi_m$$

with

$$\Phi_m \;=\; \tau(a^m x + x) + \sum_{b \in A} \tau(a^m x + a^m b\mathbf{0}) \;.$$

It is not hard to see that these inequations are sound modulo $\preceq_{\mathrm{WIF}}$. Namely, given a closed substitution ρ, we have $\mathcal{WT}(\rho(\tau(a^m x))) \subseteq \mathcal{WT}(\rho(\Phi_m))$ and $\rho(\Phi_m) \xrightarrow{\tau}$. To argue that $\rho(\tau(a^m x) + \Phi_m)$ and $\rho(\Phi_m)$ have the same weak impossible futures, we only need to consider the transition $\rho(\tau(a^m x) + \Phi_m) \xrightarrow{\tau} a^m\rho(x)$ (all other cases being trivial). If $\rho(x) = \mathbf{0}$, then $\rho(\Phi_m) \xrightarrow{\tau} a^m\mathbf{0} + \mathbf{0}$ generates the same weak impossible futures (ε, X). If, on the other hand, $b \in \mathcal{I}(\rho(x))$ for some $b \in A$, then $\mathcal{WT}(a^m\rho(x) + a^m b\mathbf{0}) = \mathcal{WT}(a^m\rho(x))$, so $\rho(\Phi_m) \xrightarrow{\tau} a^m\rho(x) + a^m b\mathbf{0}$ generates the same weak impossible futures (ε, X).

We have already defined the traces and completed traces of closed terms. Now we extend these definitions to open terms by allowing weak (completed) traces of the form $a_1 \cdots a_k x \in A^*V$. We do this by treating each variable occurrence x in a term as if it were a subterm $x\mathbf{0}$ with x a concrete action, and then apply Def. 2.1.6. Under this convention, $\mathcal{WCT}(\Phi_m) = \{a^m x, x, a^m b \mid b \in A\}$. We write $\mathcal{WT}_V(t)$ for the set of weak traces of t that end in a variable, and $\mathcal{WT}_A(t)$ for ones that end in an action.

Observation 5.3.3 Let $m > \mathit{depth}(t)$ or $a_m \in V$. Then $a_1 \cdots a_m \in \mathcal{WT}(\sigma(t))$ iff there is a $k < m$ and $y \in V$ such that $a_1 \cdots a_k y \in \mathcal{WT}_V(t)$ and $a_{k+1} \cdots a_m \in \mathcal{WT}(\sigma(y))$.

Lemma 5.3.4 If $|A| > 1$ and $t \preceq_{\mathrm{WIF}} u$, then $\mathcal{WT}_A(t) = \mathcal{WT}_A(u)$ and $\mathcal{WT}_V(t) = \mathcal{WT}_V(u)$.

Proof: Let σ be the closed substitution defined by $\sigma(x) = \mathbf{0}$ for all $x \in V$. Then $t \preceq_{\mathrm{WIF}} u$ implies $\sigma(t) \preceq_{\mathrm{WIF}} \sigma(u)$ and hence $\mathcal{WT}_A(t) = \mathcal{WT}(\sigma(t)) = \mathcal{WT}(\sigma(u)) = \mathcal{WT}_A(u)$ by Def. 2.1.6.

For the second statement fix distinct actions $a, b \in A$ and an injection $\ulcorner \cdot \urcorner : V \to \mathbb{Z}_{>0}$ (which exists because V is countable). Let $m = \mathit{depth}(u) + 1 = \mathit{depth}(t) + 1$. Define the closed substitution ρ by $\rho(z) = a^{\ulcorner z \urcorner \cdot m} b\mathbf{0}$ for all $z \in V$. Again, by Def. 2.1.6, $t \preceq_{\mathrm{WIF}} u$ implies $\mathcal{WT}(\rho(t)) = \mathcal{WT}(\rho(u))$. By Obs. 5.3.3, for all terms v we have $a_1 \cdots a_k y \in \mathcal{WT}_V(v)$ iff $a_1 \cdots a_k a^{\ulcorner y \urcorner \cdot m} b \in \mathcal{WT}(\rho(v))$ with $k < m$. Hence $\mathcal{WT}_V(v)$ is completely determined by $\mathcal{WT}(\rho(v))$ and thus $\mathcal{WT}_V(t) = \mathcal{WT}_V(u)$. ■

Lemma 5.3.5 Let $|A| > 1$. Suppose $t \preceq_{\mathrm{WIF}} u$ and $t \Rightarrow \xrightarrow{\tau} t'$. Then there is a term u' such that $u \Rightarrow \xrightarrow{\tau} u'$ and $\mathcal{WT}_V(u') \subseteq \mathcal{WT}_V(t')$.

Proof: Define ρ exactly as in the previous proof. Since $\rho(t) \Rightarrow \rho(t')$ and $t \preceq_{\mathrm{WIF}} u$ there must be a u' with $\rho(u) \Rightarrow q$ and $\mathcal{WT}(q) \subseteq \mathcal{WT}(\rho(t'))$. Since $\rho(x)$ is τ-free for $x \in V$ it must be that $q = \rho(u')$ for some term u' with $u \Rightarrow u'$. Given the relationship between $\mathcal{WT}_V(v)$ and $\mathcal{WT}(\rho(v))$ for terms v observed in the previous proof, it follows that $\mathcal{WT}_V(u') \subseteq \mathcal{WT}_V(t')$. In case $u \Rightarrow \xrightarrow{\tau} u'$ we are done, so assume $u' = u$. Let σ be the substitution with $\sigma(x) = \mathbf{0}$ for all $x \in V$. Since $\sigma(t) \xrightarrow{\tau}$ and $t \preceq_{\mathrm{WIF}} u$ we have $\sigma(u) \xrightarrow{\tau}$, so $u \xrightarrow{\tau} u''$ for some u''. Now $\mathcal{WT}_V(u'') \subseteq \mathcal{WT}_V(u) = \mathcal{WT}_V(u') \subseteq \mathcal{WT}_V(t')$. ■

Lemma 5.3.6 Let $|A| > 1$. Assume that, for some terms t, u, substitution σ, $a \in A$ and integer m:

1. $t \preceq_{\mathrm{WIF}} u$;
2. $m \geq \mathit{depth}(u)$; and
3. $\sigma(t) \Rightarrow \xrightarrow{\tau} \hat{t}$ for a term $\hat{t}$ without traces ax for $x \in V$ or $a^m b$ for $b \in A$.

Then $\sigma(u) \Rightarrow\!\!\xrightarrow{\tau} \hat{u}$ for a term $\hat{u}$ without traces ax for $x \in V$ or $a^m b$ for $b \in A$.

Proof: Based on proviso (3) there are two cases to consider.

- $y \in \mathcal{WT}_V(t)$ for some $y \in V$ and $\sigma(y) \Rightarrow\!\!\xrightarrow{\tau} \hat{t}$. In that case $y \in \mathcal{WT}_V(u)$ by Lem. 5.3.4, so $\sigma(u) \Rightarrow\!\!\xrightarrow{\tau} \hat{t}$.
- $t \Rightarrow\!\!\xrightarrow{\tau} t'$ for some term t' such that $\hat{t} = \sigma(t)$. By Lem. 5.3.5 there is a term u' with $u \Rightarrow\!\!\xrightarrow{\tau} u'$ and $\mathcal{WT}_V(u') \subseteq \mathcal{WT}_V(t')$. Take $\hat{u} = \sigma(u')$. Clearly $\sigma(u) \Rightarrow\!\!\xrightarrow{\tau} \sigma(u')$. Suppose $\sigma(u')$ would have a weak trace $a^m b$. Then, by Obs. 5.3.3, there is a $k \leq m$ and $y \in V$ such that $a^k y \in \mathcal{WT}_V(u')$ and $a^{m-k} b \in \mathcal{WT}(\sigma(y))$. Since $\mathcal{WT}_V(u') \subseteq \mathcal{WT}_V(t')$ we have $a^m b \in \mathcal{WT}(\sigma(t'))$, which is a contradiction. The case $ax \in \mathcal{WT}(\sigma(u))$ is dealt with in the same way.

The proof is now complete. ■

Lemma 5.3.7 Let $|A| > 1$ and let E be an axiomatization sound for $\preceq_{\mathrm{WIF}}$. Assume that, for some terms v, w, action a and integer m:

1. $E \vdash v \preccurlyeq w$;
2. $m \geq \max\{\mathit{depth}(u) \mid t \preccurlyeq u \in E\}$; and
3. $v \Rightarrow\!\!\xrightarrow{\tau} \hat{v}$ for a term $\hat{v}$ without traces ax for $x \in V$ or $a^m b$ for $b \in A$.

Then $w \Rightarrow\!\!\xrightarrow{\tau} \hat{w}$ for a term $\hat{w}$ without traces ax for $x \in V$ or $a^m b$ for $b \in A$.

Proof: By induction on the derivation of $E \vdash v \preccurlyeq w$.

- Suppose $E \vdash v \preccurlyeq w$ because $\sigma(t) = v$ and $\sigma(u) = w$ for some $t \preccurlyeq u \in E$ and substitution σ. The claim then follows by Lem. 5.3.6.
- Suppose $E \vdash v \preccurlyeq w$ because $E \vdash v \preccurlyeq u$ and $E \vdash u \preccurlyeq w$ for some u. By induction, $u \Rightarrow\!\!\xrightarrow{\tau} \hat{u}$ for a term $\hat{u}$ without traces ax or $a^m b$. Hence, again by induction, $w \Rightarrow\!\!\xrightarrow{\tau} \hat{w}$ for a term $\hat{w}$ without traces ax or $a^m b$.
- Suppose $E \vdash v \preccurlyeq w$ because $v = v_1 + v_2$ and $w = w_1 + w_2$ with $E \vdash v_1 \preccurlyeq w_1$ and $E \vdash v_2 \preccurlyeq w_2$. Since $v \Rightarrow\!\!\xrightarrow{\tau} \hat{v}$, either $v_1 \Rightarrow\!\!\xrightarrow{\tau} \hat{v}$ or $v_2 \Rightarrow\!\!\xrightarrow{\tau} \hat{v}$. Assume, without loss of generality, that $v_1 \Rightarrow\!\!\xrightarrow{\tau} \hat{v}$. By induction, $w_1 \Rightarrow\!\!\xrightarrow{\tau} \hat{w}$ for a term $\hat{w}$ without traces ax or $a^m b$. Now $w \Rightarrow\!\!\xrightarrow{\tau} \hat{w}$.
- Suppose $E \vdash v \preccurlyeq w$ because $v = cv_1$ and $w = cw_1$ with $c \in A$ and $E \vdash v_1 \approx w_1$. In this case, proviso (3) of the lemma can not be met.
- Suppose $E \vdash v \preccurlyeq w$ because $v = \tau v_1$ and $w = \tau w_1$ with $E \vdash v_1 \approx w_1$. Then either $v_1 = \hat{v}$ or $v_1 \Rightarrow\!\!\xrightarrow{\tau} \hat{v}$. In the first case, w_1 has no traces ax or $a^m b$ by Lem. 5.3.4 and proviso (3) of the lemma; hence w has no such traces either. In the second case, by induction, $w_1 \Rightarrow\!\!\xrightarrow{\tau} \hat{w}$ for a term $\hat{w}$ without traces ax or $a^m b$. Again $w \Rightarrow\!\!\xrightarrow{\tau} \hat{w}$.

The proof is now complete. ■

Theorem 5.3.8 If $1 < |A| < \infty$, then the inequational theory of BCCS(A) modulo $\preceq_{\mathrm{WIF}}$ does *not* have a finite basis.

Proof: Let E be a finite axiomatization over BCCS(A) that is sound modulo $\preceq_{\mathrm{WIF}}$. Let m be greater than the depth of any term in E. According to Lem. 5.3.7, the inequation $\tau(a^m x) + \Phi_m \preccurlyeq \Phi_m$ cannot be derived from E. Yet it is sound modulo $\preceq_{\mathrm{WIF}}$. ■

$|A| = 1$. We prove that the inequational theory of BCCS(A) modulo $\preceq_{\mathrm{WIF}}$ does *not* have a finite basis in case of a singleton alphabet. The cornerstone for this negative result is the following infinite family of inequations, for $m \geq 0$:

$$a^m x \;\preccurlyeq\; a^m x + x$$

If $|A| = 1$, then these inequations are clearly sound modulo $\preceq_{\mathrm{WIF}}$. Note that given a closed substitution ρ, $\mathcal{WT}(\rho(x)) \subseteq \mathcal{WT}(\rho(a^m x))$.

Lemma 5.3.9 If $t \preceq_{\mathrm{WIF}} u$ then $\mathcal{WT}_V(t) \subseteq \mathcal{WT}_V(u)$.

Proof: Fix $a \in A$ and an injection $\ulcorner\cdot\urcorner : V \to \mathbb{Z}_{>0}$. Let $m = \mathit{depth}(u) + 1$. Define the closed substitution ρ by $\rho(z) = a^{\ulcorner z \urcorner \cdot m}\mathbf{0}$ for all $z \in V$. By Lem. 5.2.10, $\mathcal{WCT}(\rho(t)) \subseteq \mathcal{WCT}(\rho(u))$. Now suppose $a_1 \cdots a_k y \in \mathcal{WT}_V(t)$. Then $a_1 \cdots a_k a^{\ulcorner y \urcorner \cdot m} \in \mathcal{WCT}(\rho(t)) \subseteq \mathcal{WCT}(\rho(u))$ and $k < m$. This is only possible if $a_1 \cdots a_k y \in \mathcal{WT}_V(u)$. ■

Lemma 5.3.10 Assume that, for terms t, u, substitution σ, $a \in A$, $x \in V$, integer m:

1. $t \preceq_{\mathrm{WIF}} u$;
2. $m > \mathit{depth}(u)$; and
3. $x \in \mathcal{WT}_V(\sigma(u))$ and $a^k x \notin \mathcal{WT}_V(\sigma(u))$ for $1 \leq k < m$.

Then $x \in \mathcal{WT}_V(\sigma(t))$ and $a^k x \notin \mathcal{WT}_V(\sigma(t))$ for $1 \leq k < m$.

Proof: Since $x \in \mathcal{WT}_V(\sigma(u))$, by Obs. 5.3.3 there is a variable y with $y \in \mathcal{WT}_V(u)$ and $x \in \mathcal{WT}_V(\sigma(y))$. Consider the closed substitution ρ given by $\rho(y) = a^m\mathbf{0}$ and $\rho(z) = \mathbf{0}$ for $z \neq y$. Then $m > \mathit{depth}(u) = \mathit{depth}(t)$, and $y \in \mathcal{WT}_V(u)$ implies $a^m \in \mathcal{WT}(\rho(u)) = \mathcal{WT}(\rho(t))$, so by Obs. 5.3.3 there is some $k < m$ and $z \in V$ such that $a^k z \in \mathcal{WT}_V(t)$ and $a^{m-k} \in \mathcal{WT}(\rho(z))$. As $k < m$ it must be that $z = y$. Since $a^k y \in \mathcal{WT}_V(t)$ and $x \in \mathcal{WT}_V(\sigma(y))$, Obs. 5.3.3 implies that $a^k x \in \mathcal{WT}_V(\sigma(t))$. By Lem. 5.3.9, $a^k x \notin \mathcal{WT}_V(\sigma(t))$ for $1 \leq k < m$. Hence we obtain $k = 0$. ■

Lemma 5.3.11 Assume that, for E an axiomatization sound for $\preceq_{\mathrm{WIF}}$ and for terms v, w, $a \in A$, $x \in V$ and integer m:

1. $E \vdash v \preccurlyeq w$;
2. $m > \max\{depth(u) \mid t \preccurlyeq u \in E\}$; and
3. $x \in \mathcal{WT}_V(w)$ and $a^k x \notin \mathcal{WT}_V(w)$ for $1 \leq k < m$.

Then $x \in \mathcal{WT}_V(v)$ and $a^k x \notin \mathcal{WT}_V(v)$ for $1 \leq k < m$.

Proof: By induction on the derivation of $E \vdash v \preccurlyeq w$.

- Suppose $E \vdash v \preccurlyeq w$ because $\sigma(t) = v$ and $\sigma(u) = w$ for some $t \preccurlyeq u \in E$ and substitution σ. The claim then follows by Lem. 5.3.10.
- Suppose $E \vdash v \preccurlyeq w$ because $E \vdash v \preccurlyeq u$ and $E \vdash u \preccurlyeq w$ for some u. By induction, $x \in \mathcal{WT}_V(u)$ and $a^k x \notin \mathcal{WT}_V(u)$ for $1 \leq k < m$. Hence, again by induction, $x \in \mathcal{WT}_V(v)$ and $a^k x \notin \mathcal{WT}_V(v)$ for $1 \leq k < m$.
- Suppose $E \vdash v \preccurlyeq w$ because $v = v_1 + v_2$ and $w = w_1 + w_2$ with $E \vdash v_1 \preccurlyeq w_1$ and $E \vdash v_2 \preccurlyeq w_2$. Since $x \in \mathcal{WT}_V(w)$, either $x \in \mathcal{WT}_V(w_1)$ or $x \in \mathcal{WT}_V(w_2)$. Assume, without loss of generality, that $x \in \mathcal{WT}_V(w_1)$. Since $a^k x \notin \mathcal{WT}_V(w)$ for $1 \leq k < m$, surely $a^k x \notin \mathcal{WT}_V(w_1)$ for $1 \leq k < m$. By induction, $x \in \mathcal{WT}_V(v_1)$, and hence $x \in \mathcal{WT}_V(v)$. For $1 \leq k < m$ we have $a^k x \notin \mathcal{WT}_V(w)$ and hence $a^k x \notin \mathcal{WT}_V(v)$, by Lem. 5.3.9.
- Suppose $E \vdash v \preccurlyeq w$ because $v = cv_1$ and $w = cw_1$ with $c \in A$ and $E \vdash v_1 \approx w_1$. In this case, proviso (3) of the lemma can not be met.
- Suppose $E \vdash v \preccurlyeq w$ because $v = \tau v_1$ and $w = \tau w_1$ with $E \vdash v_1 \approx w_1$. Then, by proviso (3) of the lemma, $x \in \mathcal{WT}_V(w_1)$ and $a^k x \notin \mathcal{WT}_V(w_1)$ for $1 \leq k < m$. By induction, $x \in \mathcal{WT}_V(v_1)$ and $a^k x \notin \mathcal{WT}_V(v_1)$ for $1 \leq k < m$. Hence $x \in \mathcal{WT}_V(v)$ and $a^k x \notin \mathcal{WT}_V(v)$ for $1 \leq k < m$.

The proof is now complete. ■

Theorem 5.3.12 If $|A| = 1$, then the inequational theory of BCCS(A) modulo $\preceq_{\mathrm{WIF}}$ does *not* have a finite basis.

Proof: Let E be a finite axiomatization over BCCS(A) that is sound modulo $\preceq_{\mathrm{WIF}}$. Let m be greater than the depth of any term in E. According to Lem. 5.3.11, the inequation $a^m x \preccurlyeq a^m x + x$ cannot be derived from E. Yet, since $|A| = 1$, it is sound modulo $\preceq_{\mathrm{WIF}}$. ■

To conclude this subsection, Thm. 5.3.8 and Thm. 5.3.12 yield

Theorem 5.3.13 If $|A| < \infty$, then the inequational theory of BCCS(A) modulo $\preceq_{\mathrm{WIF}}$ does *not* have a finite basis.

Again, this result, together with the link established in Section 3.3, yields

Corollary 5.3.14 If $|A| < \infty$, then the inequational theory of BCCSP(A) modulo $\precsim_{\mathrm{IF}}$ does *not* have a finite basis.

5.4 n-Nested Impossible Futures

Similar to the n-nested semantics and n-nested possible futures semantics (see, e.g., [AFvGI04]), one can define n-nested impossible futures semantics.[2]

Definition 5.4.1 Assume a labeled transition system. For each $n \geq 0$, the *n-nested impossible futures preorder* $\precsim_n$ on states is defined by:

- $s_1 \precsim_0 s_2$ for any states s_1, s_2;
- $s_1 \precsim_{n+1} s_2$ if $s_1 \xrightarrow{a_1 \cdots a_k} s_1'$ implies $s_2 \xrightarrow{a_1 \cdots a_k} s_2'$ with $s_2' \precsim_n s_1'$.

We write $\simeq_n$ for $\precsim_n \cap \succsim_n$.

$\precsim_{n+1} \subset \simeq_n \subset \precsim_n$ for $n \geq 1$. Moreover, $\precsim_1$ coincides with trace preorder, while $\precsim_2 = \precsim_{\mathrm{IF}}$. Moreover, it is easy to see that the limit of n-nested impossible futures preorder or equivalence coincides with the concrete bisimulation. We will argue that apart from $\precsim_{\mathrm{IF}}$, no nested impossible futures semantics allows a finite, ground-complete axiomatization.

In the proof of this result, which basically consists of a generalization of the proofs of Lem. 6.4.2, Lem. 5.2.13 and Corollary 5.2.16, we shall make use of formulae in the modal characterization of the n-nested impossible futures preorders. A state s satisfies the modal formula $\langle a \rangle \varphi$ if there exists a transition $s \xrightarrow{a} s'$ where s' satisfies the modal formula φ.

Definition 5.4.2 For $n \geq 0$, we define a set $\mathcal{L}_n$ of modal formulae:

- $\mathcal{L}_0$ contains only $\top$ and $\bot$;
- $\mathcal{L}_{n+1}$ is given by the BNF

$$\varphi ::= \langle a_1 \rangle \cdots \langle a_k \rangle \neg \varphi' \quad (a_1 \cdots a_k \in A^*,\ \varphi' \in \mathcal{L}_n).$$

Lemma 5.4.3 Let $n \geq 0$. If $s_1 \precsim_n s_2$, then $\forall \varphi \in \mathcal{L}_n$: $s_1 \models \varphi \ \Rightarrow\ s_2 \models \varphi$.

Proof: By induction on n. The base case is trivial. Suppose $s_1 \precsim_{n+1} s_2$, and let $s_1 \models \varphi \in \mathcal{L}_{n+1}$, where $\varphi = \langle a_1 \rangle \cdots \langle a_k \rangle \neg \varphi$ with $\varphi \in \mathcal{L}_n$. Then $s_1 \xrightarrow{a_1 \cdots a_k} s_1'$ with $s_1' \not\models \varphi$. Since $s_1 \precsim_{n+1} s_2$, $s_2 \xrightarrow{a_1 \cdots a_k} s_2'$ with $s_2' \precsim_n s_1'$. By the induction hypothesis, $s_2' \not\models \varphi$. Then $s_2' \models \neg\varphi$, and thus $q \models \varphi$. ■

The operator $\_ ;_m a^\ell$ adds a sequence of ℓ a-transitions to every state at depth m from which no transition is available.

[2] Here we only deal with *concrete* version; And we omit the subscript "IF" in $\precsim_{\mathrm{IF}}$ since it is clear from the context.

Definition 5.4.4 [AFvGI04, Def. 31] For $k, \ell \geq 0$, define the operator $\_;_k a^\ell$ on closed terms inductively by

$$\begin{array}{rcl} (\sum_{i=1}^m a_i p_i);_{k+1} a^\ell & = & \Sigma_{i=1}^m a_i (p_i;_k a^\ell) \\ (bp+q);_0 a^\ell & = & bp + q \\ \mathbf{0};_0 a^\ell & = & a^\ell \mathbf{0} \ . \end{array}$$

In the remainder of this section, we assume without loss of generality that $A = \{a\}$. This is justified because in the coming proofs we will only consider inequations $t \preccurlyeq u$ and equations $t \approx u$ where no actions $b \neq a$ occur in t and u; and it is easy to see that any sound derivation of an such an (in)equation cannot contain an occurrence of an action $b \neq a$.

For $n \geq 1$ and $m \geq 0$, we define formulae φ_n^m:

$$\begin{array}{rcl} \varphi_1^m & = & \langle a \rangle^m \neg \langle a \rangle \top \\ \varphi_{n+1}^m & = & \langle a \rangle \neg \varphi_n^m \ . \end{array}$$

In other words, $\varphi_n^m = (\langle a \rangle \neg)^{n-1} \langle a \rangle^m \neg \langle a \rangle \top$. By induction on n, it is easy to see that $\varphi_n^m \in \mathcal{L}_{n+1}$.

We now formulate a slight variation of [AFvGI04, Lem. 36].

Lemma 5.4.5 Let t be a term with $\mathit{depth}(t) < m$ and $\mathit{depth}(\rho(t)) < 2m + n$, for some $m, n \geq 1$. Let ρ be a closed substitution with $\rho(y) = \mathbf{0}$ for each variable y that occurs at multiple depths in t. Let ρ' be a closed substitution with $\rho'(x) = \rho(x);_{m+n-1-d_x} a^{m+1}\mathbf{0}$ if $\rho(x) \neq \mathbf{0}$ and $x \in \mathit{var}_{d_x}(t)$, and $\rho'(x) = \mathbf{0}$ if $\rho(x) = \mathbf{0}$. Then

$$\rho(t) \models \varphi_n^m \ \Leftrightarrow \ \rho'(t) \models (\langle a \rangle \neg)^{n-1} \langle a \rangle^{2m+1} \top \ .$$

Proof: (Sketch) By induction on m, we can show

$$\rho'(t) = \rho(t);_{m+n-1} a^{m+1} \ .$$

And by induction on $m + n$, we can show

$$\rho(t) \models \varphi_n^m \ \Leftrightarrow \ \rho(t);_{m+n-1} a^{m+1} \models (\langle a \rangle \neg)^{n-1} \langle a \rangle^{2m+1} \top \ .$$

(The latter proof uses that $A = \{a\}$.) ■

Lemma 5.4.6 Let $n \geq 1$. Assume that, for some terms t, u and closed substitution ρ:

1. $t \precsim_n u$;
2. $m > \mathit{depth}(u)$;
3. $\mathcal{WCT}(\rho(u)) \subseteq \{a^{m+n-1}, a^{2m+n-1}\}$; and

4. $\rho(t) \models \varphi_n^m$.

Then $\rho(u) \models \varphi_n^m$.

Proof: From provisos (2) and (3), it is not hard to see that $\rho(y) = \mathbf{0}$ for each variable y that occurs at multiple depths in u. So by Lem. 4.2.1, the same holds for t. Let ρ' be defined as in Lem. 5.4.5. By proviso (4), $\rho(t) \models \varphi_n^m$, so by Lem. 5.4.5, $\rho'(t) \models (\langle a\rangle\neg)^{n-1}\langle a\rangle^{2m+1}\top$. Note that $(\langle a\rangle\neg)^{n-1}\langle a\rangle^{2m+1}\top \in \mathcal{L}_n$. By proviso (1), $\rho'(t) \precsim_n \rho'(u)$, so by Lem. 5.4.3, $\rho'(u) \models (\langle a\rangle\neg)^{n-1}\langle a\rangle^{2m+1}\top$. Hence, again by Lem. 5.4.5, $\rho(u) \models \varphi_n^m$. ■

Lemma 5.4.7 Let $n \geq 2$. Let the finite axiomatization E be sound modulo $\simeq_n$. Assume that, for some closed terms p, q:

1. $E \vdash p \approx q$;
2. $m > \mathit{depth}(q)$;
3. $\mathcal{WCT}(q) \subseteq \{a^{m+n-1}, a^{2m+n-1}\}$; and
4. $p \models \varphi_n^m$.

Then $q \models \varphi_n^m$.

Proof: By induction on the derivation of $E \vdash p \approx q$.

The case $\rho(t) = p$ and $\rho(u) = q$ for some $t \approx u \in E$ and closed substitution ρ, follows from Lem. 5.4.6.

The other three cases ((1) $E \vdash p \approx r$ and $E \vdash r \approx q$; (2) $p = p_1 + p_2$ and $q = q_1 + q_2$ with $E \vdash p_1 \approx q_1$ and $E \vdash p_2 \approx q_2$; (3) $p = ap'$ and $q = aq'$ with $E \vdash p' \approx q'$) can be dealt with in the same way as in the proof of Lem. 5.2.13. ■

Theorem 5.4.8 Let $n \geq 2$. There is no finite, sound, ground-complete axiomatization for BCCSP(A) modulo $\simeq_n$.

Proof: Let E be a finite axiomatization that is sound modulo $\simeq_n$. Let m be greater than the depth of any term in E. For $k \geq 0$, we define closed terms p_k^m and q_k^m:

$$\begin{array}{lcllcl} p_0^m & = & a^{2m-1}\mathbf{0} & q_0^m & = & a^{m-1}\mathbf{0} \\ p_{k+1}^m & = & ap_k + aq_k & q_{k+1}^m & = & ap_k \ . \end{array}$$

Clearly, $q_k \precsim_{k+1} p_k$. This induces that $p_k^m \simeq_k q_k^m$.

It is not hard to see that $p_k^m \models \varphi_k^m$ while $q_k^m \not\models \varphi_k^m$ (for $k \geq 1$). So by Lem. 5.4.7, $p_n^m \approx q_n^m$ cannot be derived from E. Hence, E is not ground-complete. ■

Likewise we can prove this inaxiomatizability result for $\precsim_n$ in case $n \geq 3$. The reason that the proof can be shifted from equivalences to preorders without

problem, is that the key result Lem. 5.4.6 is formulated for $\precsim_n$. The reason that the proof does not extend to $\precsim_2$ is that $\precsim_2 \not\subseteq \simeq_{\mathrm{CT}}$, while this inclusion is essential in the proof of Lem. 5.4.7 (see also the proof of Lem. 5.2.13). On the other hand, $\precsim_3 \subseteq \simeq_{\mathrm{CT}}$ does hold (see Lem. 5.2.10).

Theorem 5.4.9 Let $n \geq 3$. There is no finite, sound, ground-complete axiomatization for BCCSP(A) modulo $\precsim_n$.

5.5 Conclusion

We have performed a comprehensive and systematic study on the axiomatizability of concrete and weak impossible futures semantics over process algebra BCCSP and BCCS. Tab. 5.2 presents an overview, (as in Tab. 4.1) with a $+$ indicating that a finite basis exists and a $-$ indicating that a finite basis does not exist. The table expands in two dimensions: ground-completeness v.s. ω-completeness and preorder v.s. equivalence. When necessarily, we distinguish two categories, according to the cardinality of the alphabet A: finite or infinite.

	ground-comp.	ω-comp.	
	$1 \leq \lvert A\rvert \leq \infty$	$\lvert A\rvert = \infty$	$1 \leq \lvert A\rvert < \infty$
(concrete/weak) preorder	$+$	$+$	$-$
(concrete/weak) equivalence	$-$	$-$	$-$
$n(\geq 3)$-nested preorder	$-$	$-$	$-$
$n(\geq 2)$-nested equivalence	$-$	$-$	$-$

Table 5.2: Summary of the axiomatizability of impossible futures

Chapter 6

Priority

6.1 Introduction

Programming and specification languages often include constructs to describe mode switches (see, e.g., [Mau91, MTHM97]). Indeed, some form of mode transfer in computation appears in operating systems in the guise of interrupts, in programming languages as exceptions, and in the behavior of control programs and embedded systems as discrete "mode switches" triggered by changes in the state of their environment. Such mode changes are often used to encode different levels of urgency amongst the actions that can be performed by a system as it computes, and implement variations on the notion of pre-emption.

In light of the ubiquitous nature of mode changes in computation, it is not surprising that classic process description languages include primitive operators to describe mode changes – for example, LOTOS [Bri85, ISO87] offers the so-called disruption operator – or have been extended with variations on mode transfer operators. Examples of such operators that may be added to the process algebra CCS are discussed by Milner in [Mil89a, page 192–193], and Dsouza and Bloom offer in [DB95] some discussion on the benefits of adding one of those, viz. the checkpointing operator, to CCS.

One of the most widely studied, and natural, notions used to implement different levels of urgency between system actions is priority. (A thorough and clear discussion of the different approaches to the study of priority in process description languages may be found in [CLN01].) In this chapter, we consider the well-known priority operator Θ studied by Baeten, Bergstra and Klop [BBK86] in the context of process algebra. (See [CH90, CW95, CLNS96, CLN07, Phi08] for later accounts of the priority operator in the setting of process description languages.) The priority operator Θ gives certain actions priority over others based on an irreflexive partial ordering relation $<$ over the set of actions. Intuitively, $a < b$ is interpreted as "b has priority over a". This means that, in the context of the priority operator Θ, action a is pre-empted by action b. For example, if p is some process that can initially perform both a and b, then $\Theta(p)$ will initially only be able to execute the action b.

In their classic paper [BBK86], Baeten, Bergstra and Klop provided a sound

and ground-complete axiomatization for this operator modulo bisimulation equivalence. Their axiomatization uses predicates on actions (to express priorities between actions) and one extra auxiliary operator. Bergstra showed in the earlier paper [Ber85] that, in case of a finite alphabet of actions, there exists a finite, ground-complete axiomatization for Θ, without action predicates and help operators. So, if the set of actions is finite, neither equations with action predicates as conditions nor auxiliary operators, as used in [BBK86], are actually necessary to obtain a finite axiomatization of bisimulation equivalence over basic process description languages enriched with the priority operator. However, can Bergstra's positive result be extended to a setting with a countably infinite collection of actions? Or are equations with action predicates as conditions and auxiliary operators necessary to obtain a finite axiomatization of bisimulation equivalence in the presence of an infinite collection of actions? (Note that infinite sets of actions are common in process calculi, and arise, for instance, in the setting of value- or name-passing calculi, e.g. value-passing CCS [Mil89a], μCRL [GR01] and π-calculus [MPW92a, MPW92b].) The aim of this chapter is to provide a thorough answer to these questions in the setting of the process algebra BCCSP enriched with the priority operator Θ. In case of an infinite alphabet, we permit the occurrence of action variables in axioms.

This chapter considers the equational theory of BCCSP with the priority operator Θ from [BBK86] modulo bisimulation equivalence. Our first main result is a theorem indicating that the use of *equations with action predicates as conditions* is indeed inevitable in order to offer a finite, ground-complete axiomatization of bisimulation equivalence over the basic process language we consider in this study. To this end, we prove that, in case of an infinite alphabet and in the presence of at least one priority relation $a < b$ between a pair of actions, there is no finite, ground-complete axiomatization for BCCSP enriched with the priority operator (Thm. 6.4.3). This result even applies if one is allowed to add an arbitrary collection of help operators to the syntax. Thm. 6.4.3 offers a very strong indication that the use of equations with action predicates as conditions is essential for axiomatizing Θ, and cannot be circumvented by introducing auxiliary operators. (This is in contrast to the classic positive and negative results on the existence of finite, ground-complete axiomatizations for parallel composition offered in [BK84, Mol90a, Mol90b].)

The idea underlying the proof of Thm. 6.4.3 is that for each finite sound axiomatization E there is a pair of actions c, d that does not occur in E. If c and d are incomparable, then

$$\Theta(c\mathbf{0} + d\mathbf{0}) \approx c\mathbf{0} + d\mathbf{0}$$

is sound modulo bisimulation equivalence. However, using a simple renaming argument, we show that a derivation of this equation from E would give rise to a derivation of the unsound equation $\Theta(a\mathbf{0} + b\mathbf{0}) \approx a\mathbf{0} + b\mathbf{0}$. Likewise, if $c < d$, then

$$\Theta(c\mathbf{0} + d\mathbf{0}) \approx d\mathbf{0}$$

is sound modulo bisimulation equivalence. But we prove that a derivation of

this equation from E would give rise to a derivation of the unsound equation $\Theta(d\mathbf{0} + c\mathbf{0}) \approx c\mathbf{0}$.

Having established that equations with action predicates as conditions are necessary in order to obtain a finite, ground-complete axiomatization of bisimulation equivalence, we then proceed to investigate whether, in the presence of an infinite set of actions, this equivalence can be finitely axiomatized using equations with action predicates as conditions, but without auxiliary operators like the unless operator used in [BBK86]. We show that, in general, the answer to this question is negative. This we do by exhibiting a priority structure with respect to which bisimulation equivalence affords no finite, sound and ground-complete axiomatization in terms of equations with action predicates as conditions (Thm. 6.5.6). This shows that, in general, the use of auxiliary operators is indeed necessary to axiomatize bisimulation equivalence finitely, even using equations with action predicates as conditions and over the simple language considered in this study. The priority structure used in the proof of Thm. 6.5.6 consists of actions a_i and b_i for $i \geq 1$ together with an action c, where $a_i < b_i < c$ for each $i \geq 1$. We prove that given a finite sound axiomatization E consisting of equations with action predicates as conditions, the sound equation

$$\Theta(b_1\mathbf{0} + \cdots + b_n\mathbf{0}) \approx b_1\mathbf{0} + \cdots + b_n\mathbf{0}$$

cannot be derived from E, for a sufficiently large n.

In contrast to the aforementioned negative results, we exhibit a countably infinite, ground-complete axiomatization for bisimulation equivalence over BCCSP with the priority operator in terms of equations with action predicates as conditions (Thm. 6.5.9). This axiomatization suggests that, in general, infinite collections of pairwise incomparable actions with respect to the priority relation $<$ are the source of our negative result presented in Thm. 6.5.6. It is therefore natural to ask ourselves whether there are conditions that can be imposed on the poset of actions that are sufficient to guarantee that bisimulation equivalence be finitely axiomatizable using equations with action predicates as conditions, but without auxiliary operators. We conclude the technical developments in this chapter by proposing some such sufficient conditions. The most general of these applies to all priority structures such that

1. the collection of the sizes of the finite, maximal anti-chains is finite,
2. there are only finitely many infinite, maximal anti-chains, and
3. for each infinite, maximal anti-chain A, each element of A is above the same set of actions, that is, for each $a, b \in A$ and action c, we have that $c < a$ iff $c < b$.

Our results add the priority operator to the list of operators whose addition to a process algebra spoils finite axiomatizability modulo bisimulation equivalence; see, e.g., [AFIL05, AFIN06, Mol90a, Mol90b, Sew97] for other examples of non-finite axiomatizability results over process algebras. Notably, in [AFIN06] two mode transfer operators from [BB00] are studied in the setting of the basic

process algebra BPA. It is shown that, even in the presence of just one action, the interrupt operator does not have a finite, ground-complete axiomatization, while the disrupt operator does. In the interrupt operator, a process p can be interrupted by another process q; upon termination of q, process p resumes its computation. In the disrupt operator, a process p can be preempted by another process q, after which the execution of p is aborted.

Structure of the chapter. Section 6.2 contains the preliminaries. In Section 6.3, the finite axiomatization for the priority operator Θ from [Ber85] is presented. Section 6.4 contains a proof of a result to the effect that, in case of an infinite alphabet, there is no finite, ground-complete axiomatization for the priority operator modulo bisimulation equivalence, even in the presence of auxiliary operators. Finally, we show that, in the presence of an infinite set of actions, in general bisimulation equivalence does not afford a finite axiomatization in terms of equations with action predicates as conditions without the use of auxiliary operators (Section 6.5.1), and we identify sufficient conditions on the priority structure over actions that lead to the existence of a finite axiomatization using equations with action predicates as conditions (Section 6.5.2). Section 6.6 concludes the chapter.

6.2 Preliminaries

We begin by introducing the basic definitions and results on which the technical developments to follow are based.

6.2.1 The Language BCCSP_Θ

In this chapter, we use Act to denote a non-empty alphabet of atomic actions, with typical elements a, b, c, d, e. Over Act we assume an irreflexive, transitive partial ordering $<$ to express priorities between actions.[1] Intuitively, $a < b$ expresses that the action b has priority over the action a. We say that actions $a_1, \ldots, a_n$ are *incomparable* if they are distinct and $a_i < a_j$ does not hold for all $1 \leq i, j \leq n$.

The language of processes we shall consider in this chapter, henceforth referred to as BCCSP_Θ, is obtained by adding the unary priority operator Θ from [BBK86] to the basic process algebra BCCSP [vG90, vG01]. The language is given by the following grammar:

$$t ::= \mathbf{0} \mid at \mid t + t \mid \Theta(t) \mid x \mid \alpha t \ ,$$

where a ranges over Act, x is a *process variable* and α is an *action variable*. Process and action variables range over given, disjoint countably infinite sets. We use x, y, z to range over the collection of process variables, and α, β as typical action variables[2].

[1] We use Act instead of A in previous chapters to emphasize the partial ordering structure.
[2] N.B. this is in contrast to previous chapters where α, β range over A_τ.

We use t, u, v to range over the collection of *open* process terms $\mathbb{T}(\mathrm{BCCSP}_\Theta)$. As usual, a process term is *closed* if it does not contain any variables, and p, q, r, range over the set of closed terms $\mathrm{T}(\mathrm{BCCSP}_\Theta)$. The *size* of a term is defined as follows: $size(\mathbf{0}) = size(x) = 0$; $size(at) = size(\alpha t) = 1 + size(t)$; $size(t + u) = size(t) + size(u)$; and $size(\Theta(t)) = size(t)$.

Remark 6.2.1 The readers might have already noticed that we consider a slightly extended syntax for BCCSP, in that we allow for the use of prefixing operators of the form $\alpha\_$, where α is an *action variable*. The use of action variables is natural in the presence of infinite sets of actions, and will allow us to formulate stronger versions of the negative results to follow.

A substitution maps each process variable to a process term, and each action variable to an action or action variable. A substitution is *closed* if it maps process variables to closed process terms and action variables to actions. For every term t and substitution σ, the term obtained by replacing occurrences of process variables x and action variables α in t with $\sigma(x)$ and $\sigma(\alpha)$, respectively, is written $\sigma(t)$. Note that $\sigma(t)$ is closed if so is σ. For example, $\sigma(\alpha x) = a\mathbf{0}$ if $\sigma(\alpha) = a$ and $\sigma(x) = \mathbf{0}$.

As usual, the operational semantics of the language BCCSP_Θ is specified by the transitions rules given in Tab. 6.1, where a ranges over Act. This gives

$$\frac{}{ax \xrightarrow{a} x} \qquad \frac{x_1 \xrightarrow{a} y}{x_1 + x_2 \xrightarrow{a} y} \qquad \frac{x_2 \xrightarrow{a} y}{x_1 + x_2 \xrightarrow{a} y} \qquad \frac{x \xrightarrow{a} y \quad x \overset{b}{\not\to} \text{ for } a < b}{\Theta(x) \xrightarrow{a} \Theta(y)}$$

Table 6.1: SOS for BCCSP_Θ

rise to an Act-labeled transition relation, which is the *unique supported model* in the sense of [BIM95]. Intuitively, closed terms in the language BCCSP_Θ represent finite process behaviors, where $\mathbf{0}$, $p + q$ and ap are exactly the same as in BCCSP while the process graph of $\Theta(p)$ is obtained by eliminating all transitions $q \xrightarrow{a} q'$ from the process graph of p for which there is a transition $q \xrightarrow{b} q''$ with $a < b$.

We consider the language BCCSP_Θ modulo bisimulation equivalence.

Definition 6.2.2 A binary symmetric relation $\mathcal{R}$ over $\mathrm{T}(\mathrm{BCCSP}_\Theta)$ is a *bisimulation* if $p\ \mathcal{R}\ q$ together with $p \xrightarrow{a} p'$ imply $q \xrightarrow{a} q'$ for some q' with $p'\ \mathcal{R}\ q'$. We write $p \leftrightarroweq q$ if there is a bisimulation relating p and q. The relation $\leftrightarroweq$ will be referred to as *bisimulation equivalence* or *bisimilarity*.

It is well-known that $\leftrightarroweq$ is an equivalence relation. Moreover, the transition rules are in the GSOS format of [BIM95]. (We mention in passing that recently Aceto and Ingolfsdottir [AI08] show that the priority operator cannot

be expressed using positive rule formats for operational semantics.) Hence, bisimulation equivalence is a congruence with respect to all the operators in the signature of BCCSP_Θ, meaning that $p \leftrightarrows q$ implies $C[p] \leftrightarrows C[q]$ for each BCCSP_Θ-context $C[\cdot]$.

We can therefore consider the algebra of the closed terms in $\mathrm{T}(\mathrm{BCCSP}_\Theta)$ modulo $\leftrightarrows$. In Section 6.4, we shall offer results that apply to any signature Σ which extends that for BCCSP_Θ. To this end, we shall tacitly assume that all of the new operators in Σ also preserve bisimulation equivalence, and are semantically interpreted as operations over finite synchronization trees.

Our order of business in the remainder of this chapter will be to offer a thorough study of the equational theory of the language BCCSP_Θ modulo bisimulation equivalence. We begin our investigation by considering the case in which the set of actions Act is finite in the following section. We then move on to investigate the equational properties of bisimulation equivalence over BCCSP_Θ when the set of actions is infinite (Sections 6.4 and 6.5).

6.3 $|Act| < \infty$

In this section, we assume that the action set is *finite*. The axioms in Tab. 6.2 were put forward by Bergstra in [Ber85]. Note that, in the case of a finite action set, this axiom system is finite, since then the axiom schemas PR2-4 give rise to finitely many equations.

PR1	$\Theta(\mathbf{0})$	$\approx$	$\mathbf{0}$
PR2	$\Theta(ax+ay+z)$	$\approx$	$\Theta(ax+z)+\Theta(ay+z)$
PR3	$\Theta(ax+by+z)$	$\approx$	$\Theta(by+z) \qquad (a<b)$
PR4	$\Theta(a_1x_1+\cdots+a_nx_n)$	$\approx$	$a_1\Theta(x_1)+\cdots+a_n\Theta(x_n)$ $(a_1,\ldots,a_n$ incomparable$)$

Table 6.2: Axiomatization in case of $|Act| < \infty$

Theorem 6.3.1 ([Ber85]) The axiom system consisting of A1-4 in Tab. 2.4 and PR1-4 in Tab. 6.2 is sound and ground-complete for BCCSP_Θ modulo $\leftrightarrows$.

Proof: (Sketch) Since $\leftrightarrows$ is a congruence with respect to BCCSP_Θ, soundness can be checked for each axiom separately. This is an easy exercise.

Next observe that, using PR1-4, one can remove all occurrences of Θ from closed terms. Then ground-completeness follows from the well-known ground-completeness of A1-4 for BCCSP modulo $\leftrightarrows$ (see, e.g., [HM85]). ■

In the remainder of this chapter, as usual, process terms are considered modulo associativity and commutativity of $+$. In other words, we do not distinguish

$t + u$ and $u + t$, nor $(t + u) + v$ and $t + (u + v)$. We use a *summation* $\sum_{i=1}^{n} t_i$ to denote $t_1 + \cdots + t_n$, where the empty sum represents $\mathbf{0}$. Such a summation is said to be in *head normal form* if each term t_i is of the form $a_i t'_i$ or $\alpha_i t'_i$ for some action a_i or action variable α_i, and term t'_i.

It is easy to see that modulo the axioms A1 and A2, every term t in the language BCCSP_Θ has the form $\sum_{i \in I} t_i$, for some finite index set I, and terms t_i $(i \in I)$ that do not have the form $t' + t''$. The terms t_i $(i \in I)$ will be referred to as the *summands* of t. For example, the term $\Theta(a\mathbf{0} + b\mathbf{0})$ has only itself as summand.

Remark 6.3.2 Note that the axiom system in Tab. 6.2 is not strong enough to prove all of the sound equations over the language BCCSP_Θ modulo bisimulation equivalence. For instance, as the readers can check, the equation

$$\Theta(\Theta(x) + y) \approx \Theta(x + y)$$

is sound modulo bisimulation equivalence irrespective of the cardinality of the set of actions Act and of the ordering relation $<$. That equation, however, cannot be proven from those in Tab. 6.2.

6.4 $|Act| = \infty$

In this section, we deal with the case that the action set is *infinite*. Our main result is that bisimulation equivalence does *not* afford a finite, ground-complete axiomatization over the language BCCSP_Θ, provided that Act contains at least two actions a, b with $a < b$. (Otherwise, the equation $\Theta(x) \approx x$ would be sound, and the priority operator could be eliminated from all terms.) This negative result even applies if BCCSP_Θ is extended with an arbitrary collection of operators (over finite synchronization trees) for which bisimulation equivalence is a congruence.

The idea behind the proof of our main result of this section is that a finite axiom system E can mention only finitely many action names. So, since Act is infinite, we can find a pair c, d of distinct actions that do not occur in E. If c and d are incomparable, then the equation $\Theta(c\mathbf{0} + d\mathbf{0}) \approx c\mathbf{0} + d\mathbf{0}$ is sound; if $c < d$, then $\Theta(c\mathbf{0} + d\mathbf{0}) \approx d\mathbf{0}$ is sound. In the first case, we show that an equational proof of $\Theta(c\mathbf{0} + d\mathbf{0}) \approx c\mathbf{0} + d\mathbf{0}$ from E would give rise to a proof of the unsound equation $\Theta(a\mathbf{0} + b\mathbf{0}) \approx a\mathbf{0} + b\mathbf{0}$ from E. This follows by a simple renaming argument, using that c and d do not occur in E. Likewise, in the second case, a proof of $\Theta(c\mathbf{0} + d\mathbf{0}) \approx d\mathbf{0}$ from E would give rise to a proof of the unsound equation $\Theta(d\mathbf{0} + c\mathbf{0}) \approx c\mathbf{0}$ from E.

To present the formal proof of the aforementioned negative result, first we introduce the action renaming mentioned in the proof idea sketched above.

Definition 6.4.1 Let $A \subseteq Act$, and let Σ be a signature that includes the set of operators in BCCSP_Θ. We extend each renaming function $\rho : A \to Act$ to a

function $\rho : \mathbb{T}(\Sigma) \to \mathbb{T}(\Sigma)$ as follows, where f is any operator that is not of the form $a\_$.

$$\begin{aligned}
\rho(\mathbf{0}) &\stackrel{\text{def}}{=} \mathbf{0} \\
\rho(at) &\stackrel{\text{def}}{=} \begin{cases} \rho(a)\rho(t) & \text{if } a \in A \\ a\rho(t) & \text{if } a \notin A \end{cases} \\
\rho(f(t_1, \ldots, t_n)) &\stackrel{\text{def}}{=} f(\rho(t_1), \ldots, \rho(t_n)) \\
\rho(x) &\stackrel{\text{def}}{=} x \\
\rho(\alpha t) &\stackrel{\text{def}}{=} \alpha\rho(t)
\end{aligned}$$

For each substitution σ, the substitution $\rho(\sigma)$ is defined by $\rho(\sigma)(x) \stackrel{\text{def}}{=} \rho(\sigma(x))$ and

$$\rho(\sigma)(\alpha) \stackrel{\text{def}}{=} \begin{cases} \rho(\sigma(\alpha)) & \text{if } \sigma(\alpha) \in A \\ \sigma(\alpha) & \text{otherwise .} \end{cases}$$

The following lemma states that renaming of actions that are not mentioned in an axiom system E preserves provability.

Lemma 6.4.2 Let $A \subseteq Act$ and $\rho : A \to Act$. Let Σ be a signature that includes the set of operators in BCCSP_Θ. Let E be a collection of equations over Σ, and assume that all of the actions $a \in A$ do not occur in E. Then $E \vdash p \approx q$ implies $E \vdash \rho(p) \approx \rho(q)$.

Proof: The proof is by induction on the depth of a closed proof of $p \approx q$ from E. We proceed by a case analysis on the last rule used in the proof of $p \approx q$ from E. The case of reflexivity is trivial, and that of transitivity follows immediately by using the induction hypothesis. Below we only consider the other cases, namely the instantiation of an axiom and closure under contexts. (Since we are dealing with closed proofs, closure with respect to prefixing by action variables need not be considered.)

- Case $E \vdash p \approx q$ because $\sigma(t) = p$ and $\sigma(u) = q$ for some equation $t \approx u \in E$ and closed substitution σ. Then $\rho(p) = \rho(\sigma(t)) = \rho(\sigma)(\rho(t))$. According to the proviso of the lemma, no action $a \in A$ occurs in t, so clearly $\rho(t) = t$. Similarly, $\rho(q) = \rho(\sigma(u)) = \rho(\sigma)(\rho(u)) = \rho(\sigma)(u)$. Since $t \approx u \in E$, by substitution instance, $E \vdash \rho(\sigma)(t) \approx \rho(\sigma)(u)$. In other words, $E \vdash \rho(p) \approx \rho(q)$, which was to be shown.

- Case $E \vdash p \approx q$ because $p = ap'$ and $q = aq'$ where $E \vdash p' \approx q'$. If $a \in A$, then $\rho(p) = \rho(a)\rho(p')$ and $\rho(q) = \rho(a)\rho(q')$; otherwise, $\rho(p) = a\rho(p')$ and $\rho(q) = a\rho(q')$. In either case, by induction, $E \vdash \rho(p') \approx \rho(q')$. By context closure, $E \vdash \rho(p) \approx \rho(q)$.

- Case $E \vdash p \approx q$ because $p = f(p_1, \ldots, p_n)$ and $q = f(q_1, \ldots, q_n)$, for some operator f in the signature that is not of the form $a\_$, where $E \vdash p_i \approx q_i$ for $i = 1, \ldots, n$. By definition, $\rho(p) = f(\rho(p_1), \ldots, \rho(p_n))$ and $\rho(q) =$

$f(\rho(q_1), \ldots, \rho(q_n))$. By induction, $E \vdash \rho(p_i) \approx \rho(q_i)$ for $i = 1, \ldots, n$. By context closure, $E \vdash \rho(p) \approx \rho(q)$.

The proof is now complete. ■

With Lem. 6.4.2 at our disposal, we are now in a position to show the first main result of this chapter:

Theorem 6.4.3 Let $|Act| = \infty$, and $a < b$ for two actions $a, b \in Act$. Let Σ be a signature consisting of the operators in BCCSP_Θ, together with auxiliary operators for which bisimulation equivalence is a congruence. Then bisimulation equivalence has no finite, sound and ground-complete axiomatization over $\mathrm{T}(\Sigma)$.

Proof: We need to show that no finite axiom system is both sound and ground-complete for $\mathrm{T}(\Sigma)$ modulo $\leftrightarrows$. Let E be a finite axiom system over $\mathrm{T}(\Sigma)$ that is sound modulo $\leftrightarrows$. Fix a pair of distinct actions $c, d \in Act$ that do not occur in E. We can select c, d such that either they are incomparable, or $c < d$. In the first case, the following equation is sound modulo $\leftrightarrows$:

$$\Theta(c\mathbf{0} + d\mathbf{0}) \approx c\mathbf{0} + d\mathbf{0} \ .$$

Assume, towards a contradiction, that this equation can be derived from E. Consider the renaming function ρ defined as: $\rho(c) = a$ and $\rho(d) = b$. Since c, d do not occur in E, Lem. 6.4.2 yields that $E \vdash \rho(\Theta(c\mathbf{0} + d\mathbf{0})) \approx \rho(c\mathbf{0} + d\mathbf{0})$. That is, $E \vdash \Theta(a\mathbf{0} + b\mathbf{0}) \approx a\mathbf{0} + b\mathbf{0}$, which is not sound modulo $\leftrightarrows$, since $a < b$. This contradicts the soundness of E.

In the second case, the following equation is sound modulo $\leftrightarrows$:

$$\Theta(c\mathbf{0} + d\mathbf{0}) \approx d\mathbf{0} \ .$$

Again, assume, towards a contradiction, that this equation can be derived from E. Consider the renaming function ρ defined as: $\rho(c) = d$ and $\rho(d) = c$. Since c, d do not occur in E, Lem. 6.4.2 yields that $E \vdash \rho(\Theta(c\mathbf{0} + d\mathbf{0})) \approx \rho(d\mathbf{0})$. That is, $E \vdash \Theta(d\mathbf{0} + c\mathbf{0}) \approx c\mathbf{0}$, which is not sound modulo $\leftrightarrows$. Once more, this contradicts the soundness of E.

In either case, we can conclude that the axiom system E is not ground-complete. ■

6.5 Axiomatizing Priority over an Infinite Action Set, Conditionally

Thm. 6.4.3 offers a very strong evidence that, in the presence of an infinite set of actions, equational logic is inherently not sufficiently powerful to achieve a finite axiomatization of bisimilarity over closed terms in the language BCCSP_Θ. Indeed, that result holds true even in the presence of an arbitrary number of auxiliary operators.

In the presence of action variables, it is natural to view our language as consisting of two sorts: one for actions and the other for processes. This is all the more true because the set of actions has the structure of a partial order, and we should like to express axioms over processes that reflect the influence that this poset structure on actions has on the behavior of processes. In case our set of actions is finite, this can be done by means of a finite number of equations that are instances of PR3 and PR4 in Tab. 6.2.

In the presence of an infinite action set, however, the axiom schemas PR3 and PR4, as well as PR2, have infinitely many instances. One way to try and capture their effects finitely is to take seriously the idea that, in the presence of action variables, the equation schemas PR3 and PR4 can be phrased as *equations with action predicates as conditions* thus:

$$\begin{aligned}
&\text{CPR3} \quad (\alpha < \beta) \Rightarrow \\
&\qquad\qquad \Theta(\alpha x + \beta y + z) \;\approx\; \Theta(\beta y + z) \\
&\text{CPR4}_n \quad \Big(\bigwedge_{1 \le i,j \le n} \neg(\alpha_i < \alpha_j)\Big) \Rightarrow \\
&\qquad\qquad \Theta(\alpha_1 x_1 + \cdots + \alpha_n x_n) \;\approx\; \alpha_1 \Theta(x_1) + \cdots + \alpha_n \Theta(x_n) \quad (n \ge 0) \;.
\end{aligned}$$

In both of the above equations, we use predicates over actions to restrict the applicability of the equation on the right-hand side of the implication. In general, henceforth in this study we shall consider equations of the form

$$P \Rightarrow t \approx u \;,$$

where P is a predicate over actions, and $t \approx u$ is an equation over the language BCCSP_Θ.

In what follows, we shall take a semantic view of predicates over actions. An action predicate P will be simply identified with the collection of closed substitutions that satisfy it – with the proviso that two closed substitutions that agree over the collection of action variables are either both in P or neither of them is. As we did above for CPR3 and $\mathrm{CPR4}_n$, we shall often express predicates over actions using formulae in first-order logic with equality and the binary relation symbol $<$. The definition of the collection of closed substitutions that satisfy a predicate P expressed using such formulae is entirely standard, and we omit the details. For example, a closed substitution σ satisfies the predicate $\alpha < \beta$ if, and only if, $\sigma(\alpha) < \sigma(\beta)$ holds in the poset $(Act, <)$. We sometimes write $\sigma(P) = \mathit{true}$ if the closed substitution σ satisfies the predicate P. We say that a predicate is *satisfiable* if some closed substitution satisfies it. If P is a tautology, then we simply write $t \approx u$. For instance, a version of PR2 with action variables will be written thus:

$$\text{CPR2} \qquad \Theta(\alpha x + \alpha y + z) \;\approx\; \Theta(\alpha x + z) + \Theta(\alpha y + z) \;.$$

Note that PR1 in Tab. 6.2 is just $\mathrm{CPR4}_0$. Moreover, since $<$ is irreflexive, $\mathrm{CPR4}_1$ reduces to

$$\Theta(\alpha x) \;\approx\; \alpha \Theta(x) \;. \tag{6.1}$$

(Note that the above equation can be derived from each of the $\mathrm{CPR4}_n$ with $n \geq 1$ and axiom A3 in Tab. 6.2.)

An equation of the form $P \Rightarrow t \approx u$ is *sound* with respect to bisimilarity, if $\sigma(t) \leftrightarrows \sigma(u)$ holds for each closed substitution σ that satisfies the predicate P. It is not hard to see that:

Lemma 6.5.1 For each partial order of actions $(Act, <)$, CPR2-3 and $\mathrm{CPR4}_n$ $(n \geq 0)$ are sound modulo bisimilarity over the language BCCSP_Θ.

A proof in conditional equational logic of an equation from a set E of axioms with action predicates as conditions uses the same rules presented in Section 2.1.3. However, the rule for substitution instance now reads

$$\frac{P \Rightarrow t \approx u}{\sigma(t) \approx \sigma(u)} \quad (\sigma(P) = \mathit{true}) \ ,$$

where $P \Rightarrow t \approx u$ is one of the equations with action predicates as conditions in the set E. Again, by postulating that for each equation of the form $P \Rightarrow (t \approx u)$ in E also its symmetric counterpart $P \Rightarrow (u \approx t)$ is present in E, we can disregard applications of symmetry in conditional equational proofs.

A natural question to ask at this point, and one that we shall address in the remainder of this study, is whether, unlike standard equational logic, equations with action predicates as conditions suffice to obtain a finite, ground-complete axiomatization of bisimulation equivalence over the language BCCSP_Θ.

In their classic paper [BBK86], Baeten, Bergstra and Klop offered a finite, ground-complete axiomatization of bisimilarity over the language BPA_δ with the priority operator that employs equations with action predicates as conditions. Their axiomatization, however, relied upon the introduction of a binary auxiliary operator, the so-called *unless* operator $\triangleleft$. Operationally, the behavior of the unless operator is specified by the rules

$$\frac{x \xrightarrow{a} x' \quad y \overset{b}{\not\rightarrow} \text{ for } a < b}{x \triangleleft y \xrightarrow{a} x'} \ ,$$

where $a \in Act$.

In the setting of BCCSP_Θ, and using action variables in lieu of concrete action names, the relation between the priority operator and the unless operator is expressed by the axioms in Tab. 6.3. It is not too hard to see that those axioms, together with A1-4, yield a ground-complete, finite axiomatization of bisimulation equivalence. Therefore, even in the presence of an infinite set of actions, bisimulation equivalence affords a finite, ground-complete axiomatization using equations with action predicates as conditions at the price of introducing a single auxiliary operator. But, if the set of actions is infinite, is the use of an auxiliary operator like the unless operator really necessary to obtain a finite axiomatizability result for bisimulation equivalence over BCCSP_Θ using equations with action predicates as conditions? We address this question in what follows. In particular, we first show that, in general, the use of auxiliary operators is

	$\Theta(\alpha x)$	$\approx$	αx
	$\Theta(\mathbf{0})$	$\approx$	$\mathbf{0}$
	$\Theta(x+y)$	$\approx$	$(\Theta(x) \triangleleft y) + (\Theta(y) \triangleleft x)$
$\neg(\alpha < \beta) \;\Rightarrow$	$(\alpha x) \triangleleft (\beta y)$	$\approx$	αx
$(\alpha < \beta) \;\Rightarrow$	$(\alpha x) \triangleleft (\beta y)$	$\approx$	$\mathbf{0}$
	$(\alpha x) \triangleleft \mathbf{0}$	$\approx$	αx
	$\mathbf{0} \triangleleft (\alpha x)$	$\approx$	$\mathbf{0}$
	$(x+y) \triangleleft z$	$\approx$	$(x \triangleleft z) + (y \triangleleft z)$
	$x \triangleleft (y+z)$	$\approx$	$(x \triangleleft y) \triangleleft z$

Table 6.3: Axioms for Θ in the presence of $\triangleleft$

indeed necessary to obtain a finite, ground-complete axiomatization of bisimulation equivalence using equations with action predicates as conditions. This we do in Section 6.5.1 by exhibiting a poset of actions for which no finite set of sound equations with action predicates as conditions is ground-complete with respect to bisimulation equivalence over BCCSP_Θ. This negative result, however, does not entail that, in the presence of an infinite set of actions, auxiliary operators are always needed to give a finite, ground-complete axiomatization of bisimulation equivalence over the language BCCSP_Θ. In fact, we then isolate sufficient conditions on the priority structure over actions that guarantee the finite axiomatizability of bisimulation equivalence over the language BCCSP_Θ using equations with action predicates as conditions (Section 6.5.2).

6.5.1 A Negative Result

Our order of business will now be to prove that, in the presence of an infinite set of actions, in general auxiliary operators are indeed necessary in order to obtain a finite ground-complete axiomatization of bisimulation equivalence over the language BCCSP_Θ, even if we permit the use of equations of the form $P \;\Rightarrow\; (t \approx u)$. In this section, $Act = \{a_i, b_i \mid i \geq 1\} \cup \{c\}$, where $a_i < b_i < c$ for each $i \geq 1$, and these are the only inequalities. Moreover, for convenience, we consider terms not only modulo associativity and commutativity of $+$, but also modulo the sound equations $x + \mathbf{0} \approx x$ and $\Theta(\Theta(x) + y) \approx \Theta(x + y)$ – see Rem. 6.3.2. So we can assume, without loss of generality, that terms contain neither redundant $\mathbf{0}$ summands nor nested occurrences of Θ.

We will prove the following claim, which will be used to argue that bisimulation equivalence has no finite, ground-complete axiomatization consisting of equations with action predicates as conditions over the language BCCSP_Θ (Thm. 6.5.6 to follow).

Claim 6.5.2 Let E be a finite collection of equations with action predicates as conditions that is sound modulo $\leftrightarrows$. Let $n \geq 2$ be larger than the size of any

term in the equations of E. Then from E we cannot derive the equation

$$\Theta(\Phi_n) \approx \Phi_n \ ,$$

where Φ_n denotes $\sum_{i=1}^{n} b_i\mathbf{0}$.

Note that the equation above is sound modulo $\leftrightarrows$, because the actions b_i ($i \geq 1$) are pairwise incomparable.

First we establish a technical lemma.

Lemma 6.5.3 Let $P \Rightarrow t \approx u$ be an equation that is sound modulo $\leftrightarrows$, where P is satisfiable. If some process variable x occurs as a summand in t, then x also occurs as a summand in u.

Proof: Since P is satisfiable, there exists a closed substitution σ such that $\sigma(P) = \mathit{true}$. Take some action $d \in \mathit{Act}$ that does not occur in $\sigma(u)$; such an action exists because A is infinite. Consider the closed substitution σ' that maps x to $d(b_1\mathbf{0} + c\mathbf{0})$, each other process variable to $\mathbf{0}$, and agrees with σ on action variables. As $P \Rightarrow t \approx u$ is sound modulo $\leftrightarrows$ and $\sigma'(P) = \sigma(P) = \mathit{true}$, we have that $\sigma'(t) \leftrightarrows \sigma'(u)$. Since x is a summand of t and $\sigma'(t) \xrightarrow{d} b_1\mathbf{0} + c\mathbf{0}$, it follows that $\sigma'(u) \xrightarrow{d} q \leftrightarrows b_1\mathbf{0} + c\mathbf{0}$ for some q. Since d does not occur in $\sigma(u)$ and $b_1 < c$, it is not hard to see that x must be a summand of u. ■

The following lemma is the crux in the proof of our claim. It states a property of closed terms that holds for all of the closed instantiations of axioms in any finite, sound collection of equations with action predicates as conditions. As we shall see later on, this property is also preserved by arbitrary conditional equational proofs from a finite, sound collection of equations with action predicates as conditions (Prop. 6.5.5).

Lemma 6.5.4 Let $P \Rightarrow t \approx u$ be sound modulo $\leftrightarrows$. Let σ be a closed substitution with $\sigma(P) = \mathit{true}$. Assume that:

- n is larger than the size of t, where $n \geq 2$; and
- the summands of $\sigma(t)$ are all bisimilar to either Φ_n or $\mathbf{0}$.

Then the summands of $\sigma(u)$ are all bisimilar to either Φ_n or $\mathbf{0}$.

Proof: First, suppose that all summands of $\sigma(t)$ are bisimilar to $\mathbf{0}$. Then $\sigma(t) \leftrightarrows \mathbf{0}$, so the soundness of $P \Rightarrow t \approx u$ together with $\sigma(P) = \mathit{true}$ yields $\sigma(u) \leftrightarrows \mathbf{0}$. This means that all summands of $\sigma(u)$ are bisimilar to $\mathbf{0}$, and we are done.

So we can assume that some summand of $\sigma(t)$ is bisimilar to Φ_n. Then $\sigma(t) \leftrightarrows \sigma(u) \leftrightarrows \Phi_n$, by the proviso of the lemma and the soundness of $P \Rightarrow t \approx u$.

We know that we can write $t = \sum_{i \in I} t_i$ and $u = \sum_{j \in J} u_j$ for some non-empty, finite index sets I and J, where the terms t_i and u_j are of the form x, av, αv or $\Theta(v)$. By the proviso of the lemma, for each $i \in I$, the summands of

$\sigma(t_i)$ are all bisimilar to Φ_n or $\mathbf{0}$. Since $n \geq 2$, for each $i \in I$, the term t_i is not of the form av or αv. Hence either it is a process variable x, or it is of the form

$$\Theta(\sum_{\ell \in L_i} d_{i\ell} t'_{i\ell} + \sum_{m \in M_i} \alpha_m t''_{im} + \sum_{k \in K_i} z_{ik})$$

(modulo the equations $x + \mathbf{0} \approx x$ and $\Theta(\Theta(x) + y) \approx \Theta(x + y)$). Let $I' \subseteq I$ be the set of indices of summands of t that have the above form. Observe that $K_i \neq \emptyset$ for each $i \in I'$ such that $\sigma(t_i)$ is bisimilar to Φ_n (because n is larger than the size of t). Note moreover that summands t_i of t having the above form such that $\sigma(t_i) \leftrightarrows \mathbf{0}$ must have $L_i = M_i = \emptyset$, and for such summands $\sigma(z_{ik}) \leftrightarrows \mathbf{0}$ for each $k \in K_i$.

Let us assume, towards a contradiction, that there is an index $j \in J$ such that $\sigma(u_j)$ has a summand that is bisimilar neither to Φ_n nor to $\mathbf{0}$. We proceed by a case analysis on the form of u_j.

1. Case $u_j = x$. By assumption, $\sigma(x)$ has a summand that is bisimilar neither to Φ_n nor to $\mathbf{0}$. Since $P \Rightarrow t \approx u$ is sound modulo $\leftrightarrows$ and P is satisfiable (because $\sigma(P) = \mathit{true}$ by the proviso of the lemma), by Lem. 6.5.3, t also has x as a summand. Consequently $\sigma(t)$ has a summand that is bisimilar neither to Φ_n nor to $\mathbf{0}$. This contradicts one the assumptions of the lemma.

2. Case $u_j = au'_j$ or $u_j = \alpha u'_j$. Since $\sigma(u) \leftrightarrows \Phi_n$, we have that $a = b_h$ or $\sigma(\alpha) = b_h$ for some $1 \leq h \leq n$. Define the substitution σ' as

$$\sigma'(y) = \begin{cases} c\mathbf{0} & \text{if } y = z_{ik} \text{ for some } i \in I' \text{ and } k \in K_i \\ \mathbf{0} & \text{otherwise} \end{cases}$$

   for process variables y, and let σ' agree with σ on action variables. Then $\sigma'(t) \overset{b_h}{\not\to}$, because

   - $c > b_h$,
   - $K_i \neq \emptyset$ for every $i \in I'$ with $\sigma(t_i) \leftrightarrows \Phi_n$,
   - $L_i = M_i = \emptyset$ for every $i \in I'$ with $\sigma(t_i) \leftrightarrows \mathbf{0}$ and
   - t does not contain summands of the form $b_h v$ or αv.

   On the other hand, as σ and σ' agree on action variables, $\sigma'(u_j) \xrightarrow{b_h} \sigma'(u'_j)$. It follows that $\sigma'(u) \xrightarrow{b_h} \sigma'(u'_j)$, and thus $\sigma'(t) \not\leftrightarrows \sigma'(u)$. Since $\sigma'(P) = \sigma(P) = \mathit{true}$, this contradicts the soundness of $P \Rightarrow t \approx u$ modulo $\leftrightarrows$.

3. Case $u_j = \Theta(u')$. Then u_j consists of a single summand, so by assumption, we have that $\sigma(u_j) \not\leftrightarrows \Phi_n$ and $\sigma(u_j) \not\leftrightarrows \mathbf{0}$.

   Since $\sigma(u) \leftrightarrows \Phi_n$, and terms are considered modulo the equations $x + \mathbf{0} \approx x$ and $\Theta(\Theta(x) + y) \approx \Theta(x + y)$, we can take u' to be of the form

$$\sum_{\ell \in L} e_\ell u'_\ell + \sum_{m \in M} \beta_m u''_m + \sum_{k \in K} y_k \ ,$$

for some finite index sets L, M, K. We distinguish two cases.

(a) For each $i \in I'$ with $\sigma(t_i) \not\leftrightarrows \mathbf{0}$ there is a $k_i \in K_i$ such that z_{ik_i} is not a summand of u'.
Define the substitution σ' as

$$\sigma'(y) = \begin{cases} c\mathbf{0} & \text{if } y = z_{ik_i} \text{ for some } i \in I' \text{ with } \sigma(t_i) \not\leftrightarrows \mathbf{0}, \text{ or} \\ & \text{if } y \text{ is a summand of } t \text{ with } \sigma(y) \not\leftrightarrows \mathbf{0} \\ \sigma(y) & \text{otherwise} \end{cases}$$

for process variables y, and let σ' agree with σ on action variables. It is not hard to see that $\sigma'(t) \overset{b_i}{\not\rightarrow}$ for $i = 1, \ldots, n$ (because $c > b_i$ and t has no summand of the form av or αv). On the other hand, since $\sigma(u_j) \not\leftrightarrows \mathbf{0}$ and $\sigma(u) \leftrightarrows \Phi_n$, there is an h with $1 \leq h \leq n$ such that $\sigma(u') \xrightarrow{b_h}$. Furthermore, $\sigma(u') \overset{c}{\not\rightarrow}$. By assumption, z_{ik_i} is not a summand of u' for each $i \in I'$ with $\sigma(t_i) \not\leftrightarrows \mathbf{0}$. Moreover, for any variable summand y of t with $\sigma(y) \not\leftrightarrows \mathbf{0}$, y is not a summand of u', because by assumption $\sigma(y) \leftrightarrows \Phi_n$ while $\sigma(u') \not\leftrightarrows \Phi_n$. So $\sigma(u') \xrightarrow{b_h}$ and $\sigma(u') \overset{c}{\not\rightarrow}$ imply $\sigma'(u') \xrightarrow{b_h}$ and $\sigma'(u') \overset{c}{\not\rightarrow}$. It follows that $\sigma'(u_j) \xrightarrow{b_h}$, and so $\sigma'(u) \xrightarrow{b_h}$. Hence $\sigma'(t) \not\leftrightarrows \sigma'(u)$. Since $\sigma'(P) = \sigma(P) = \mathit{true}$, this contradicts the fact that $P \Rightarrow t \approx u$ is sound modulo $\leftrightarrows$.

(b) $\{z_{i_0k} \mid k \in K_{i_0}\} \subseteq \{y_k \mid k \in K\}$, for some $i_0 \in I'$ with $\sigma(t_{i_0}) \not\leftrightarrows \mathbf{0}$.
In this case, K is non-empty since, as previously observed, K_{i_0} is non-empty. By the proviso of the lemma, $\sigma(t_{i_0}) \leftrightarrows \Phi_n$, so (since n is larger than the size of t_{i_0}) there is a $k_0 \in K_{i_0}$ with $\sigma(z_{i_0k_0}) \not\leftrightarrows \mathbf{0}$. Furthermore, by the assumption for case 3 of the proof, $\sigma(u_j) \not\leftrightarrows \mathbf{0}$ and $\sigma(u_j) \not\leftrightarrows \Phi_n$. Therefore, there is an h with $1 \leq h \leq n$ such that $\sigma(\Theta(u')) \overset{b_h}{\not\rightarrow}$. Define the substitution σ' as

$$\sigma'(y) = \begin{cases} a_h\mathbf{0} & \text{if } y = z_{i_0k_0} \\ \sigma(y) & \text{otherwise} \end{cases}$$

for process variables y, and let σ' agree with σ on action variables. We argue that $\sigma'(t) \overset{a_h}{\not\rightarrow}$. To this end, observe, first of all, that, since $\sigma(\Theta(u')) \overset{b_h}{\not\rightarrow}$, we have $\sigma(\sum_{k \in K} y_k) \overset{b_h}{\not\rightarrow}$, and so $\sigma(z_{i_0k_0}) \overset{b_h}{\not\rightarrow}$. We are now ready to show that no summand of $\sigma'(t)$ affords an a_h-labeled transition. We consider three exhaustive possibilities:

i. Let $i \in I'$ with $z_{i_0k_0} \notin \{z_{ik} \mid k \in K_i\}$. Then clearly $\sigma'(t_i) \overset{a_h}{\not\rightarrow}$.

ii. Let $i \in I'$ with $z_{i_0k_0} \in \{z_{ik} \mid k \in K_i\}$. Then $\sigma(t_i) \not\leftrightarrows \mathbf{0}$ because $\sigma(z_{i_0k_0}) \not\leftrightarrows \mathbf{0}$, so by assumption $\sigma(t_i) \leftrightarrows \Phi_n$. This implies $\sigma(t_i) \xrightarrow{b_h}$, so since $\sigma(z_{i_0k_0}) \overset{b_h}{\not\rightarrow}$, it follows that $\sigma'(t_i) \xrightarrow{b_h}$. Since the outermost function symbol of t_i is Θ, we can conclude that $\sigma'(t_i) \overset{a_h}{\not\rightarrow}$.

iii. Finally, since $\sigma(z_{i_0k_0}) \not\leftrightarrows \mathbf{0}$ and $\sigma(z_{i_0k_0}) \stackrel{b_h}{\to}$, the proviso of the lemma yields that $z_{i_0k_0}$ cannot be a summand of t.

Since t has no other types of summands, from the three cases above we can conclude that $\sigma'(t) \stackrel{a_h}{\to}$. On the other hand, $\sigma'(\Theta(u')) \stackrel{a_h}{\longrightarrow}$, because $\sigma(\Theta(u')) \stackrel{b_h}{\to}$ and $z_{i_0k_0} \in \{y_k \mid k \in K\}$. Hence $\sigma'(u) \stackrel{a_h}{\longrightarrow}$, and so $\sigma'(t) \not\leftrightarrows \sigma'(u)$. Since $\sigma'(P) = \sigma(P) = \mathit{true}$, this contradicts the fact that $P \Rightarrow t \approx u$ is sound modulo $\leftrightarrows$.

In summary, the assumption that, for some $j \in J$, the term $\sigma(u_j)$ has a summand that is bisimilar neither to Φ_n nor to $\mathbf{0}$, leads to a contradiction. This completes the proof. ■

The following proposition states that the property of closed instantiations of sound equations with action predicates as conditions mentioned in the above lemma is preserved under equational derivations from a finite collection of sound equations. This is the key to the promised proof of our claim.

Proposition 6.5.5 Let E be a finite collection of equations with action predicates as conditions that is sound modulo $\leftrightarrows$. Let $n \geq 2$ be larger than the size of any term in the equations of E. Assume, furthermore, that

- $E \vdash p \approx q$; and
- the summands of p are all bisimilar to Φ_n or $\mathbf{0}$.

Then the summands of q are all bisimilar to Φ_n or $\mathbf{0}$.

Proof: By induction on the depth of the closed proof of $p \approx q$ from E. We proceed by a case analysis on the last rule used in the proof of $p \approx q$ from E.

- $E \vdash p \approx q$ because $\sigma(t) = p$ and $\sigma(u) = q$ for some equation $P \Rightarrow t \approx u \in E$ and closed substitution σ with $\sigma(P) = \mathit{true}$. The claim follows immediately from Lem. 6.5.4.
- $E \vdash p \approx q$ because $p = p' + p''$ and $q = q' + q''$ for some p', q', p'', q'' such that $E \vdash p' \approx q'$ and $E \vdash p'' \approx q''$. Since the summands of p are all bisimilar to Φ_n or $\mathbf{0}$, the same holds for p' and p''. By induction, the summands of q' and q'' are all bisimilar to Φ_n or $\mathbf{0}$. The claim now follows because the summands of q are those of q' and q''.
- $E \vdash p \approx q$ because $p = ap'$ and $q = aq'$ for some p', q' such that $E \vdash p' \approx q'$. This case is vacuous, because $n \geq 2$ and $p \not\leftrightarrows \Phi_n$.
- $E \vdash p \approx q$ because $p = \alpha p'$ and $q = \alpha q'$ for some p', q' such that $E \vdash p' \approx q'$. This case is vacuous, because p and q are closed.
- $E \vdash p \approx q$ because $p = \Theta(p')$ and $q = \Theta(q')$ for some p', q' such that $E \vdash p' \approx q'$. The claim is immediate, because both p and q consist of a single summand, and $p \leftrightarrows q$ by the soundness of E.

The proof is now complete. ■

Theorem 6.5.6 Let $Act = \{a_i, b_i \mid i \geq 1\} \cup \{c\}$, where $a_i < b_i < c$ for each $i \geq 1$, and these are the only inequalities. Then bisimulation equivalence has no ground-complete axiomatization over BCCSP_Θ consisting of a finite set of sound equations with action predicates as conditions.

Proof: Let E be a finite collection of equations with action predicates as conditions that is sound modulo $\leftrightarrows$. Let $n \geq 2$ be larger than the size of any term in the equations of E. According to Prop. 6.5.5, from E we cannot derive $\Theta(\Phi_n) \approx \Phi_n$. This equation is sound modulo $\leftrightarrows$, and therefore E is not ground-complete. ■

6.5.2 Positive Results

We have offered an example of a priority structure $(Act, <)$ with respect to which it is impossible to give a finite, ground-complete axiomatization of bisimulation equivalence over BCCSP_Θ in terms of equations with action predicates as conditions without recourse to auxiliary operators. That result, however, does not imply that auxiliary operators are always necessary to achieve a finite basis of equations with action predicates as conditions for bisimulation equivalence. Our aim in this section is to substantiate this claim by providing some general conditions over the priority structure $(Act, <)$ that are sufficient to guarantee the existence of a finite, ground-complete axiomatization of bisimulation equivalence over BCCSP_Θ that uses equations with action predicates as conditions.

Definition 6.5.7 An anti-chain in a poset $(Act, <)$ is a subset of Act consisting of pairwise incomparable actions. The *width* of a poset $(Act, <)$ is the least upper bound of the cardinalities of its anti-chains. A poset $(Act, <)$ has *finite width* if its width is finite.

Example 6.5.8 The poset of actions we considered in Section 6.5.1 has uncountably many infinite, maximal anti-chains. (Each such anti-chain can, in fact, be obtained by picking exactly one of a_i and b_i for each $i \geq 1$.) The width of that poset is therefore infinite.

We now offer a countably infinite, ground-complete axiomatization of bisimulation equivalence over BCCSP_Θ using equations with action predicates as conditions. Such an axiomatization reduces to a finite one if the poset of actions has finite width.

Theorem 6.5.9 Let $(Act, <)$ be an infinite poset of actions. Then the following statements hold:

1. The axiom system consisting of the CPR2-3 and $\mathrm{CPR4}_n$ ($n \geq 0$), together with A1-4 is ground-complete for bisimilarity over the language BCCSP_Θ.

2. Assume that the width of $(Act, <)$ is k. Then the axiom system consisting of CPR2-3, and CPR4$_k$, together with A1-4 and PR1 in Tab. 6.2, is ground-complete for bisimilarity over the language BCCSP$_\Theta$. Therefore bisimilarity has a finite, ground-complete axiomatization using equations with action predicates as conditions if $(Act, <)$ has finite width.

Proof: We only present a sketch of the proof for statement 2. (That for statement 1 follows similar lines.)

First of all, observe that it suffices only to show that, if the cardinality of each anti-chain in $(Act, <)$ is at most k, CPR2-3, CPR4$_k$ and PR1 can be used to remove all occurrences of Θ from closed terms. Indeed, if we can do so, then ground-completeness follows from the well-known ground-completeness of A1-4 for BCCSP modulo $\leftrightarrows$ (see, e.g., [HM85]).

To prove that all occurrences of Θ can be removed from closed terms, assume that we have a closed term p that does not contain occurrences of Θ. We show that $\Theta(p)$ can be proven equal to a term q that does not contain occurrences of Θ by induction on the size of p. To this end, note that, modulo associativity and commutativity of $+$, the term p can be written $\sum_{i=1}^{n} a_i p_i$ for some $n \geq 0$, actions a_i and closed terms p_i that do not contain occurrences of Θ.

If $n = 0$, then PR1 yields that $\Theta(\mathbf{0}) \approx \mathbf{0}$, and we are done. If $n = 1$, then the claim follows using (6.1) and the induction hypothesis. (Recall that, since $k \geq 1$, Eq. (6.1) is derivable from CPR4$_k$.) Consider now the case when $n \geq 2$. We proceed by examining the following three sub-cases:

- there are i, j such that $1 \leq i < j \leq n$ and $a_i = a_j$,
- there are i, j such that $1 \leq i, j \leq n$ and $a_i < a_j$, and
- the collection of actions $\{a_1, \ldots, a_n\}$ is an anti-chain in the poset $(Act, <)$.

The first two sub-cases are handled using the induction hypothesis, and CPR2 and CPR3, respectively.

If the proviso for the third sub-case applies, then we know that $n \leq k$. Using A3 if $n < k$, we can therefore reason as follows:

$$\begin{aligned}
\Theta(\sum_{i=1}^{n} a_i p_i) &\approx \Theta(\sum_{i=1}^{n} a_i p_i + \underbrace{a_n p_n + \cdots + a_n p_n}_{(k-n) \text{ times}}) \\
&\approx \sum_{i=1}^{n} a_i \Theta(p_i) \quad \text{(by CPR4}_k \text{ and possibly A3)} \\
&\approx \sum_{i=1}^{n} a_i q_i \quad \text{(by the induction hypothesis)}
\end{aligned}$$

for some closed terms $q_1, \ldots, q_n$ that do not contain occurrences of Θ.

Using this result, a simple argument by structural induction over closed terms shows that each closed term in the language BCCSP$_\Theta$ is provably equal

to one that does not contain occurrences of the Θ operator. The proof is complete. ■

So bisimilarity affords a finite, ground-complete axiomatization that uses equations with action predicates as conditions if the poset $(Act, <)$ has finite width. (Moreover, the equations with action predicates as conditions making up the axiom systems used in Thm. 6.5.9 only involve predicates over actions that can be expressed as conjunctions of, possibly negated, atomic formulae of the form $\alpha < \beta$.) A natural question to ask at this point is whether this result holds for more general priority structures. We now proceed to address this question in some detail.

Let us begin by observing that there are priority structures with infinite anti-chains that *do* allow for a finite, ground-complete axiomatization of bisimilarity over the language BCCSP_Θ. Consider, by way of example, the flat priority structure $(\{\bot, a_0, a_1, \ldots\}, <)$, where the only ordering relations are given by $\bot < a_i$ for each $i \geq 0$. Membership of the countably infinite anti-chain $\{a_0, a_1, \ldots\}$ can be characterized by the predicate

$$P(\alpha) = \forall \beta \; \neg(\alpha < \beta) \;\; .$$

We can therefore write the following equation that allows us to reduce the number of summands within the scope of a Θ operator:

$$P(\alpha) \wedge P(\beta) \Rightarrow \Theta(\alpha x + \beta y + z) \;\approx\; \Theta(\alpha x + z) + \Theta(\beta y + z) \;\; . \tag{6.2}$$

It is not hard to see that the above equation is sound. (In fact, the soundness of this equation will follow from the more general result in Lem. 6.5.12.) Moreover, following the lines of the proof sketch for Thm. 6.5.9(2), one can argue that, together with PR1, CPR2-3 and (6.1), this equation can be used to remove all occurrences of Θ from closed terms. It follows that:

Proposition 6.5.10 Consider the priority poset $(\{\bot, a_0, a_1, \ldots\}, <)$, where the only ordering relations are given by $\bot < a_i$ for each $i \geq 0$. Then the axiom system consisting of the equations (6.2), CPR2-3 and (6.1), together with A1-4 and PR1 in Tab. 6.2, is ground-complete for bisimilarity over the language BCCSP_Θ.

As another example, consider the priority structure

$$\mathcal{A} = (\{a_0, a_1, \ldots\} \cup \{b_0, b_1, c\}, <) \;\; ,$$

where the relation $<$ is the least transitive relation satisfying

$$\begin{array}{lcll} b_i & < & a_j & \text{for all } i \in \{0,1\}, j \geq 0 \text{ and} \\ a_j & < & c & \text{for each } j \geq 0 \;\; . \end{array}$$

This poset has one non-trivial maximal finite anti-chain, namely $\{b_0, b_1\}$, and one maximal countably infinite anti-chain, namely

$$A \;\; = \;\; \{a_0, a_1, \ldots\} \;\; .$$

Membership of A is characterized by the predicate P_A defined thus:

$$P_A(\alpha) \quad = \quad \exists \beta_1, \beta_2.\ \beta_1 < \alpha < \beta_2 \ .$$

As the readers can check, the instance of Eq. (6.2) associated with this predicate is sound. (Again, the soundness of this equation will follow from the more general result in Lem. 6.5.12.) Moreover, following the lines of the proof sketch for Thm. 6.5.9(2), one can argue that, together with PR1, CPR2-3 and CPR4$_2$ (to handle the finite anti-chain $\{b_0, b_1\}$), this equation can be used to remove all occurrences of Θ from closed terms. It follows that:

Proposition 6.5.11 Consider the priority poset $\mathcal{A}$. Then the axiom system consisting of Eq. (6.2) for predicate P_A, CPR2-3 and CPR4$_2$, together with A1-4 and PR1 in Tab. 6.2, is ground-complete for bisimilarity over BCCSP$_\Theta$.

In both of the examples we have just presented, Eq. (6.2) plays a key role in that it allows us to reduce the size of terms in "head normal form" having summands of the form ap and bq with a, b contained in an infinite anti-chain within the scope of a Θ operator. The following lemma states a necessary and sufficient condition on the infinite anti-chain that guarantees that axiom (6.2) be sound modulo bisimilarity.

Lemma 6.5.12 Let A be an anti-chain in the poset $(Act, <)$ whose membership is described by predicate P_A. Then Eq. (6.2) for predicate P_A is sound modulo bisimilarity iff each element of A is above the same set of actions, that is, for each $a, b \in A$ and $c \in Act$, we have that $c < a$ iff $c < b$.

Proof: We first prove the "if implication". To this end, assume that $a, b \in A$ and p, q, r are closed terms in the language BCCSP$_\Theta$. We claim that

$$\Theta(ap + bq + r) \leftrightarrows \Theta(ap + r) + \Theta(bq + r) \ .$$

To see that this claim does hold, it suffices only to observe that the following statements hold for each closed term p':

1. $\Theta(ap + bq + r) \xrightarrow{a} p'$ iff $\Theta(ap + r) + \Theta(bq + r) \xrightarrow{a} p'$,
2. $\Theta(ap + bq + r) \xrightarrow{b} p'$ iff $\Theta(ap + r) + \Theta(bq + r) \xrightarrow{b} p'$, and
3. $\Theta(ap + bq + r) \xrightarrow{c} p'$ iff $\Theta(ap + r) + \Theta(bq + r) \xrightarrow{c} p'$, for each action c different from a, b.

We only offer a proof for the last of these statements. To this end, assume, first of all, that $\Theta(ap + bq + r) \xrightarrow{c} p'$ for some action c different from a, b and closed term p'. Since c is different from a, b, there is a closed term r' such that

- $p' = \Theta(r')$,
- $r \xrightarrow{c} r'$,

- $r \stackrel{d}{\not\longrightarrow}$ for each action d such that $c < d$, and
- neither $c < a$ nor $c < b$ holds.

It is now a simple matter to see that, for instance, $\Theta(ap+r) \stackrel{c}{\longrightarrow} p'$. This yields that $\Theta(ap+r) + \Theta(bq+r) \stackrel{c}{\longrightarrow} p'$, which was to be shown.

Conversely, suppose that $\Theta(ap+r) + \Theta(bq+r) \stackrel{c}{\longrightarrow} p'$ for some action c different from a, b and closed term p'. Without loss of generality, we may assume that this is because $\Theta(ap+r) \stackrel{c}{\longrightarrow} p'$. Since c is different from a, b, there is a closed term r' such that

- $p' = \Theta(r')$,
- $r \stackrel{c}{\longrightarrow} r'$,
- $r \stackrel{d}{\not\longrightarrow}$ for each action d such that $c < d$, and
- $c < a$ does not hold.

Observe now that $c < b$ does not hold either, because a and b are above the same actions by the proviso of the lemma. It follows that $\Theta(ap+bq+r) \stackrel{c}{\longrightarrow} p'$, which was to be shown.

To establish the "only if implication", assume that A contains two distinct incomparable actions a and b that are *not* above the same set of actions. Suppose, without loss of generality, that $c < a$, but $c < b$ does not hold, for some action c. Then

$$\Theta(a\mathbf{0} + b\mathbf{0} + c\mathbf{0}) \leftrightarrows a\mathbf{0} + b\mathbf{0} \not\leftrightarrows a\mathbf{0} + b\mathbf{0} + c\mathbf{0} \leftrightarrows \Theta(a\mathbf{0} + c\mathbf{0}) + \Theta(b\mathbf{0} + c\mathbf{0}) \ .$$

(The last equivalence holds true because b and c must be incomparable, as $c < a$ and a and b are incomparable.) Therefore Eq. (6.2) for predicate P_A is not sound modulo bisimilarity. ■

Remark 6.5.13 Let A, B be two different, maximal anti-chains in the poset $(Act, <)$. Assume that each element of A is above the same set of actions, that is, for each $a, b \in A$ and $c \in Act$, we have that $c < a$ iff $c < b$, and so is each element of B. Then A and B are disjoint.

To see this, assume, towards a contradiction, that $a \in A \cap B$. Since A and B are maximal anti-chains, neither one is a subset of the other. Therefore, since $A \neq B$, there are actions b, c such that $b \in A \setminus B$ and $c \in B \setminus A$. It follows that a, b, c are above the same set of actions in Act. However, $b \notin B$. Therefore, since B is maximal, there must be some action $d \in B$ with $b < d$ or $d < b$. If $b < d$, we have that $b < a$ because $a, d \in B$ and each element of B is above the same actions. This contradicts the assumption that A is an anti-chain. If $d < b$ then reasoning as above we can reach a contradiction to the assumption that B is an anti-chain. Therefore, A and B must be disjoint.

Suppose that p is a closed term in head normal form whose set of initial actions is included in an infinite anti-chain satisfying the constraint in the statement of Lem. 6.5.12. Then the sound equation (6.2) offers a way of "simplifying" the term $\Theta(p)$. The use of this axiom is the key to the proof of the following generalization of Thm. 6.5.9(2), and of Prop. 6.5.10 and 6.5.11.

Theorem 6.5.14 Let $(Act, <)$ be an infinite poset of actions. Assume that

1. the collection of the sizes of the finite, maximal anti-chains in $(Act, <)$ is finite,
2. $(Act, <)$ has finitely many infinite, maximal anti-chains, and
3. for each infinite, maximal anti-chain A in $(Act, <)$, each element of A is above the same set of actions, that is, for each $a, b \in A$ and $c \in Act$, we have that $c < a$ iff $c < b$.

Let k be the size of the largest finite, maximal anti-chain in $(Act, <)$, or 1 if all maximal anti-chains are infinite. Then the axiom system consisting of one instance of Eq. (6.2) for predicate P_A for each infinite anti-chain A in $(Act, <$), CPR2-3 and CPR4$_k$, together with A1-4 and PR1 in Tab. 6.2, is ground-complete for bisimilarity over the language BCCSP$_\Theta$.

Proof: The soundness of the axiom system is easily established, using Lem. 6.5.12 for the instances of axiom (6.2). The completeness of the axiom system can be shown along the lines of the proof of Thm. 6.5.9. The key of the argument is again to prove that each term $\Theta(\sum_{i=1}^{n} a_i p_i)$, where the p_i do not contain occurrences of Θ, can be proven equal to a term q that does not contain occurrences of Θ by induction on the size of $\sum_{i=1}^{n} a_i p_i$. This we do by considering several sub-cases depending on the number n of summands in $\sum_{i=1}^{n} a_i p_i$.

If $n = 0$, then the claim follows using PR1. If $n = 1$, then it suffices only to use (6.1) and the induction hypothesis. (Recall that (6.1) is derivable from CPR4$_k$.) If $n \geq 2$, then we distinguish the following sub-cases:

- there are i, j such that $1 \leq i < j \leq n$ and $a_i = a_j$,
- there are i, j such that $1 \leq i, j \leq n$ and $a_i < a_j$,
- the collection of actions $\{a_1, \ldots, a_n\}$ is an anti-chain in the poset $(Act, <)$.

The first two sub-cases are handled using the induction hypothesis, and the equations with action predicates as conditions CPR2 and CPR3, respectively.

The last sub-case is handled using CPR4$_k$ as in the proof of Thm. 6.5.9 if the set of actions $\{a_1, \ldots, a_n\}$ is included in a finite maximal anti-chain. Assume now that $\{a_1, \ldots, a_n\}$ is only included in an infinite maximal anti-chain, say A. (In fact, Rem. 6.5.13 ensures that such an anti-chain A is unique.) Using the instance of Eq. (6.2) for predicate P_A and induction, the claim follows.

The rest of the proof follows the lines of that of Thm. 6.5.9, and is therefore omitted. ■

Remark 6.5.15 The priority structure we employed in our proof of Thm. 6.5.6 satisfies neither condition 2 nor condition 3 in the proviso of the above theorem.

In light of the above result, bisimilarity has a finite, ground-complete axiomatization using equations with action predicates as conditions over the language BCCSP_Θ if the poset of actions satisfies the proviso of the above theorem. The above theorem therefore generalizes Prop. 6.5.10 and 6.5.11. A further example of a priority structure that satisfies the conditions stated in Thm. 6.5.14 is one having a finite collection of "priority levels" each consisting of an infinite set of actions – consider, for instance, the poset

$$(\{a_{ij} \mid 1 \leq i \leq N, j \geq 1\}, <) \ ,$$

where N is a positive integer and $a_{ij} < a_{hk}$ holds iff $i < h$.

We have not yet attempted a complete classification of the priority structures for which bisimulation equivalence affords a finite axiomatization in terms of equations with action predicates as conditions over the language BCCSP_Θ. This is most likely a hard problem which we leave for future research.

6.6 Conclusion

We have investigated the equational theory of the bisimulation equivalence over the process algebra BCCSP extended with the priority operator of Baeten, Bergstra and Klop in depth. We show that, in the presence of an infinite set of actions, the bisimulation equivalence has no finite, sound, ground-complete axiomatization over that language. This negative result applies even if the syntax is extended with an arbitrary collection of auxiliary operators. We then considered axiomatizations using equations with action predicates as conditions where in the presence of an infinite set of actions, it is shown that in general, the bisimulation equivalence still has no finite, sound, ground-complete axiomatization. Finally, we identified sufficient conditions on the priority structure over actions that lead to a finite, ground-complete axiomatization of bisimulation equivalence using equations with action predicates as conditions.

Open question. In this chapter, generally we considered some specific priority structure. However, a natural question[3] is left open, namely, whether the set of equations that hold in *all* priority structures is finitely based? We note that $\Theta(\Theta(x) + y) = \Theta(x + y)$ serves as an example of such sound equations. Another example might be $\Theta(x) + \Theta(y) \approx \Theta(x) + \Theta(y) + \Theta(x + y)$. Also note that the equation we gave in Claim. 6.5.2, i.e., $\Theta(\sum_{i=1}^{n} b_i\mathbf{0}) \approx \sum_{i=1}^{n} b_i\mathbf{0}$, does *not* hold in all priority structures.

[3]This is raised by Jaco van de Pol when he reviewed a manuscript of the dissertation.

Part II

Verification of Probabilistic Real-time Systems

Chapter 7

Model Checking of CTMCs Against DTA Specifications

7.1 Introduction

Model checking *continuous-time Markov chains* (CTMCs) has been focused on the *continuous stochastic logic* (CSL, [ASSB00, BHHK03]), which is *branching-time* in nature. This chapter concerns the problem of verifying CTMCs versus *linear* real-time specifications, which are based on *deterministic timed automata* (DTA, [AD94]). Concretely speaking, we explore the following problem: given a CTMC $\mathcal{C}$, and a *linear* real-time property provided as a DTA, $\mathcal{A}$, what is the probability of the set of paths of $\mathcal{C}$ accepted by $\mathcal{A}$? (Below we term it as "probability of $\mathcal{C} \models \mathcal{A}$" for simplicity.) We show that this set of paths is *measurable* and computing its probability can be reduced to computing the reachability probability in a *piecewise deterministic Markov process* (PDP, [Dav84]), a model that is used in an enormous variety of applied problems of engineering, operations research, (stochastic) control theory, management science and economics; examples include queueing systems, stochastic scheduling, fault detection in process systems, etc. This result relies on a product construction of CTMC $\mathcal{C}$ and DTA $\mathcal{A}$, denoted by $\mathcal{C} \otimes \mathcal{A}$, yielding a *deterministic Markov timed automaton* (DMTA), a variant of DTA in which, besides the usual ingredients of timed automata, like guards and clock resets, the location residence time is exponentially distributed. We show that the probability of $\mathcal{C} \models \mathcal{A}$ coincides with the reachability probability of accepting paths in $\mathcal{C} \otimes \mathcal{A}$. The underlying PDP of a DMTA is obtained by a slight adaption of the standard region construction [AD94]. The desired reachability probability is characterized as the least solution of a system of *integral equations* that is obtained from the PDP. Finally, this probability is shown to be approximated by solving a system of *partial differential equations* (PDEs). For single-clock DTA, we show that the system of integral equations can be transformed into a system of *linear equations*, where the coefficients are solutions of some *ordinary differential equations* (ODEs), for which either an analytical solution can be obtained or a numerical solution can be approximated arbitrarily close in an efficient way.

Related work. [BCH+07] and [DHS09] present model checking algorithms of asCSL and $\mathrm{CSL}^{\mathrm{TA}}$ respectively. asCSL allows one to impose a time constraint on action sequences described by regular expressions; its model-checking algorithm is based on a deterministic Rabin automaton construction. In $\mathrm{CSL}^{\mathrm{TA}}$, time constraints are specified by *single-clock* DTA. (Note that a similar extension of real-time temporal logic is $\mathrm{TECTL}_{\exists}$ [BLY96].) In [DHS09], $\mathcal{C} \otimes \mathcal{A}$ is interpreted as a Markov renewal process and model checking $\mathrm{CSL}^{\mathrm{TA}}$ is reduced to computing reachability probabilities in a DTMC whose transition probabilities are given by so called subordinate CTMCs. This technique cannot be generalized to multiple clocks. Our approach does not restrict the number of clocks and supports more specifications than $\mathrm{CSL}^{\mathrm{TA}}$ (when combined with CSL as in [DHS09]). For the single-clock case, our approach produces the same result as [DHS09], but yields a conceptually simpler formulation whose correctness can be derived from the simplification of the system of integral equations obtained in the general case. Moreover, measurability has not been addressed in [DHS09]. As for other work, [BBB+07a, BBB+08, BBBM08] provide a quantitative interpretation to timed automata where delays and discrete choices are interpreted probabilistically. In this approach, delays of unbounded clocks are governed by exponential distributions like in CTMCs. Decidability results have been obtained for almost-sure properties [BBB+08] and quantitative verification [BBBM08] for (a subclass of) single-clock timed automata.

Structure of the chapter. Section 7.2 contains definitions of CTMCs, DTA and PDPs. Section 7.3 presents the approach of model checking CTMCs with respect to general DTA in detail. Section 7.4 deals with the special case of a single-clock DTA. Section 7.5 concludes the chapter.

7.2 Preliminaries

For a set H (typically associated with a topology), let $\Pr : \mathcal{F}(H) \to [0,1]$ be a probability measure on the measurable space $(H, \mathcal{F}(H))$, where $\mathcal{F}(H)$ is a σ-algebra over H. Let $\mathit{Distr}(H)$ denote the set of probability measures on this measurable space.

7.2.1 Continuous-time Markov Chains

As stated in Section 1.2, CTMCs can be seen as generalizations of *discrete-time Markov chains* (DTMCs, see Section 2.2.2) by enhancing them with negative exponential state residence time distributions (given by the exit rate function E in the definition below).

Definition 7.2.1 A (labeled) *continuous-time Markov chain* (CTMC) is a tuple $\mathcal{C} = (S, \mathrm{AP}, L, \alpha, \mathbf{P}, E)$ where:

- S is a *finite* set of states;
- AP is a finite set of atomic propositions;

- $L : S \to 2^{\mathrm{AP}}$ is the labeling function;
- $\alpha \in \mathit{Distr}(S)$ is the initial distribution;
- $\mathbf{P} : S \times S \to [0,1]$ is a stochastic transition probability matrix; and
- $E : S \to \mathbb{R}_{\geq 0}$ is the exit rate function.

The probability to exit the state s and to take the transition $s \to s'$ in t time units are

$$\int_0^t E(s) \cdot e^{-E(s)\tau} d\tau \quad \text{and} \quad \mathbf{P}(s,s') \cdot \int_0^t E(s) \cdot e^{-E(s)\tau} d\tau \ ,$$

respectively. A state s is *absorbing* if $\mathbf{P}(s,s) = 1$. The *embedded* DTMC of CTMC $\mathcal{C}$ is obtained by deleting the exit rate function E, i.e., $\mathit{emb}(\mathcal{C}) = (S, \mathrm{AP}, L, \alpha, \mathbf{P})$.

Definition 7.2.2 (Timed path) Let $\mathcal{C} = (S, \mathrm{AP}, L, \alpha, \mathbf{P}, E)$ be a CTMC. An *infinite* path ρ is a sequence $s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} s_2 \cdots$ with, for $i \in \mathbb{N}$, $s_i \in S$ and $t_i \in \mathbb{R}_{>0}$ such that $\mathbf{P}(s_i, s_{i+1}) > 0$ for all i. A *finite* path is a prefix of an infinite path. We write $\mathit{Paths}_n^{\mathcal{C}} := S \times (\mathbb{R}_{>0} \times S)^n$ for the set of (finite) paths of length n in $\mathcal{C}$; the set of *finite* paths in $\mathcal{C}$ is thus defined by $\mathit{Paths}_\star^{\mathcal{C}} = \bigcup_{n \in \mathbb{N}} \mathit{Paths}_n^{\mathcal{C}}$ and $\mathit{Paths}_\omega^{\mathcal{C}} := (S \times \mathbb{R}_{>0})^\omega$ is the set of *infinite* paths in $\mathcal{C}$. $\mathit{Paths}^{\mathcal{C}} = \mathit{Paths}_\star^{\mathcal{C}} \cup \mathit{Paths}_\omega^{\mathcal{C}}$ denotes the set of all paths in $\mathcal{C}$ and $\mathit{Paths}^{\mathcal{C}}(s)$ denotes the set of paths that start from state s. The superscript $\mathcal{C}$ is omitted whenever convenient.

Given any path ρ, we define $|\rho|$ as the number of transitions in ρ if ρ is finite; $|\rho| = \infty$ if ρ is infinite. For $n \leq |\rho|$, $\rho[n] := s_n$ is the $(n+1)$-st state of ρ and $\rho\langle n \rangle := t_n$ is the time spent in state s_n. Let $\rho@t$ be the state occupied in ρ at time $t \in \mathbb{R}_{\geq 0}$, i.e., $\rho@t := \rho[n]$ where n is the smallest index such that $\sum_{i=0}^{n} \rho\langle i \rangle > t$.

The definition of a Borel space on paths through CTMCs follows [BHHK03] (see similar notions in Section 2.2.2 for DTMCs). A CTMC $\mathcal{C}$ with the initial distribution α yields a probability measure $\Pr^{\mathcal{C}}$ on paths as follows: For $s_0, \cdots, s_k \in S$ with $\mathbf{P}(s_i, s_{i+1}) > 0$ for $0 \leq i < k$, and $I_0, \cdots, I_{k-1}$ nonempty intervals in $\mathbb{R}_{\geq 0}$, $C(s_0, I_0, \cdots, I_{k-1}, s_k)$ denotes the *cylinder set* consisting of all paths $\rho \in \mathit{Paths}(s_0)$ such that $\rho[i] = s_i$ $(i \leq k)$ and $\rho\langle i \rangle \in I_i$ $(i < k)$. $\mathcal{F}(\mathit{Paths}(s_0))$ is the smallest σ-algebra on $\mathit{Paths}(s_0)$ which contains all sets $C(s_0, I_0, \cdots, I_{k-1}, s_k)$ for all state sequences $(s_0, \cdots, s_k) \in S^{k+1}$ with $\mathbf{P}(s_i, s_{i+1}) > 0$ $(0 \leq i < k)$ and all sequences $I_0, \cdots, I_{k-1}$ of nonempty intervals in $\mathbb{R}_{\geq 0}$. The probability measure $\Pr^{\mathcal{C}}$ on $\mathcal{F}(\mathit{Paths}(s_0))$ is the unique measure determined by the one on cylinder sets, which is defined by induction on k as $\Pr^{\mathcal{C}}(C(s_0)) = \alpha(s_0)$ and for $k > 0$:

$$\begin{aligned} \Pr^{\mathcal{C}}\big(C(s_0, I_0, \cdots, I_{k-1}, s_k)\big) &= \Pr^{\mathcal{C}}\big(C(s_0, I_0, \cdots, I_{k-2}, s_{k-1})\big) \\ &\cdot \int_{I_{k-1}} \mathbf{P}(s_{k-1}, s_k) E(s_{k-1}) \cdot e^{-E(s_{k-1})\tau} d\tau \ . \end{aligned} \tag{7.1}$$

Example 7.2.3 An example CTMC is illustrated in Fig. 7.1(a) (page 139), where $\mathrm{AP} = \{a, b, c\}$ and s_0 is the initial state, i.e., $\alpha(s_0) = 1$ and $\alpha(s) = 0$ for any $s \neq s_0$. The exit rates r_i and the transition probabilities are as shown.

7.2.2 Deterministic Timed Automata

Definition 7.2.4 A *deterministic timed automaton* (DTA) is a tuple $\mathcal{A} = (\Sigma, \mathcal{X}, Q, q_0, Q_F, \rightarrow)$ where:

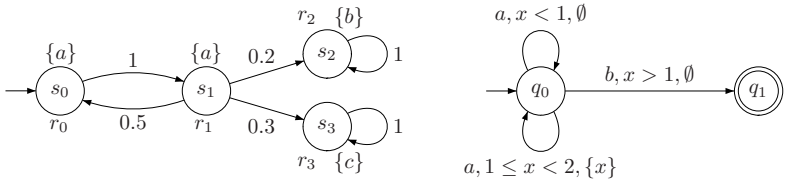
- Σ is a finite *alphabet*;
- $\mathcal{X}$ is a finite set of *clocks*;
- Q is a nonempty finite set of *locations*, with $q_0 \in Q$ the *initial location*;
- $Q_F \subseteq Q$ is a set of *accepting locations*; and
- $\rightarrow \subseteq (Q \setminus Q_F) \times \Sigma \times \mathcal{B}(\mathcal{X}) \times 2^{\mathcal{X}} \times Q$ is a finite *edge relation* satisfying that: (1) g is diagonal-free; and (2) $q \xrightarrow{a,g,X} q'$ and $q \xrightarrow{a,g',X'} q''$ implies $[\![g]\!] \cap [\![g']\!] = \emptyset$ (see Section 2.2.3 for general TA).

As usual, we refer to $q \xrightarrow{a,g,X} q'$ as a *transition*, where $a \in \Sigma$ is the input symbol, the *guard* g is a clock constraint on the clocks of $\mathcal{A}$, $X \subseteq \mathcal{X}$ is a set of clocks to be reset and q' is the successor location. The intuition is that the DTA $\mathcal{A}$ can move from location q to location q' when the input symbol is a and the guard g holds, while the clocks in X should be reset when entering q'. Note that in this chapter, as a convention, we assume that each location $q \in Q_F$ is a sink. An example DTA is shown in Fig. 7.1(b) (page 139).

A (finite) timed path in $\mathcal{A}$ is of the form $\theta = q_0 \xrightarrow{a_0,t_0} q_1 \cdots q_n \xrightarrow{a_n,t_n} q_{n+1}$, for $t_i > 0$ $(0 \leq i \leq n)$. All the definitions on timed paths in CTMCs can be adopted here. A timed path θ of length n (i.e. $|\theta| = n$) is *accepted* by $\mathcal{A}$ if there exists a sequence of clock valuations $\{\eta_j\}_{0 \leq j \leq n}$ such that $\theta[n] \in Q_F$ and for all $0 \leq j < n$, $\eta_0 = \vec{0}$, $\eta_j + t_j \models g_j$ and $\eta_{j+1} = (\eta_j + t_j)[X_j := 0]$, where intuitively η_j is the clock evaluation on *entering* q_j. Given a CTMC $\mathcal{C}$ s.t. $\Sigma = 2^{\mathrm{AP}}$, we say that an infinite timed path $\rho = s_0 \xrightarrow{t_0} s_1 \cdots$ in $\mathcal{C}$ is accepted by $\mathcal{A}$ if there exists some $n \in \mathbb{N}$ such that the finite fragment of ρ, i.e. $s_0 \xrightarrow{t_0} s_1 \cdots s_{n-1} \xrightarrow{t_{n-1}} s_n$ gives rise to an augmented timed path $\hat{\rho} = q_0 \xrightarrow{L(s_0),t_0} q_1 \cdots q_{n-1} \xrightarrow{L(s_{n-1}),t_{n-1}} q_n$, which is accepted by $\mathcal{A}$. Note that here we do not introduce invariants for locations of DTA as in *safety* TA [HNSY94]. However, the obtained result in this chapter can be adapted there without any difficulty.

7.2.3 Piecewise-deterministic Markov Processes

PDPs constitute a general framework that can model virtually any stochastic system without diffusions [Dav84], and for which powerful analysis and control techniques exist [LL85, LY91, CD88]. A PDP consists of a finite set of *locations*, each with a *location invariant* over a set of *variables*. A PDP can jump



(a) CTMC $\mathcal{C}$ (b) DTA $\mathcal{A}$

Figure 7.1: Example CTMC $\mathcal{C}$ and DTA $\mathcal{A}$

between locations either randomly, in which case the residence time of a location is governed by an exponential distribution, or when the location invariant is violated. While staying in a location, a PDP evolves *deterministically* according to a flow function (which is the solution of a system of ODEs). A *state* of the PDP consists of a location and a valuation of the variables. The target state of the jump is determined by a probability measure depending on the source state. The process is *Markovian* as the current state contains all the information to predict the future progress of the process. For a comprehensive exposition of PDPs, we refer the readers to [Dav93].

Let $\mathcal{X} = \{x_1, \ldots, x_n\}$ be a set of variables in $\mathbb{R}$. An $\mathcal{X}$-valuation is a function $\eta : \mathcal{X} \to \mathbb{R}$ assigning to each variable x a value $\eta(x)$. Let $\mathcal{V}(\mathcal{X})$ denote the set of all valuations over $\mathcal{X}$. A *constraint* on $\mathcal{X}$, denoted by g, is a subset of $\mathbb{R}^n$. Let $\mathcal{B}(X)$ denote the set of constrains over $\mathcal{X}$. An $\mathcal{X}$-valuation η *satisfies* constraint g, denoted as $\eta \models g$ if $(\eta(x_1), \cdots, \eta(x_n)) \in g$. Note that here we do not restrict the constraint to a particular form (as in clock constraints), but rather follow a semantical formulation.

Definition 7.2.5 (PDP, [Dav84]) A *piecewise-deterministic (Markov) process* (PDP) is a tuple $\mathcal{Z} = (Z, \mathcal{X}, \mathit{Inv}, \phi, \Lambda, \mu)$ where:

- Z is a finite set of *locations*;
- $\mathcal{X}$ is a finite set of *variables*;
- $\mathit{Inv} : Z \to \mathcal{B}(\mathcal{X})$ is an *invariant function*;
- $\phi : Z \times \mathcal{V}(\mathcal{X}) \times \mathbb{R} \to \mathcal{V}(\mathcal{X})$ is a *flow function*, which is assumed to be the solution of a system of ODEs with a Lipschitz continuous vector field;
- $\Lambda : \mathbb{S} \to \mathbb{R}_{\geq 0}$ is an *exit rate function*; and
- $\mu : \mathbb{S} \cup \partial\mathbb{S} \to \mathit{Distr}(\mathbb{S})$ is a *transition probability function*, where

$\mathbb{S} := \{\xi := (z, \eta) \mid z \in Z, \eta \in \mathit{Inv}(z)\}$ is the state space of the PDP $\mathcal{Z}$, $\mathring{\mathbb{S}}$ is the interior of $\mathbb{S}$ and $\partial\mathbb{S} = \bigcup_{z \in Z} \{z\} \times \partial\mathit{Inv}(z)$ is the boundary of $\mathbb{S}$, with $\partial\mathit{Inv}(z) = \overline{\mathit{Inv}(z)} \setminus \mathring{\mathit{Inv}}(z)$ as the boundary of $\mathit{Inv}(z)$ and $\overline{\mathit{Inv}(z)}$ the closure of $\mathit{Inv}(z)$. Functions Λ and μ satisfy the following conditions:

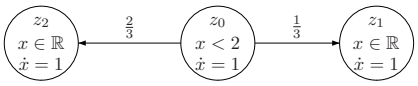


Figure 7.2: An example PDP

- $\forall \xi \in \mathbb{S}.\ \exists \epsilon(\xi) > 0.$ function $t \mapsto \Lambda(\xi \oplus t)$ is integrable on $[0, \epsilon(\xi))$, where $\xi \oplus t = \big(z, \phi(z, \eta, t)\big)$, for $\xi = (z, \eta)$; and
- Function $\xi \mapsto \mu(\xi, A)$[1] is measurable for any $A \in \mathcal{F}(\mathbb{S})$, where $\mathcal{F}(\mathbb{S})$ is a σ-algebra generated by the countable union $\bigcup_{z \in Z} \{z\} \times A_z$ with A_z being a subset of $\mathcal{F}(\mathit{Inv}(z))$ and $\mu(\xi, \{\xi\}) = 0$.

A PDP is only allowed to stay in location z when the constraint $\underline{\mathit{Inv}(z)}$ is satisfied. If e.g., $\mathit{Inv}(z)$ is $x_1^2 - 2x_2 < 1.5 \wedge x_3 > 2$, then its closure $\overline{\mathit{Inv}(z)}$ is $x_1^2 - 2x_2 \leq 1.5 \wedge x_3 \geq 2$, and the boundary $\partial \mathit{Inv}(z)$ is $x_1^2 - 2x_2 = 1.5 \wedge x_3 = 2$. When the variable valuation satisfies the boundary ($\eta \models \partial \mathit{Inv}(z)$), the PDP is forced to jump and leave the current location z. The flow function ϕ defines the time-dependent behavior in a single location, in particular, how the variable valuations change when time elapses. State $\xi \oplus t$ is the timed successor of state ξ (on the same location) given that t time units have passed. The PDP is *piecewise-deterministic* because in each location (one piece) the behavior is deterministically determined by ϕ. In summary, when a new state $\xi = (z, \eta)$ is entered and $\mathit{Inv}(z)$ is valid, i.e., $\xi \in \mathbb{S}$, the PDP can either *delay* to state $\xi' = (z, \eta') \in \mathbb{S} \cup \partial\mathbb{S}$ according to both the flow function ϕ and the time delay t (in this case $\xi' = \xi \oplus t$); or take a *Markovian jump* to state $\xi'' = (z'', \eta'') \in \mathbb{S}$ with probability $\mu(\xi, \{\xi''\})$. Note that the residence time of a location is exponentially distributed. When $\mathit{Inv}(z)$ is invalid, i.e., $\xi \in \partial\mathbb{S}$, ξ will be forced to take a *boundary jump* to ξ'' with probability $\mu(\xi, \{\xi''\})$.

The embedded *discrete-time Markov process* (DTMP) $\mathit{emb}(\mathcal{Z})$ of the PDP $\mathcal{Z}$ has the same state space $\mathbb{S}$ as $\mathcal{Z}$. The (one-jump) *transition probability* from a state ξ to a set $A \subseteq \mathbb{S}$ of states (on different locations as ξ), denoted by $\hat{\mu}(\xi, A)$, is given by [Dav93, CD88]:

$$\hat{\mu}(\xi, A) = \int_0^{\flat(\xi)} (\mathcal{Q}\mathbf{1}_A)(\xi \oplus t) \cdot \Lambda(\xi \oplus t)\, e^{-\int_0^t \Lambda(\xi \oplus \tau) d\tau}\, dt \tag{7.2}$$

$$+ (\mathcal{Q}\mathbf{1}_A)(\xi \oplus \flat(\xi)) \cdot e^{-\int_0^{\flat(\xi)} \Lambda(\xi \oplus \tau) d\tau}\ , \tag{7.3}$$

where $\flat(\xi) = \inf\{t > 0 \,|\, \xi \oplus t \in \partial\mathbb{S}\}$ is the minimal time to hit the boundary if such time exists; $\flat(\xi) = \infty$ otherwise. $(\mathcal{Q}\mathbf{1}_A)(\xi) = \int_{\mathbb{S}} \mathbf{1}_A(\xi') \mu(\xi, d\xi')$ is the accumulative (one-jump) transition probability from ξ to A, and $\mathbf{1}_A(\xi)$ is the characteristic function such that $\mathbf{1}_A(\xi) = 1$ when $\xi \in A$ and $\mathbf{1}_A(\xi) = 0$ otherwise. Term (7.2) specifies the probability to delay to state $\xi \oplus t$ (on the same location) and take a Markovian jump from $\xi \oplus t$ to A. Note that the delay t can take a value from $[0, \flat(\xi))$. Term (7.3) is the probability to stay in the same

[1] $\mu(\xi, A)$ is a shorthand for $(\mu(\xi))(A)$.

location for $\flat(\xi)$ time units and then it is forced to take a boundary jump from $\xi \oplus \flat(\xi)$ to A since $Inv(z)$ is invalid.

Example 7.2.6 Fig. 7.2 depicts a 3-location PDP $\mathcal{Z}$ with one variable x, where $Inv(z_0)$ is $x < 2$ and $Inv(z_1)$, $Inv(z_2)$ are both $x \in [0, \infty)$. Solving $\dot{x} = 1$ gives the flow function $\phi(z_i, \eta(x), t) = \eta(x) + t$ for $i = 0, 1, 2$. The state space of $\mathcal{Z}$ is $\{(z_0, \eta) \mid 0 < \eta(x) < 2\} \cup \{(z_1, \mathbb{R})\} \cup \{(z_2, \mathbb{R})\}$. Let exit rate $\Lambda(\xi) = 5$ for any $\xi \in \mathbb{S}$. For $\eta \models Inv(z_0)$, let $\mu\big((z_0, \eta), \{(z_1, \eta)\}\big) := \frac{1}{3}$, $\mu\big((z_0, \eta), \{(z_2, \eta)\}\big) := \frac{2}{3}$ and the boundary measure $\mu\big((z_0, 2), \{(z_1, 2)\}\big) := 1$. Given state $\xi_0 = (z_0, 0)$ and the set of states $A = (z_1, \mathbb{R})$, the time for ξ_0 to hit the boundary is $\flat(\xi_0) = 2$. Then $(\mathcal{Q}\mathbf{1}_A)(\xi_0 \oplus t) = \frac{1}{3}$ if $t < 2$, and $(\mathcal{Q}\mathbf{1}_A)(\xi_0 \oplus t) = 1$ if $t = 2$. In the embedded DTMP $emb(\mathcal{Z})$, the transition probability from state ξ_0 to A is:

$$\hat{\mu}(\xi_0, A) = \int_0^2 \frac{1}{3} \cdot 5 \cdot e^{-\int_0^t 5 \, d\tau} \, dt + 1 \cdot e^{-\int_0^2 5 \, d\tau} = \frac{1}{3} + \frac{2}{3} e^{-10} \quad .$$

7.3 Model Checking DTA Specifications

In this section, we deal with model checking $\mathcal{C}$ against linear real-time properties specified by DTA $\mathcal{A}$, namley, to compute the probability of $\mathcal{C} \models \mathcal{A}$. We shall prove that this can be reduced to computing the reachability probability in the product of $\mathcal{C}$ and $\mathcal{A}$ (Thm. 7.3.8), which can be further reduced to computing the reachability probability in a corresponding PDP (Thm. 7.3.12). To simplify the notations, we assume, without loss of generality, that a CTMC has only one initial state s_0, i.e., $\alpha(s_0) = 1$, and $\alpha(s) = 0$ for $s \neq s_0$.

7.3.1 Deterministic Markovian Timed Automata

To model check a DTA specification, we will exploit the product of a CTMC and a DTA, which is a *deterministic Markovian timed automaton* (DMTA):

Definition 7.3.1 (DMTA) A *deterministic Markovian timed automaton* is a tuple $\mathcal{M} = (Loc, \mathcal{X}, \ell_0, Loc_F, E, \rightsquigarrow)$, where:

- Loc is a finite set of *locations* with $\ell_0 \in Loc$ the *initial location*;
- $\mathcal{X}$ is a finite set of *clocks*;
- $Loc_F \subseteq Loc$ is the set of *accepting locations*;
- $E : Loc \to \mathbb{R}_{\geq 0}$ is the *exit rate function*; and
- $\rightsquigarrow \subseteq Loc \times \mathcal{B}(\mathcal{X}) \times 2^{\mathcal{X}} \times Distr(Loc)$ is an *edge relation* satisfying that: $(\ell, g, X, \zeta), (\ell, g', X', \zeta') \in \rightsquigarrow$ implies $[\![g]\!] \cap [\![g']\!] = \emptyset$.

The set of clocks $\mathcal{X}$ and the related concepts, e.g., clock valuation, clock constraints are defined as for TA (see Section 2.2.3). We refer to $\ell \overset{g,X}{\rightsquigarrow} \zeta$ for

distribution $\zeta \in \mathit{Distr}(\mathit{Loc})$ as an *edge* and refer to $\ell \xmapsto[\zeta(\ell')]{g,X} \ell'$ as a *transition* of this edge. The intuition is that when entering location ℓ, the DMTA chooses a residence time which is governed by the exponential distribution, i.e., the probability to leave ℓ in t time units is $1 - e^{-E(\ell)t}$. When it decides to jump, at most one edge, say $\ell \overset{g,X}{\rightsquigarrow} \zeta$, due to the determinism, is enabled and the probability to jump to ℓ' is given by $\zeta(\ell')$. The DMTA is *deterministic* as it has a unique initial location and disjoint guards for all edges emanating from any location.

Example 7.3.2 The DMTA in Fig. 7.3(a) has initial location ℓ_0 with two edges, with guards $x < 1$ and $1 < x < 2$. Assume t time units elapsed. If $t < 1$, then the upper edge is enabled and the probability to go to ℓ_1 in time t is $(1 - e^{-r_0 t})\cdot 1$, where $E(\ell_0) = r_0$; no clock is reset. The process is similar for $1 < t < 2$, except that x will be reset. Location ℓ_3 is accepting.

Paths in DMTAs. Given a DMTA $\mathcal{M}$ and a finite *symbolic* path

$$\ell_0 \xmapsto[p_0]{g_0,X_0} \ell_1 \ \cdots \ \ell_{n-1} \xmapsto[p_{n-1}]{g_{n-1},X_{n-1}} \ell_n \quad ,$$

where $p_i = \zeta_i(\ell_{i+1})$ is the transition probability of $\ell_i \xmapsto[\zeta_i(\ell_{i+1})]{g_i,X_i} \ell_{i+1}$, the induced finite paths in $\mathcal{M}$ are of the form $\sigma = \ell_0 \xrightarrow{t_0} \ell_1 \cdots \ell_{n-1} \xrightarrow{t_{n-1}} \ell_n$ and have the property that $\eta_0 = \vec{0}$, $(\eta_i + t_i) \models g_i$, and $\eta_{i+1} = (\eta_i + t_i)[X_i := 0]$ where $0 \le i < n$ and η_i is the clock valuation of $\mathcal{X}$ in $\mathcal{M}$ on *entering* location ℓ_i. Finite path σ is *accepting* if $\ell_n \in \mathit{Loc}_F$. All the definitions on paths in CTMCs can be carried over to paths in DMTA.

Given a DMTA $\mathcal{M}$, the cylinder set $C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n)$ where $(\ell_0, \cdots, \ell_n) \in \mathit{Loc}^{n+1}$ and $I_i \subseteq \mathbb{R}_{\ge 0}$ denotes a set of paths σ in $\mathcal{M}$ such that $\sigma[i] = \ell_i$ and $\sigma\langle i \rangle \in I_i$. Now we define the measure $\Pr_\eta^{\mathcal{M}}$, which is the probability of the cylinder set $C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n)$ such that the initial clock valuation in location ℓ_0 is η_0 as

$$\Pr_{\eta_0}^{\mathcal{M}}\left(C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n)\right) := \mathbb{P}_0^{\mathcal{M}}(\eta_0) \ .$$

Here $\mathbb{P}_i^{\mathcal{M}}(\cdot)$ for $0 \le i \le n$ is defined as: $\mathbb{P}_n^{\mathcal{M}}(\eta) = 1$, and for $0 \le i < n$, we note that there exists a transition from ℓ_i to ℓ_{i+1} with $\ell_i \xmapsto[p_i]{g_i,X_i} \ell_{i+1}$ $(0 \le i < n)$, and thus we define

$$\mathbb{P}_i^{\mathcal{M}}(\eta) = \int_{I_i} \underbrace{\mathbf{1}_{g_i}(\eta+\tau)\cdot p_i \cdot E(\ell_i)\cdot e^{-E(\ell_i)\tau}}_{(\star)} \cdot \underbrace{\mathbb{P}_{i+1}^{\mathcal{M}}(\eta')}_{(\star\star)} \, d\tau \ , \tag{7.4}$$

where $\eta' := (\eta + \tau)[X_i := 0]$ and the *characteristic function* $\mathbf{1}_{g_i}(\eta + \tau) = 1$, if $\eta + \tau \models g_i$; 0, otherwise.

Intuitively, $\mathbb{P}_i^{\mathcal{M}}(\eta_i)$ is the probability of the suffix cylinder set starting from ℓ_i and η_i to ℓ_n. It is recursively computed by the product of the probability of taking a transition from ℓ_i to ℓ_{i+1} in time interval I_i (cf. $(\star)$) and the probability of the suffix cylinder set from ℓ_{i+1} and η_{i+1} on (cf. $(\star\star)$), where $(\star)$ is computed by first ruling out the paths that do not belong to the cylinder set by $\mathbf{1}_{g_i}(\eta+\tau)$ and then computing the transition probability using the density function $p_i{\cdot}E(\ell_i){\cdot}e^{-E(\ell_i)\tau}$ as in CTMCs. The characteristic function is Riemann integrable, as it is bounded and its support is an interval, and thus $\mathbb{P}_i^{\mathcal{M}}(\eta)$ is well-defined.

7.3.2 Product DMTAs

Given a CTMC $\mathcal{C}$ and a DTA $\mathcal{A}$, the product $\mathcal{C}\otimes\mathcal{A}$ is a DMTA defined by:

Definition 7.3.3 (Product of CTMC & DTA) Let $\mathcal{C}=(S,\mathrm{AP},L,s_0,\mathbf{P},E)$ be a CTMC and $\mathcal{A}=(2^{\mathrm{AP}},\mathcal{X},Q,q_0,Q_F,\rightarrow)$ be a DTA. We define $\mathcal{C}\otimes\mathcal{A}=(\mathit{Loc},\mathcal{X},\ell_0,\mathit{Loc}_F,E,\rightsquigarrow)$ as the *product* DMTA, where $\mathit{Loc}:=S\times Q$; $\ell_0:=\langle s_0,q_0\rangle$; $\mathit{Loc}_F:=S\times Q_F$; $E(\langle s,q\rangle):=E(s)$; and $\rightsquigarrow$ is defined as the relation satisfying the following rule:

$$\frac{\mathbf{P}(s,s')>0\ \wedge\ q\xrightarrow{L(s),g,X}q'}{\langle s,q\rangle\overset{g,X}{\rightsquigarrow}\zeta},\ \text{s.t. }\zeta(\langle s',q'\rangle)=\mathbf{P}(s,s')\ .$$

Example 7.3.4 Let CTMC $\mathcal{C}$ be in Fig. 7.1(a) and DTA $\mathcal{A}$ be in Fig. 7.1(b) (page 139). The product DMTA $\mathcal{C}\otimes\mathcal{A}$ is depicted as in Fig. 7.3(a) (page 144).

Remark 7.3.5 It is easy to see from the construction that $\mathcal{C}\otimes\mathcal{A}$ is indeed a DMTA. The determinism of the DTA $\mathcal{A}$ guarantees that the induced product is also deterministic. In $\mathcal{C}\otimes\mathcal{A}$, there is at most one "action" possible, viz. $L(s)$, from each location $\ell=\langle s,q\rangle$, probably via different edges, but with disjoint guards. We can thus omit it from the product DMTA.

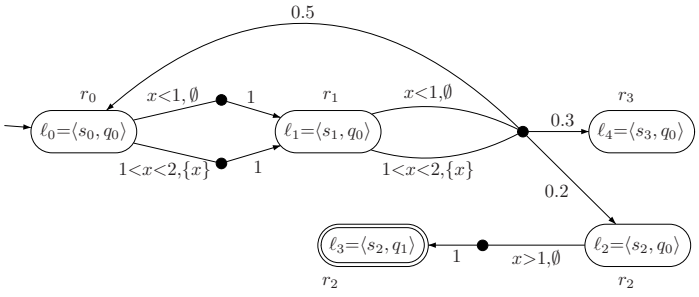
We denote

$$\mathit{Paths}^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\,\mathit{Loc}_F):=\{\sigma\in\mathit{Paths}_\star^{\mathcal{C}\otimes\mathcal{A}}\mid\sigma\text{ is accepted by }\mathcal{C}\otimes\mathcal{A}\}$$

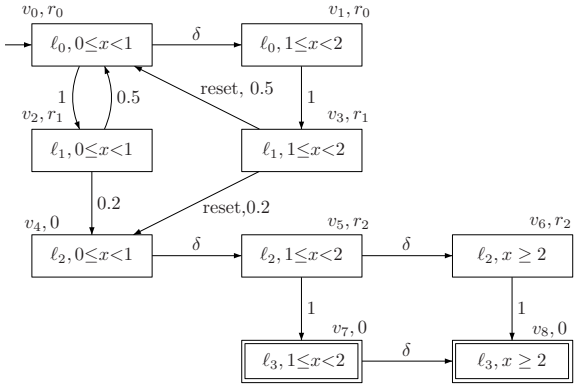
as the set of accepted paths in $\mathcal{C}\otimes\mathcal{A}$, and

$$\mathit{Paths}^{\mathcal{C}}(\mathcal{A}):=\{\rho\in\mathit{Paths}_\star^{\mathcal{C}}\mid\rho\text{ is accepted by DTA }\mathcal{A}\}$$

as the set of paths in CTMC $\mathcal{C}$ that are accepted by DTA $\mathcal{A}$. For any n-ary tuple J, let $J\!\downarrow_i$ denote the i-th entry in J, for $1\le i\le n$. For a $\mathcal{C}\otimes\mathcal{A}$ path $\sigma=\langle s_0,q_0\rangle\xrightarrow{t_0}\cdots\xrightarrow{t_{n-1}}\langle s_n,q_n\rangle$, let $\sigma\!\downarrow_1:=s_0\xrightarrow{t_0}\cdots\xrightarrow{t_{n-1}}s_n$, and for any set Π of $\mathcal{C}\otimes\mathcal{A}$ paths, let $\Pi\!\downarrow_1=\bigcup_{\sigma\in\Pi}\sigma\!\downarrow_1$.



(a) DMTA $\mathcal{M} = \mathcal{C} \otimes \mathcal{A}$



(b) Reachable region graph

Figure 7.3: Example product construction of CTMC and DTA

Lemma 7.3.6 Given any CTMC $\mathcal{C}$ and DTA $\mathcal{A}$, we have that $Paths^{\mathcal{C}}(\mathcal{A}) = Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)\downarrow_1$.

Proof: ($\Longrightarrow$) We show that for any path $\rho \in Paths^{\mathcal{C}}(\mathcal{A})$, there exists a path $\sigma \in Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)$ such that $\sigma\downarrow_1 = \rho$.

We assume, without loss of generality, that $\rho = s_0 \xrightarrow{t_0} s_1 \cdots s_{n-1} \xrightarrow{t_{n-1}} s_n \in Paths^{\mathcal{C}}$ is accepted by $\mathcal{A}$, i.e., $s_n \in Q_F$ and for $0 \le i < n$, $\eta_0 \models \vec{0}$ and $\eta_i + t_i \models g_i$ and $\eta_{i+1} = (\eta_i + t_i)[X_i := 0]$, where η_i is the time valuation on entering state s_i. We can then construct a path $\theta \in Paths^{\mathcal{A}}$ from ρ such that $\theta = q_0 \xrightarrow{L(s_0),t_0} q_1 \cdots q_{n-1} \xrightarrow{L(s_{n-1}),t_{n-1}} q_n$, where s_i and q_i have the same entering clock valuation. From ρ and θ, we can construct the path

$$\sigma = \langle s_0, q_0\rangle \xrightarrow{t_0} \langle s_1, q_1\rangle \cdots \langle s_{n-1}, q_{n-1}\rangle \xrightarrow{t_{n-1}} \langle s_n, q_n\rangle \ ,$$

where $\langle s_n, q_n\rangle \in Loc_F$. It follows from the definition of an accepting path in a DTMA that $\sigma \in Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)$ and $\sigma\downarrow_1 = \rho$.

($\Longleftarrow$) We show that for any path $\sigma \in Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)$, $\sigma\downarrow_1 \in Paths^{\mathcal{C}}(\mathcal{A})$.

We assume, without loss of generality, that for path

$$\sigma = \langle s_0, q_0\rangle \xrightarrow{t_0} \cdots \xrightarrow{t_{n-1}} \langle s_n, q_n\rangle \in Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)\ ,$$

it holds that $\langle s_n, q_n\rangle \in Loc_F$, and for $0 \leq i < n$, $\eta_0 \models \vec{0}$ and $\eta_i + t_i \models g_i$ and $\eta_{i+1} = (\eta_i + t_i)[X_i := 0]$, where η_i is the time valuation on entering state $\langle s_i, q_i\rangle$. It then directly follows that $q_n \in Q_F$ and $\sigma\downarrow_1 \in Paths^{\mathcal{C}}(\mathcal{A})$, given η_i the entering clock valuation of state s_i. ■

Theorem 7.3.7 For any CTMC $\mathcal{C}$ and DTA $\mathcal{A}$, $Paths^{\mathcal{C}}(\mathcal{A})$ is measurable.

Proof: We first deal with the case that $\mathcal{A}$ contains only *strict* inequalities. Since $Paths^{\mathcal{C}}(\mathcal{A})$ is a set of *finite* paths,

$$Paths^{\mathcal{C}}(\mathcal{A}) = \bigcup_{n\in\mathbb{N}} Paths^{\mathcal{C}}_n(\mathcal{A})\ ,$$

where $Paths^{\mathcal{C}}_n(\mathcal{A})$ is the set of paths of $\mathcal{C}$ accepted by $\mathcal{A}$ of length n. For any path $\rho{:=}s_0 \xrightarrow{t_0} s_1 \cdots s_{n-1} \xrightarrow{t_{n-1}} s_n \in Paths^{\mathcal{C}}_n(\mathcal{A})$, we can associate ρ with a path $\theta{:=}q_0 \xrightarrow{L(s_0),t_0} q_1 \cdots q_{n-1} \xrightarrow{L(s_{n-1}),t_{n-1}} q_n$ of $\mathcal{A}$ induced by the location sequence:

$$q_0 \xrightarrow{L(s_0),g_0,X_0} q_1 \cdots q_{n-1} \xrightarrow{L(s_{n-1}),g_{n-1},X_{n-1}} q_n\ ,$$

such that $q_n \in Q_F$ and there exist $\{\eta_i\}_{0\leq i<n}$ with 1) $\eta_0 = \vec{0}$; 2) $(\eta_i + t_i) \models g_i$; and 3) $\eta_{i+1} = (\eta_i + t_i)[X_i := 0]$, where η_i is the clock valuation on entering q_i.

To prove the measurability of $Paths^{\mathcal{C}}_n(\mathcal{A})$, it suffices to show that for each path $\rho := s_0 \xrightarrow{t_0} \cdots \xrightarrow{t_{n-1}} s_n \in Paths^{\mathcal{C}}_n(\mathcal{A})$, there exists a cylinder set $C(s_0, I_0, \cdots, I_{n-1}, s_n)$ (C_ρ for short) that contains ρ, and that each path in C_ρ is accepted by $\mathcal{A}$. The interval I_i is constructed according to t_i as $I_i = [t_i^-, t_i^+]$ such that

- If $t_i \in \mathbb{Q}$, then $t_i^- = t_i^+ := t_i$;
- else if $t_i \in \mathbb{R}\setminus\mathbb{Q}$, then let $t_i^-, t_i^+ \in \mathbb{Q}$ such that
  - $t_i^- \leq t_i \leq t_i^+$ and $\lfloor t_i^-\rfloor = \lfloor t_i\rfloor$ and $\lceil t_i^+\rceil = \lceil t_i\rceil$;
  - $t_i^+ - t_i^- < \dfrac{\Delta}{2\cdot n}$, where

$$\Delta = \min_{0\leq j<n, x\in\mathcal{X}} \Big\{\{\eta_j(x) + t_j\}, 1 - \{\eta_j(x) + t_j\} \;\Big|\; \{\eta_j(x) + t_j\} \neq 0\Big\}.^{2}$$

[2] $\{\cdot\}$ denotes the fractional part.

Note that here we are considering DTA with *strict* inequalities. Hence for any $0 \leq i < n$ with $\eta_i + t_i \models g_i$, it must be the case that $\{\eta_i(x) + t_i\} \neq 0$.

To show that $\rho' := s_0 \xrightarrow{t'_0} \cdots \xrightarrow{t'_{n-1}} s_n \in C_\rho$ is accepted by $\mathcal{A}$, let $\eta'_0 := \vec{0}$ and $\eta'_{i+1} := (\eta'_i + t'_i)[X_i := 0]$. We will show that $\eta'_i + t'_i \models g_i$. To this end, it suffices to observe that $\eta'_0 = \eta_0$, and for any $i > 0$ and any clock variable x,

$$\left|\eta'_i(x) - \eta_i(x)\right| \leq \sum_{j=0}^{i-1} \left|t'_j - t_j\right| \leq \sum_{j=0}^{i-1} t_j^+ - t_j^- \leq n \cdot (t_j^+ - t_j^-) \leq \frac{\Delta}{2} \ .$$

We claim that since DTA $\mathcal{A}$ contains only strict inequalities, it must be the case that $\eta'_i + t'_i \models g_i$. To see this, suppose g_i is of the form $x > K$ for some $K \in \mathbb{N}$. We have that $|\eta'_i(x) - \eta_i(x)| \leq \frac{\Delta}{2}$ and $|t'_i - t_i| < \frac{\Delta}{2}$, therefore $|(\eta'_i(x) + t'_i) - (\eta_i(x) + t_i)| < \Delta$. Note that $\eta_i(x) + t_i > K$, and thus $\eta_i(x) + t_i - \{\eta_i(x) + t_i\} = \lceil \eta_i(x) + t_i \rceil \geq K$. Hence $\eta_i(x) + t_i - \Delta \geq K$ since $\Delta \leq \{\eta_i(x) + t_i\}$. It follows that $\eta'_i(x) + t'_i > K$. A similar argument applies to the case $x < K$ and can be extended to any constraint g_i. Thus, $\eta'_i + t'_i \models g_i$.

It follows that C_ρ is a cylinder set of $\mathcal{C}$ and each path in this cylinder set is accepted by $\mathcal{A}$, i.e., $\rho \in C_\rho$ and $C_\rho \subseteq Paths^{\mathcal{C}}_n(\mathcal{A})$ with $|\rho| = n$. Together with the fact that $Paths^{\mathcal{C}}_n(\mathcal{A}) \subseteq \bigcup_{\rho \in Paths^{\mathcal{C}}_n(\mathcal{A})} C_\rho$, we have:

$$Paths^{\mathcal{C}}_n(\mathcal{A}) = \bigcup_{\rho \in Paths^{\mathcal{C}}_n(\mathcal{A})} C_\rho$$

and thus

$$Paths^{\mathcal{C}}(\mathcal{A}) = \bigcup_{n \in \mathbb{N}} \bigcup_{\rho \in Paths^{\mathcal{C}}_n(\mathcal{A})} C_\rho \ .$$

We note that each interval in the cylinder set C_ρ has *rational* bounds. Hence C_ρ is measurable. It follows that $Paths^{\mathcal{C}}(\mathcal{A})$ is a union of *countably many* cylinder sets, and thus is measurable.

We then deal with $\mathcal{A}$ with equalities of the form $x = n$ for $n \in \mathbb{N}$. We prove the measurability by induction on the number of such equalities appearing in $\mathcal{A}$. We have shown the base case (DTAs with only strict inequalities). Now we focus on the inductive case. Suppose there exists a transition $\iota = q \xrightarrow{a,g,X} q'$ where g contains $x = n$. We first consider a DTA $\mathcal{A}_\iota$ obtained from $\mathcal{A}$ by deleting the transitions from q other than ι. We then consider three DTA $\bar{\mathcal{A}}_\iota$, $\mathcal{A}^>_\iota$ and $\mathcal{A}^<_\iota$ where $\bar{\mathcal{A}}_\iota$ is obtained from $\mathcal{A}_\iota$ by replacing $x = n$ by *true*; $\mathcal{A}^>_\iota$ is obtained from $\mathcal{A}_\iota$ by replacing $x = n$ by $x > n$ and $\mathcal{A}^<_\iota$ is obtained from $\mathcal{A}_\iota$ by replacing $x = n$ by $x < n$. It is not difficult to see that

$$Paths^{\mathcal{C}}(\mathcal{A}_\iota) = Paths^{\mathcal{C}}(\bar{\mathcal{A}}_\iota) \setminus (Paths^{\mathcal{C}}(\mathcal{A}^>_\iota) \cup Paths^{\mathcal{C}}(\mathcal{A}^<_\iota)) \ .$$

Note that this holds since $\mathcal{A}$ is deterministic. By the induction hypothesis, $Paths^{\mathcal{C}}(\bar{\mathcal{A}}_\iota)$, $Paths^{\mathcal{C}}(\mathcal{A}^>_\iota)$ and $Paths^{\mathcal{C}}(\mathcal{A}^<_\iota)$ are measurable. Hence $Paths^{\mathcal{C}}(\mathcal{A}_\iota)$

is measurable. Furthermore, we note that

$$Paths^{\mathcal{C}}(\mathcal{A}) = \bigcup_{\iota = q \xrightarrow{a,g,X} q'} Paths^{\mathcal{C}}(\mathcal{A}_\iota) \ ,$$

therefore $Paths^{\mathcal{C}}(\mathcal{A})$ is measurable as well.

For arbitrary $\mathcal{A}$ with time constraints of the form $x \bowtie n$ where $\bowtie \in \{\geq, \leq\}$, we consider two DTAs $\mathcal{A}_=$ and $\mathcal{A}_{\bowtie}$. Clearly

$$Paths^{\mathcal{C}}(\mathcal{A}) = Paths^{\mathcal{C}}(\mathcal{A}_=) \cup Paths^{\mathcal{C}}(\mathcal{A}_{\bar{\bowtie}}) \ ,$$

where $\bar{\bowtie} = >$ if $\bowtie = \geq$; $<$ otherwise. It follows that $Paths^{\mathcal{C}}(\mathcal{A})$ is measurable. ■

We remark that the set of time-convergent paths in a CTMC has probability measure 0 (see [BHHK03]). The following theorem establishes the link between CTMC $\mathcal{C}$ and DMTA $\mathcal{C} \otimes \mathcal{A}$.

Theorem 7.3.8 For any CTMC $\mathcal{C}$ and DTA $\mathcal{A}$,

$$\Pr^{\mathcal{C}}\Big(Paths^{\mathcal{C}}(\mathcal{A})\Big) = \Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\Big(Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond Loc_F)\Big) \ .$$

Proof: According to Thm. 7.3.7, $Paths^{\mathcal{C}}(\mathcal{A})$ can be rewritten as the combination of cylinder sets of the form $C(s_0, I_0, \cdots, I_{n-1}, s_n)$ which are all accepted by DTA $\mathcal{A}$.[3] By Lem. 7.3.6, namely by path lifting, we can establish exactly the same combination of cylinder sets $C(\ell_0, I_0, \cdots, I_{n-1}, \ell_0)$ for $Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond Loc_F)$, where $s_i = \ell_i\downarrow_1$. It then suffices to show that for each cylinder set $C(s_0, I_0, \cdots, I_{n-1}, s_n)$ which is accepted by $\mathcal{A}$, $\Pr^{\mathcal{C}}$ and $\Pr^{\mathcal{C}\otimes\mathcal{A}}$ yield the same probabilities. Note that a cylinder set C is accepted by a DTA $\mathcal{A}$, if each path that C generates can be accepted by $\mathcal{A}$.

For the measure $\Pr^{\mathcal{C}}$, according to Eq. (7.1),

$$\Pr^{\mathcal{C}}\big(C(s_0, I_0, \cdots, I_{n-1}, s_n)\big) = \prod_{0\leq i<n} \int_{I_i} \mathbf{P}(s_i, s_{i+1}) \cdot E(s_i) e^{-E(s_i)\tau} d\tau \ .$$

For the measure $\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}$, according to Section 7.3.1, it is given by $\mathbb{P}_0^{\mathcal{C}\otimes\mathcal{A}}(\vec{0})$ where $\mathbb{P}_n^{\mathcal{C}\otimes\mathcal{A}}(\eta) = 1$ for any clock valuation η and

$$\mathbb{P}_i^{\mathcal{C}\otimes\mathcal{A}}(\eta_i) = \int_{I_i} \mathbf{1}_{g_i}(\eta_i + \tau_i) p_i E(\ell_i) e^{-E(\ell_i)\tau_i} \cdot \mathbb{P}_{i+1}^{\mathcal{C}\otimes\mathcal{A}}(\eta_{i+1}) \ d\tau_i \ ,$$

where $\eta_{i+1} = (\eta_i + \tau_i)[X_i := 0]$ and $\mathbf{1}_{g_i}(\eta_i + \tau_i) = 1$, if $\eta_i + \tau_i \models g_i$; 0, otherwise.

We will show, by induction, that $\mathbb{P}_i^{\mathcal{C}\otimes\mathcal{A}}(\eta_i)$ is a constant, i.e., is independent of η_i, if the cylinder set $C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n)$ is accepted by $\mathcal{C} \otimes \mathcal{A}$. Firstly

[3] Note that this means each path in the cylinder set is accepted by $\mathcal{A}$.

let us note that for $C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n)$, there must exist some sequence of transitions

$$\ell_0 \xrightarrow[p_0]{g_0, X_0} \ell_1 \ \cdots \ \ell_{n-1} \xrightarrow[p_{n-1}]{g_{n-1}, X_{n-1}} \ell_n$$

with $\eta_0 = \vec{0}$ and $\forall t_i \in I_i$ with $0 \leq i < n$, $\eta_i + t_i \models g_i$ and $\eta_{i+1} := (\eta_i + t_i)[X_i := 0]$. Moreover, according to Def. 7.3.3, we have:

$$p_i = \mathbf{P}(s_i, s_{i+1}) \qquad \text{and} \qquad E(\ell_i) = E(s_i) \ . \tag{7.5}$$

We apply a backward induction on n down to 0. The base case is trivial since $\mathbb{P}_n^{\mathcal{C}\otimes\mathcal{A}}(\eta) = 1$. By the induction hypothesis, $\mathbb{P}_{i+1}^{\mathcal{C}\otimes\mathcal{A}}(\eta)$ is a constant. For the inductive case, consider $i < n$. For any $\tau_i \in I_i$, since $\eta_i + \tau_i \models g_i$, $\mathbf{1}_{g_i}(\eta_i + \tau_i) = 1$, it follows that

$$\begin{aligned}
\mathbb{P}_i^{\mathcal{C}\otimes\mathcal{A}}(\eta_i) &= \int_{I_i} \mathbf{1}_{g_i}(\eta_i + \tau_i) p_i E(\ell_i) e^{-E(\ell_i)\tau_i} \cdot \mathbb{P}_{i+1}^{\mathcal{C}\otimes\mathcal{A}}(\eta_{i+1}) \ d\tau_i \\
&\overset{\text{I.H.}}{=} \int_{I_i} p_i E(\ell_i) e^{-E(\ell_i)\tau_i} d\tau_i \cdot \mathbb{P}_{i+1}^{\mathcal{C}\otimes\mathcal{A}}(\eta_{i+1}) \\
&\overset{(7.5)}{=} \int_{I_i} \mathbf{P}(s_i, s_{i+1}) E(s_i) e^{-E(s_i)\tau_i} d\tau_i \cdot \mathbb{P}_{i+1}^{\mathcal{C}\otimes\mathcal{A}}(\eta_{i+1}) \ .
\end{aligned}$$

Clearly, this is a constant. It is thus easy to see that

$$\begin{aligned}
&\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\big(C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n)\big) := \\
&\qquad \mathbb{P}_0^{\mathcal{C}\otimes\mathcal{A}}(\vec{0}) = \prod_{0 \leq i < n} \int_{I_i} \mathbf{P}(s_i, s_{i+1}) E(s_i) e^{-E(s_i)\tau} d\tau \ ,
\end{aligned}$$

which completes the proof. ■

7.3.3 Region Construction for DMTA

In the remainder of this section, we shall focus on how to compute the probability measure $\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\left(Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)\right)$ in an effective way. We start with adopting the standard *region construction* [AD94] to DMTA. As we will see, this allows us to obtain a PDP from a DMTA in a natural way.

As usual, a region is represented as a (clock) constraint. For regions $\Theta, \Theta' \in \mathcal{B}(\mathcal{X})$, Θ' is the *successor region* of Θ if for all $\eta \models \Theta$ there exists $\delta \in \mathbb{R}_{>0}$ such that $\eta + \delta \models \Theta'$ and for all $\delta' < \delta$, $\eta + \delta' \models \Theta \vee \Theta'$. A region Θ *satisfies* a guard g (denoted by $\Theta \models g$) iff $\forall \eta \models \Theta.\ \eta \models g$. A *reset operation* on region Θ is defined as $\Theta[X := 0] := \{\eta[X := 0] \mid \eta \models \Theta\}$.

Definition 7.3.9 (Region graph of DMTA) Given DMTA $\mathcal{M} = (Loc, \mathcal{X}, \ell_0, Loc_F, E, \rightsquigarrow)$, the region graph $\mathcal{G}(\mathcal{M})$ is defined as a tuple $(V, v_0, V_F, \Lambda, \hookrightarrow)$, where:

- $V := Loc \times \mathcal{B}(\mathcal{X})$ is a finite set of *vertices*, consisting of a location ℓ in $\mathcal{M}$ and a region Θ;
- $v_0 \in V$ is the *initial vertex* if $(\ell_0, \vec{0}) \in v_0$;
- $V_F := \{v \mid v\lfloor_1 \in Loc_F\}$ is the set of *accepting vertices*;
- $\hookrightarrow \subseteq V \times \big(([0,1] \times 2^{\mathcal{X}}) \cup \{\delta\}\big) \times V$ is the *transition (edge) relation*, such that:
  - $v \overset{\delta}{\hookrightarrow} v'$ is a delay transition if $v\lfloor_1 = v'\lfloor_1$ and $v'\lfloor_2$ is a successor region of $v\lfloor_2$; and
  - $v \overset{p,X}{\hookrightarrow} v'$ is a Markovian transition if there is a transition $v\lfloor_1 \overset{g,X}{\underset{p}{\longmapsto}} v'\lfloor_1$ in $\mathcal{M}$ such that $v\lfloor_2 \models g$ and $v\lfloor_2[X := 0] \models v'\lfloor_2$.
- $\Lambda : V \to \mathbb{R}_{\geq 0}$ is the *exit rate function* where $\Lambda(v) := E(v\lfloor_1)$ if there exists a Markovian transition from v, 0 otherwise;

Note that in the obtained region graph, Markovian transitions emanating from any boundary region do *not* contribute to the reachability probability as the time to hit the boundary is always zero (cf. Eq.(7.7)). Therefore, we can remove all the Markovian transitions emanating from boundary regions and then collapse each of them with its unique *non-boundary* (direct) successor. In the sequel we still denote this *collapsed* region graph $\mathcal{G}(\mathcal{M})$ by slightly abusing the notation.

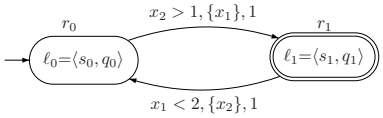
Example 7.3.10 For the DMTA $\mathcal{C}\otimes\mathcal{A}$ in Fig. 7.4(a), the reachable part (forward reachable from the initial vertex and backward reachable from the accepting vertices) of the collapsed region graph $\mathcal{G}(\mathcal{C}\otimes\mathcal{A})$ is in Fig. 7.4(b). The accepting vertices are sinks.

We now define the underlying PDP of a DMTA based on the region graph $\mathcal{G}(\mathcal{M})$ given above:

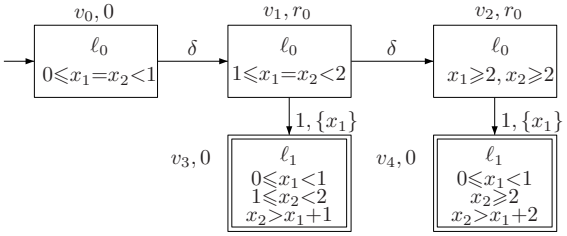
Definition 7.3.11 For DMTA $\mathcal{M} = (Loc, \mathcal{X}, \ell_0, Loc_F, E, \rightsquigarrow)$ and region graph $\mathcal{G}(\mathcal{M}) = (V, v_0, V_F, \Lambda, \hookrightarrow)$, we define PDP $\mathcal{Z}(\mathcal{M}) = (V, \mathcal{X}, Inv, \phi, \Lambda, \mu)$ where for any $v \in V$,

- $Inv(v) := v\lfloor_2$ and the state space $\mathbb{S} := \{(v, \eta) \mid v \in V, \eta \in Inv(v)\}$;
- $\phi(v, \eta, t) := \eta + t$ for $\eta \in Inv(v)$;
- $\Lambda(v, \eta) := \Lambda(v)$ is the exit rate of state (v, η);
- [*boundary jump*] for each delay transition $v \overset{\delta}{\hookrightarrow} v'$ in $\mathcal{G}(\mathcal{M})$ we have $\mu(\xi, \{\xi'\}) := 1$, where $\xi = (v, \eta)$, $\xi' = (v', \eta)$ and $\eta \in \partial Inv(v)$; and
- [*Markovian jump*] for each Markovian transition $v \overset{p,X}{\hookrightarrow} v'$ in $\mathcal{G}(\mathcal{M})$ we have $\mu(\xi, \{\xi'\}) := p$, where $\xi = (v, \eta)$, $\eta \in Inv(v)$ and $\xi' = (v', \eta[X := 0])$.

From now on we write $\Lambda(v)$ instead of $\Lambda(v, \eta)$ as they coincide.



(a) DMTA $\mathcal{M} = \mathcal{C} \otimes \mathcal{A}$



(b) Reachable region graph $\mathcal{G}(\mathcal{C} \otimes \mathcal{A})$

Figure 7.4: Example of a region graph

7.3.4 Characterizing Reachability Probabilities

Recall that our task is to compute $\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\Big(Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)\Big)$, owning to Thm. 7.3.8. To this end, we now reduce it to the problem of computing the *(time-unbounded) reachability probability* in the PDP $\mathcal{Z}(\mathcal{C} \otimes \mathcal{A})$, given the initial state $(v_0, \vec{0})$ and the set of goal states $\{(v,\eta) \mid v \in V_F, \eta \in Inv(v)\}$ $\big((V_F,\cdot)$ for short$\big)$. It is not difficult to see that reachability probabilities of untimed events in a PDP $\mathcal{Z}$ can be computed in the embedded DTMP $emb(\mathcal{Z})$. Note that the set of locations of $\mathcal{Z}$ and $emb(\mathcal{Z})$ are equal. In the sequel, let $\mathcal{D}$ denote $emb(\mathcal{Z})$.

For each vertex $v \in V$, we define recursively $Prob^{\mathcal{D}}\big((v,\eta),(V_F,\cdot)\big)$ $\big(Prob_v^{\mathcal{D}}(\eta)$ for short$\big)$ as the probability to reach the goal states $(V_F,\cdot)$ in $\mathcal{D}$ from state (v,η). Firstly, we define

- For the delay transition $v \overset{\delta}{\hookrightarrow} v'$,

$$Prob_{v,\delta}^{\mathcal{D}}(\eta) = e^{-\Lambda(v)\flat(v,\eta)} \cdot Prob_{v'}^{\mathcal{D}}\big(\eta + \flat(v,\eta)\big)\ . \tag{7.6}$$

  Recall that $\flat(v,\eta)$ is the minimal time for (v,η) to hit the boundary $\partial Inv(v)$.

- For the Markovian transition $v \overset{p,X}{\hookrightarrow} v'$,

$$Prob_{v,v'}^{\mathcal{D}}(\eta) = \int_0^{\flat(v,\eta)} p{\cdot}\Lambda(v){\cdot}e^{-\Lambda(v)\tau}{\cdot}Prob_{v'}^{\mathcal{D}}\big((\eta+\tau)[X:=0]\big)\ d\tau\ . \tag{7.7}$$

Overall, for each vertex $v \in V$, we obtain:

$$Prob_v^{\mathcal{D}}(\eta) = \begin{cases} Prob_{v,\delta}^{\mathcal{D}}(\eta) + \sum_{v \overset{p,X}{\hookrightarrow} v'} Prob_{v,v'}^{\mathcal{D}}(\eta), & \text{if } v \notin V_F \\ 1, & \text{otherwise} \end{cases} . \tag{7.8}$$

Note that here the notation η is slightly abused. It represents a vector of clock variables (see Ex. 7.3.14). Eq. (7.6) and Eq. (7.7) are derived based on Eq. (7.3) and Eq. (7.2), respectively. In particular the multi-step reachability probability is computed using a sequence of one-step transition probabilities.

Hence we obtain a system of *integral equations* Eq. (7.8). One can read Eq. (7.8) either in the form $f(\xi) = \int_{Dom(\xi)} K(\xi, \xi') f(d\xi')$, where K is the kernel and $Dom(\xi)$ is the domain of integration depending on the continuous state space $\mathbb{S}$; or in the operator form $f(\xi) = (\mathcal{J}f)(\xi)$, where $\mathcal{J}$ is the integration operator. Generally, Eq. (7.8) does *not* need to have a unique solution. It turns out that the reachability probability $Prob_{v_0}^{\mathcal{D}}(\vec{0})$ coincides with the least fixpoint of the operator $\mathcal{J}$ (denoted by $\mathrm{lfp}\mathcal{J}$), i.e., $Prob_{v_0}^{\mathcal{D}}(\vec{0}) = (\mathrm{lfp}\mathcal{J})(v_0, \vec{0})$. Formally, we have:

Theorem 7.3.12 For any CTMC $\mathcal{C}$ and DTA $\mathcal{A}$, $\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\big(Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)\big)$ is the least solution of $Prob_{v_0}^{\mathcal{D}}(\cdot)$, where $\mathcal{D}$ is the embedded DTMP of $\mathcal{C} \otimes \mathcal{A}$.

Proof: We can express the set of all finite paths in $\mathcal{C}\otimes\mathcal{A}$ ending in some accepting location $\ell_n \in Loc_F$ for $n \in \mathbb{N}$ as the union over all location sequences, i.e.,

$$\begin{aligned} \Pi^{\mathcal{C}\otimes\mathcal{A}} &= \bigcup_{n\in\mathbb{N}} \bigcup_{(\ell_0,\cdots,\ell_n)\in Loc^{n+1}} C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n) \\ &= Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F) \cup \overline{Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)} \ . \end{aligned}$$

where $C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n)$ is a cylinder set, $I_i = [0, \infty)$ and $\overline{Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond Loc_F)}\downarrow_1$ are the set of paths which are not accepted by the DTA $\mathcal{A}$. Note that we can easily extend the measure $\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}$ to $\Pi^{\mathcal{C}\otimes\mathcal{A}}$ such that

$$\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\big(\Pi^{\mathcal{C}\otimes\mathcal{A}}\big) = \Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\big(Paths^{\mathcal{C}\otimes\mathcal{A}}(\Diamond\, Loc_F)\big) \ .$$

This means that in order to prove the theorem, we need to show that

$$\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}\big(\Pi^{\mathcal{C}\otimes\mathcal{A}}\big) = Prob_{v_0}^{\mathcal{D}}(\hat{\vec{0}}) \ , \tag{7.9}$$

where $Prob_{v_0}^{\mathcal{D}}(\hat{\vec{0}})$ is the short form of $Prob^{\mathcal{D}}\big((v_0, \hat{\vec{0}}), (V_F, \cdot)\big)$, i.e., the reachability probability from state $(v_0, \hat{\vec{0}})$ to $(V_F, \cdot)$. Note that for better readability, we indicate clock valuations in $\mathcal{D}$ by adding a "$\hat{}$".

Eq. (7.9) is to be shown on cylinder sets. Note that each cylinder set $C(\ell_0, I_0, \cdots, I_{n-1}, \ell_n) \subseteq \Pi^{\mathcal{C}\otimes\mathcal{A}}$ (C_n for short) induces a region graph $\mathcal{G}(C_n) = (V, v_0, V_F, \Lambda, \hookrightarrow)$, where its underlying PDP and embedded DTMP is $\mathcal{Z}(C_n)$ and $\mathcal{D}(C_n)$, respectively. To prove Eq. (7.9), it suffices to show that for each C_n,

$$\Pr_{\vec{0}}^{\mathcal{C}\otimes\mathcal{A}}(C_n) = Prob_{v_0}^{\mathcal{D}(C_n)}(\hat{\vec{0}}) \ ,$$

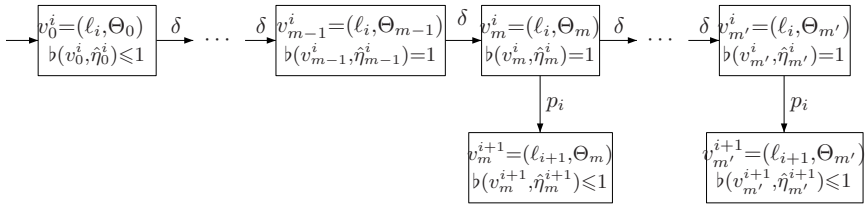


Figure 7.5: The sub-region graph for the transition from ℓ_i to ℓ_{i+1}

since $\Pi^{\mathcal{C}\otimes\mathcal{A}} = \bigcup_{n\in\mathbb{N}} \bigcup_{(\ell_0,\cdots,\ell_n)\in Loc^{n+1}} C_n$ and $\mathcal{D} = \bigcup_{n\in\mathbb{N}} \bigcup_{(\ell_0,\cdots,\ell_n)\in Loc^{n+1}} \mathcal{D}(C_n)$.

We will prove it by induction on the length n of the cylinder set $C_n \subseteq \Pi^{\mathcal{C}\otimes\mathcal{A}}$.

- For the base case of $n = 0$, i.e., $C_0 = C(\ell_i)$ and $\ell_i \in Loc_F$, it holds that $\Pr_{\eta_i}^{\mathcal{C}\otimes\mathcal{A}}(C_0) = 1$; while in the embedded DTMP $\mathcal{D}(C_0)$, since the initial vertex of $\mathcal{G}(C_0)$ is $v_0 = (\ell_i, \Theta_0)$, where $\eta_i \in \Theta_0$ and v_0 is consequently the initial location of $\mathcal{Z}(C_0)$ as well as $\mathcal{D}(C_0)$ which is accepting, $Prob_{v_0}^{\mathcal{D}(C_0)}(\hat{\eta}_i) = 1$. Note $\ell_i \in Loc$ is not necessarily the initial location ℓ_0.

- For the inductive case, we first assume, as the induction hypothesis, that for $n = k - 1$,
$$\Pr_{\eta_{i+1}}^{\mathcal{C}\otimes\mathcal{A}}(C_{k-1}) = Prob_{v_{i+1}}^{\mathcal{D}(C_{k-1})}(\hat{\eta}_{i+1}) \ ,$$
where $C_{k-1} = C(\ell_{i+1}, I_{i+1}, \cdots, I_{i+k-1}, \ell_{i+k})$ and $\ell_{i+k} \in Loc_F$. Note that $\ell_{i+1} \in Loc$ is not necessarily the initial location ℓ_0. We now proceed to consider the case of $n = k$. Let $C_k = C(\ell_i, I_i, \ell_{i+1}, I_{i+1}, \cdots, I_{i+k-1}, \ell_{i+k})$. As a result, there exists a transition $\ell_i \xmapsto[p_i]{g_i, X_i} \ell_{i+1}$ where $\eta_i + \tau_i \models g_i$ for every $\tau_i \in (t_1, t_2)$. $t_1, t_2 \in \mathbb{Q}_{\geq 0} \cup \{\infty\}$ can be obtained from g_i, such that $\tau_j \in (t_1, t_2)$ iff $\eta_i + \tau_j \models g_i$. According to the semantics of DMTA, we have
$$\Pr_{\eta_i}^{\mathcal{C}\otimes\mathcal{A}}(C_k) = \int_{t_1}^{t_2} p_i E(\ell_i) e^{-E(\ell_i)\tau_i} \cdot \Pr_{\eta_{i+1}}^{\mathcal{C}\otimes\mathcal{A}}(C_{k-1}) \ d\tau_i \ , \tag{7.10}$$
where $\eta_{i+1} = (\eta_i + \tau_i)[X_i := 0]$.

Now we deal with the inductive case for $\mathcal{D}(C_k)$. Let us assume that C_k induces the region graph $\mathcal{G}(C_k)$ whose subgraph corresponding to transition $\ell_i \xmapsto[p_i]{g_i, X_i} \ell_{i+1}$ is depicted in Fig. 7.5. For simplicity we consider that location ℓ_i induces the vertices $\{v_j^i = (\ell_i, \Theta_j) \mid 0 \leq j \leq m'\}$ and location ℓ_{i+1} induces the vertices $\{v_j^{i+1} = (\ell_{i+1}, \Theta_j) \mid m \leq j \leq m'\}$, respectively. Note that for Markovian transitions, the regions stay the same. We denote $\hat{\eta}_j^i$ (resp. $\hat{\eta}_j^{i+1}$) as the entering clock valuation on vertex v_j^i (resp. $\hat{\eta}_j^{i+1}$),

for j the indices of the regions. For any $\hat{\eta} \in \bigcup_{j=0}^{m-1} \Theta_j \cup \bigcup_{j>m'} \Theta_j$, $\hat{\eta} \not\models g_i$; or more specifically,

$$t_1 = \sum_{j=0}^{m-1} \flat(v_j^i, \hat{\eta}_j^i) \quad \text{and} \quad t_2 = \sum_{j=0}^{m'} \flat(v_j^i, \hat{\eta}_j^i) \ .$$

Recall that $\hat{\eta}_i$ (in the induction hypothesis) is the clock valuation to first hit a region with ℓ_i and $\hat{\eta}_i$. Given the fact that from v_0^i the process can only execute a delay transition before time t_1, it holds that

$$Prob_{v_0^i}^{\mathcal{D}(C_k)}(\hat{\eta}_i) = e^{-t_1 \Lambda(v_i)} \cdot Prob_{v_m^i}^{\mathcal{D}(C_k)}(\hat{\eta}_m^i) \ ,$$

and

$$Prob_{v_m^i}^{\mathcal{D}(C_k)}(\hat{\eta}_m^i) = Prob_{v_m^i,\delta}^{\mathcal{D}(C_k)}(\hat{\eta}_m^i) + Prob_{v_m^i,v_m^{i+1}}^{\mathcal{D}(C_k)}(\hat{\eta}_{i+1}) \ .$$

Therefore, we obtain by substitution of variables:

$$\begin{aligned}
& Prob_{v_0^i}^{\mathcal{D}}(\hat{\eta}_i) \\
= \ & e^{-t_1\Lambda(v_i)} \cdot Prob_{v_m^i,\delta}^{\mathcal{D}(C_k)}(\hat{\eta}_m^i) + e^{-t_1\Lambda(v_i)} \cdot Prob_{v_m^i,v_m^{i+1}}^{\mathcal{D}(C_k)}(\hat{\eta}_{i+1}) \\
= \ & e^{-t_1\Lambda(v_i)} \cdot Prob_{v_m^i,\delta}^{\mathcal{D}(C_k)}(\hat{\eta}_m^i) + \\
& e^{-t_1\Lambda(v_i)} \cdot \int_0^{\flat(v_m^i,\hat{\eta}_m^i)} p_i \Lambda(v_i) e^{-\Lambda(v_i)\tau} \cdot Prob_{v_m^{i+1}}^{\mathcal{D}(C_{k-1})}\big((\hat{\eta}_m^i + \tau)[X_i := 0]\big) d\tau \\
= \ & e^{-t_1\Lambda(v_i)} \cdot Prob_{v_m^i,\delta}^{\mathcal{D}(C_k)}(\hat{\eta}_m^i) + \\
& \int_{t_1}^{t_1+\flat(v_m^i,\hat{\eta}_m^i)} p_i \Lambda(v_i) e^{-\Lambda(v_i)\tau} \cdot Prob_{v_m^{i+1}}^{\mathcal{D}(C_{k-1})}\big((\hat{\eta}_m^i + \tau - t_1)[X_i := 0]\big) d\tau \ .
\end{aligned}$$

By evaluating each term $Prob_{v_m^i,\delta}^{\mathcal{D}(C_k)}(\hat{\eta}_m^i)$ we obtain the following sum of integrals:

$$\begin{aligned}
& Prob_{v_0^i}^{\mathcal{D}(C_k)}(\hat{\eta}_i) \\
= \ & \sum_{j=0}^{m'-m} \int_{t_1+\sum_{h=0}^{j-1} \flat(v_{m+h}^i,\hat{\eta}_{m+h}^i)}^{t_1+\sum_{h=0}^{j} \flat(v_{m+h}^i,\hat{\eta}_{m+h}^i)} p_i \Lambda(v_i) e^{-\Lambda(v_i)\tau} \cdot \\
& Prob_{v_{m+j}^{i+1}}^{\mathcal{D}(C_{k-1})}\big((\hat{\eta}_{m+j}^i + \tau - t_1 - \sum_{h=0}^{j-1} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i))[X_i := 0]\big) d\tau \ .
\end{aligned}$$

Now we define the function $F^{\mathcal{D}(C_{k-1})}(t) : [t_1, t_2] \to [0, 1]$, such that when $t \in [t_1 + \sum_{h=0}^{j-1} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i), t_1 + \sum_{h=0}^{j} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i)]$ for $j \leq m' - m$ then

$$F^{\mathcal{D}(C_{k-1})}(t) = Prob_{v_{m+j}^{i+1}}^{\mathcal{D}(C_{k-1})}\big((\hat{\eta}_{m+j}^i + t - t_1 - \sum_{h=0}^{j-1} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i))[X_i := 0]\big) \ .$$

Using $F^{\mathcal{D}(C_{k-1})}(t)$ we can rewrite $Prob_{v_0^i}^{\mathcal{D}(C_k)}(\hat{\eta}_i)$ to an equivalent form as follows:

$$\begin{aligned} & Prob_{v_0^i}^{\mathcal{D}(C_k)}(\hat{\eta}_i) \\ = & \sum_{j=0}^{m'-m} \int_{t_1+\sum_{h=0}^{j-1} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i)}^{t_1+\sum_{h=0}^{j} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i)} p_i \Lambda(v_i) e^{-\Lambda(v_i)\tau} F^{\mathcal{D}(C_{k-1})}(\tau) d\tau \\ = & \int_{t_1}^{t_2} p_i \Lambda(v_i) e^{-\Lambda(v_i)\tau} F^{\mathcal{D}(C_{k-1})}(\tau) d\tau \ . \end{aligned}$$

By the induction hypothesis, for every $t \in [t_1 + \sum_{h=0}^{j-1} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i), t_1 + \sum_{h=0}^{j} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i)]$ with $j \leq m' - m$, we have that:

$$\begin{aligned} \Pr_{\eta_{i+1}}^{\mathcal{C}\otimes\mathcal{A}}(C_{k-1}) &= Prob_{v_{m+j}^{i+1}}^{\mathcal{D}(C_{k-1})}\big((\hat{\eta}_{m+j}^i + t - t_1 - \sum_{h=0}^{j-1} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i))[X_i := 0]\big) \\ &= F^{\mathcal{D}(C_{k-1})}(t) \ , \end{aligned}$$

where $\eta_{i+1} = (\eta_i + t)[X_i := 0]$ and $\hat{\eta}_{m+j}^i = \hat{\eta}_i + t_1 + \sum_{h=0}^{j-1} \flat(v_{m+h}^i, \hat{\eta}_{m+h}^i)$. This shows that $\Pr_{\eta_i}^{\mathcal{C}\otimes\mathcal{A}}(C_k) = Prob_{v_i}^{\mathcal{D}(C_k)}(\hat{\eta}_i)$. The proof is now complete.

■

Remark 7.3.13 The region construction in Section 7.3.3 only helps us to obtain a PDP; it does *not* play an essential role as it does for, e.g. decidability of emptiness checking of TA [AD94]. This is mainly due to the continuously distributed random delays. (See [KNSS00] for a similar observation.) As a matter of fact, clock valuations η and η' in region Θ may induce different reachability probabilities. The reason is that η and η' may have different periods of time to hit the boundary, thus the probability for η and η' to either delay or take a Markovian transition may differ. This is in contrast with the traditional TA theory as well as probabilistic timed automata ([KNSS02], see also Chapter 8), where η and η' are not distinguished.

Example 7.3.14 For the region graph in Fig. 7.4(b), the system of integral equations for v_1 in location ℓ_0 is as follows: for $1 \leq x_1 = x_2 < 2$,

$$Prob_{v_1}^{\mathcal{D}}(x_1, x_2) = Prob_{v_1,\delta}^{\mathcal{D}}(x_1, x_2) + Prob_{v_1,v_3}^{\mathcal{D}}(x_1, x_2),$$

where

$$Prob_{v_1,\delta}^{\mathcal{D}}(x_1, x_2) = e^{-(2-x_1)r_0} \cdot Prob_{v_2}^{\mathcal{D}}(2, 2)$$

and

$$Prob_{v_1,v_3}^{\mathcal{D}}(x_1, x_2) = \int_0^{2-x_1} r_0 \cdot e^{-r_0\tau} \cdot Prob_{v_3}^{\mathcal{D}}(0, x_2 + \tau) \ d\tau$$

with $Prob_{v_3}^{\mathcal{D}}(0, x_2 + \tau) = 1$.

The integral equations for v_2 can be derived in the similar way.

7.3.5 Approximating Reachability Probabilities

Finally, we discuss how to obtain a solution of Eq. (7.8) (page 151). The integral equations (7.8) are *Volterra equations of the second type* [AW95]. For a general reference on solving Volterra equations, we refer the readers to, e.g. [Cor91]. Unfortunately, its exact complexity is not clear for us. (We conjecture that it lies in PSPACE, even in the *counting hierarchy* CH; see [ABKM09].) As an alternative option to solve Eq. (7.8), we proceed to give a formulation of $\Pr^{\mathcal{C}}\left(\mathit{Paths}^{\mathcal{C}}(\mathcal{A})\right)$ using a system of *partial differential equations* (PDEs), which is generally considered easier to tackle than integral equations. Let the *augmented* DTA $\mathcal{A}[t_f]$ be obtained from $\mathcal{A}$ by adding a new clock variable y which is never reset and a clock constraint $y \leq t_f$ on all edges entering the accepting locations in Loc_F, where t_f is a finite (and usually very large) integer. The purpose of this augmentation is to ensure that the values of all clocks reaching Loc_F are bounded. It is clear that $\mathit{Paths}^{\mathcal{C}}(\mathcal{A}[t_f]) \subseteq \mathit{Paths}^{\mathcal{C}}(\mathcal{A})$. More precisely, $\mathit{Paths}^{\mathcal{C}}(\mathcal{A}[t_f])$ coincides with those paths which can reach the accepting states of $\mathcal{A}$ within the time bound t_f. Note that $\lim_{t_f \to \infty} \Pr^{\mathcal{C}}(\mathit{Paths}^{\mathcal{C}}(\mathcal{A}[t_f])) = \Pr^{\mathcal{C}}(\mathit{Paths}^{\mathcal{C}}(\mathcal{A}))$. We can approximate $\Pr^{\mathcal{C}}(\mathit{Paths}^{\mathcal{C}}(\mathcal{A}))$ by solving the PDEs with a large t_f as follows:

Proposition 7.3.15 Given a CTMC $\mathcal{C}$, an augmented DTA $\mathcal{A}[t_f]$ and the underlying PDP $\mathcal{Z}(\mathcal{C} \otimes \mathcal{A}[t_f]) = (V, \mathcal{X}, \mathit{Inv}, \phi, \Lambda, \mu)$, $\Pr^{\mathcal{C}}\left(\mathit{Paths}^{\mathcal{C}}(\mathcal{A}[t_f])\right) = \hbar_{v_0}(0, \vec{0})$ (which is the probability to reach the final states in $\mathcal{Z}$ starting from initial state $(v_0, \vec{0}_{\mathcal{X}\cup\{y\}})$[4]) is the unique solution of the following system of PDEs:

$$\frac{\partial \hbar_v(y,\eta)}{\partial y} + \sum_{i=1}^{|\mathcal{X}|} \frac{\partial \hbar_v(y,\eta)}{\partial \eta^{(i)}} + \Lambda(v) \cdot \sum_{v \overset{p,X}{\hookrightarrow} v'} p \cdot \left(\hbar_{v'}(y, \eta[X := 0]) - \hbar_v(y,\eta)\right) = 0 \ ,$$

where $v \in V \setminus V_F$, $\eta \in \mathit{Inv}(v)$, $\eta^{(i)}$ is the i-th clock variable, and $y \in [0, t_f)$. For every $\eta \in \partial \mathit{Inv}(v)$ and transition $v \overset{\delta}{\hookrightarrow} v'$, the boundary conditions take the form: $\hbar_v(y,\eta) = \hbar_{v'}(y,\eta)$. For every vertex $v \in V_F$, $\eta \in \mathit{Inv}(v)$ and $y \in [0, t_f)$, we have the following PDE:

$$\frac{\partial \hbar_v(y,\eta)}{\partial y} + \sum_{i=1}^{|\mathcal{X}|} \frac{\partial \hbar_v(y,\eta)}{\partial \eta^{(i)}} + 1 = 0 \ .$$

The final boundary conditions are that for every vertex $v \in V$ and $\eta \in \mathit{Inv}(v) \cup \partial \mathit{Inv}(v)$, $\hbar_v(t_f, \eta) = 0$.

Proof: For any set of clocks $\mathcal{X}$ (n clocks) of the PDP $\mathcal{Z} = (Z, \mathcal{X}, \mathit{Inv}, \phi, \Lambda, \mu)$ we define a system of ODEs:

$$\frac{d\eta(y)}{dy} = \vec{1}, \quad \eta(y_0) = \eta_0 \in \mathbb{R}^n_{\geq 0} \ , \tag{7.11}$$

[4] $\vec{0}_{\mathcal{X}\cup\{y\}}$ denotes the valuation η with $\eta(x) = 0$ for $x \in \mathcal{X} \cup \{y\}$.

which describe the evolution of clock values $\eta(y)$ at time y given the initial value η_0 of all clocks at time y_0. Note that contrary to our DTA notation, Eq. (7.11) describes a system of ODEs where $\eta(y)$ is a vector of clock valuations at time y and $\frac{d\eta^{(i)}(y)}{dy}$ gives the timed evolution of clock $\eta^{(i)}$. Given a continuous differentiable functional $f : Z \times \mathbb{R}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$, for every $z \in Z$, we define:

$$\begin{aligned} \frac{df(z,\eta(y))}{dy} &= \sum_{i=1}^{n} \frac{\partial f(z,\eta(y))}{\partial \eta^{(i)}} \cdot \frac{d\eta^{(i)}(y)}{dy} \\ &\overset{(7.11)}{=} \sum_{i=1}^{n} \frac{\partial f(z,\eta(y))}{\partial \eta^{(i)}} \ . \end{aligned}$$

For notation simplicity we define the vector field from Eq. (7.12) as the operator Ξ which acts on functional $f(z,\eta(y))$, i.e., $\Xi f(z,\eta(y)) = \sum_{i=1}^{n} \frac{\partial f(z,\eta(y))}{\partial \eta^{(i)}}$. We also define the equivalent notation $\Xi f(\xi)$ for the state $\xi = (z,\eta(y))$ and any $y \in \mathbb{R}_{\geq 0}$.

We can define the value of $\Pr^{\mathcal{C}}\left(Paths^{\mathcal{C}}(\mathcal{A})\right)$ as the expectation $\hbar(0,\xi_0)$ on PDP $\mathcal{Z}$ as follows:

$$\begin{aligned} \hbar(0,\xi_0) &= \mathbb{E}\left[\int_0^{t_f} \mathbf{1}_{\mathcal{Z}}(X_\tau)d\tau \mid X_0 = \xi_0\right] \\ &= \mathbb{E}_{(0,\xi_0)}\left[\int_0^{t_f} \mathbf{1}_{\mathcal{Z}}(X_\tau)d\tau\right] \ , \end{aligned}$$

where the initial starting time is 0, the starting state is $\xi_0 = (z_0, \vec{0})$, X_τ is the underlying stochastic process of $\mathcal{Z}$ defined on the state space $\mathbb{S}$, and $\mathbf{1}_{\mathcal{Z}}(X_\tau) = 1$ when $X_\tau \in \{(z,\eta(\tau)) \mid z \in V_F, \eta(\tau) \in Inv(z)\}$, $\mathbf{1}_{\mathcal{Z}}(X_\tau) = 0$, otherwise. Note that we can also define the expectation in Eq. (7.12) for any starting time $y < t_f$ and state ξ as $\mathbb{E}_{(y,\xi)}\left[\int_y^{t_f} \mathbf{1}_{\mathcal{Z}}(X_\tau)d\tau\right]$.

We then obtain the expectation $\hbar(0,\xi_0)$ by following the construction in [Dav93]. For this we form the new state space $\hat{\mathbb{S}} = ([0,t_f] \times \mathbb{S}) \cup \{\Delta\}$ where Δ is the sink state and the boundary is $\partial\hat{\mathbb{S}} := ([0,t_f] \times \partial\mathbb{S}) \cup (\{t_f\} \times \mathbb{S})$. We define the following functions: $\hat{\Lambda}(y,\xi) = \Lambda(\xi)$, $\hat{\mu}((y,\xi),\{y\} \times A) = \mu(\xi,A)$ and $\hat{\mu}((t_f,\xi),\{\Delta\}) = 1$ for $y \in [0,t_f)$, $A \subseteq \mathbb{S}$ and $\xi \in \mathbb{S}$.

Given the construction we obtain an equivalent form for the expectation Eq. (7.12), i.e.:

$$\hbar(0,\xi_0) = \mathbb{E}_{(0,\xi_0)}\left[\int_0^{\infty} \vec{1}_{\mathcal{Z}}(\tau, X_\tau)d\tau\right] \ , \tag{7.12}$$

where $\vec{1}_{\mathcal{Z}} : \hat{\mathbb{S}} \to \{0,1\}$, $\vec{1}_{\mathcal{Z}}(\tau,X_\tau) = 1$ when $X_\tau \in \{(z,\eta(\tau)) \mid z \in V_F, \eta(\tau) \in Inv(z)\}$ and $\tau \in [0,t_f)$, $\vec{1}_{\mathcal{Z}}(\tau,X_\tau) = 0$ otherwise. We also define $\vec{1}_{\mathcal{Z}}(\Delta)$ to be zero. Note that we introduce the sink state Δ in order to ensure that $\lim_{y\to\infty} \mathbb{E}_{(0,\xi)}\hbar(y,X_y) = 0$, which is a crucial condition in order to obtain a unique value for the expectation $\hbar(0,\xi_0)$.

For the expectation Eq. (7.12), [Dav93] defines the following integro-differential equations (for any $y \in [0, t_f)$):

$$\mathcal{U}\hbar(y,\xi) = \Xi\hbar(y,\xi) + \hat{\Lambda}(y,\xi) \cdot \int_{\mathbb{S}} (\hbar(y,\xi') - \hbar(y,\xi))\, \hat{\mu}((y,\xi),(y,d\xi')), \xi \in \mathbb{S} \tag{7.13}$$

$$\hbar(y,\xi) = \int_{\mathbb{S}} \hbar(y,\xi')\hat{\mu}((y,\xi),(y,d\xi')), \xi \in \partial\mathbb{S} \tag{7.14}$$

$$\mathcal{U}\hbar(y,\xi) + \vec{\hat{1}}_{\mathcal{Z}}(y,\xi) = 0, \xi \in \mathbb{S} \ . \tag{7.15}$$

Eq. (7.13) denotes the generator of the stochastic process X_y, and Eq. (7.14) states the boundary conditions for Eq. (7.15). We can rewrite the integro-differential equations (7.13), (7.14) and (7.15) into a system of PDEs with boundary conditions, given the fact that the measure $\hat{\mu}$ is not uniform. For each vertex $v \notin V_F$, $\eta \in Inv(v)$ and $y \in [0, t_f)$ of the region graph $\mathcal{G}$, we write the PDE as follows (here we define $\hbar_v(y,\eta) := \hbar(y,\xi)$ for $\xi = (v,\eta)$):

$$\frac{\partial \hbar_v(y,\eta)}{\partial y} + \sum_i \frac{\partial \hbar_v(y,\eta)}{\partial \eta^{(i)}} + \Lambda(v) \sum_{v \overset{p,X}{\hookrightarrow} v'} p \cdot (\hbar_{v'}(y, \eta[X := 0]) - \hbar_v(y,\eta)) = 0 \ .$$

Note that for any edge $v \overset{p,X}{\hookrightarrow} v'$ in the region graph $\mathcal{G}$, $\hat{\mu}((y,(v,\eta)),(y,(v',\eta'))) = p$. For every $\eta \in \partial Inv(v)$ and transition $v \overset{\delta}{\hookrightarrow} v'$, the boundary conditions take the form: $\hbar_v(y,\eta) = \hbar_{v'}(y,\eta)$. For every vertex $v \in V_F$, $\eta \in Inv(v)$ and $y \in [0, t_f)$, we get:

$$\frac{\partial \hbar_v(y,\eta)}{\partial y} + \sum_i \frac{\partial \hbar_v(y,\eta)}{\partial \eta^{(i)}} + 1 = 0 \ .$$

Note that all final states are made absorbing. The final boundary conditions are that for every vertex $v \in Z$ and $\eta \in Inv(v) \cup \partial Inv(v)$, $\hbar_v(t_f,\eta)$=0. ■

7.4 Single-clock DTA Specifications

For single-clock DTA specifications, we can simplify the system of integral equations obtained in the previous section to a system of *linear* equations, where the coefficients are a solution of a system of ODEs that can be calculated efficiently.

Given a DMTA $\mathcal{M}$, we denote the set of constants appearing in the clock constraints of $\mathcal{M}$ as $\{c_0, \ldots, c_m\}$ with $c_0 = 0$. We assume the following order: $0 = c_0 < c_1 < \cdots < c_m$. Let $\Delta c_i = c_{i+1} - c_i$ for $0 \leq i < m$. Note that for single-clock DMTA, the regions in the region graph $\mathcal{G}(\mathcal{M})$ can be represented by the following intervals: $[c_0, c_1), \ldots, [c_m, \infty)$ [LMS04]. We partition the region graph $\mathcal{G}(\mathcal{M}) = (V, v_0, V_F, \Lambda, \hookrightarrow)$, or $\mathcal{G}$ for short, into a set of subgraphs $\mathcal{G}_i = (V_i, V_{F_i}, \Lambda_i, \{M_i, F_i, B_i\})$, where $0 \leq i \leq m$, and $\Lambda_i(v) = \Lambda(v)$ if $v \in V_i$, 0 otherwise. These subgraphs are obtained by partitioning V, V_F and $\hookrightarrow$ as follows:

- $V = \bigcup_{0\le i\le m}\{V_i\}$, where $V_i = \{(\ell,\Theta) \in V \mid \Theta \subseteq [c_i, c_{i+1})\}$;
- $V_F = \bigcup_{0\le i\le m}\{V_{F_i}\}$, where $v \in V_{F_i}$ iff $v \in V_i \cap V_F$; and
- $\hookrightarrow = \bigcup_{0\le i\le m}\{M_i \cup F_i \cup B_i\}$, where M_i is the set of *Markovian transitions (without reset)* between vertices inside $\mathcal{G}_i$; F_i is the set of *delay transitions* from the vertices in $\mathcal{G}_i$ to that in $\mathcal{G}_{i+1}$ (**F**orward) and B_i is the set of *Markovian transitions (with reset)* from $\mathcal{G}_i$ to $\mathcal{G}_0$ (**B**ackward). It is easy to see that M_i, F_i, and B_i are pairwise disjoint.

Since the initial vertex of $\mathcal{G}_0$ is v_0 and the initial vertices of $\mathcal{G}_i$ for $0 < i \le m$ are implicitly given by $\mathbf{F}_{i-1}$, we omit them in the definition. As an example, the vertices in Fig. 7.6 are partitioned by the ovals and the M_i edges are unlabeled, while the F_i and B_i edges are labeled with δ and "reset", respectively. The V_F vertices (double circles) may appear in any $\mathcal{G}_i$. Actually, if $v = (\ell, [c_i, c_{i+1})) \in V_F$, then $v' = (\ell, [c_j, c_{j+1})) \in V_F$ for $i < j \le m$. This is true because $V_F = \{(\ell, \mathsf{true}) \mid \ell \in Loc_F\}$. It implies that for each final vertex not in the last region, there is a delay transition from it to the next region, see the final vertex in $\mathcal{G}_{i+1}$ in Fig. 7.6. The exit rate functions and the probabilities on Markovian edges are omitted in the graph.

Given a subgraph $\mathcal{G}_i$ $(0{\le}i{\le}m)$ of $\mathcal{G}$ with k_i states, let the probability vector $\vec{U}_i(x) = [u_i^1(x), \ldots, u_i^{k_i}(x)]^\top \in \mathbb{R}^{k_i\times 1}$, where $u_i^j(x)$ is the probability to go from vertex $v_i^j \in V_i$ to vertices in V_F (in $\mathcal{G}$) at time x. Starting from Eq. (7.6)-Eq. (7.8), we provide a set of integral equations for $\vec{U}_i(x)$, which we later on reduce to a system of linear equations. Distinguish two cases:

Case 1: $0 \le i < m$. $\vec{U}_i(x)$ is given by:

$$\vec{U}_i(x) = \int_0^{\Delta c_i - x} \mathbf{M}_i(\tau)\vec{U}_i(x+\tau)d\tau \tag{7.16}$$

$$+ \int_0^{\Delta c_i - x} \mathbf{B}_i(\tau)d\tau \cdot \vec{U}_0(0) \tag{7.17}$$

$$+ \mathbf{D}_i(\Delta c_i - x)\cdot \mathbf{F}_i\vec{U}_{i+1}(0)\ , \tag{7.18}$$

where $x \in [0, \Delta c_i]$, and

- $\mathbf{D}_i(x) \in \mathbb{R}^{k_i\times k_i}$ is the *delay probability matrix*, where for any $0 \le j \le k_i$, $\mathbf{D}_i(x)[j,j] = e^{-E(v_i^j)x}$ (the off-diagonal elements are zero);
- $\mathbf{M}_i(x) = \mathbf{D}_i(x){\cdot}\mathbf{P}_i{\cdot}\mathbf{E}_i \in \mathbb{R}^{k_i\times k_i}$ is the *probability density matrix* for the Markovian transitions inside $\mathcal{G}_i$, where $\mathbf{P}_i$ and $\mathbf{E}_i$ are the *transition probability matrix* and exit rate matrix for vertices inside $\mathcal{G}_i$, respectively;
- $\mathbf{B}_i(x) \in \mathbb{R}^{k_i\times k_0}$ is the *probability density matrix* for the reset edges B_i, where $\mathbf{B}_i(x)[j,j']$ indicates the probability density function to take the Markovian jump with reset from the j-th vertex in $\mathcal{G}_i$ to the j'-th vertex in $\mathcal{G}_0$; and

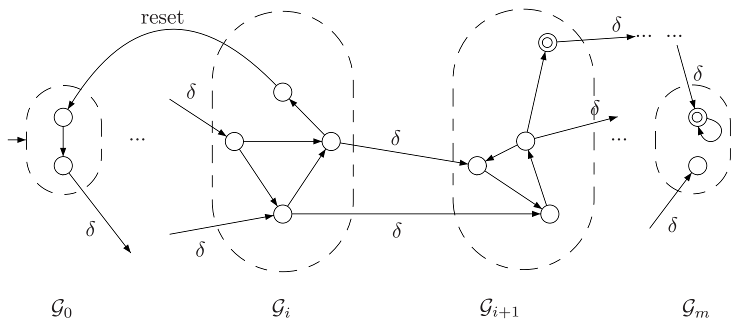


Figure 7.6: Partitioning the region graph

- $\mathbf{F}_i \in \mathbb{R}^{k_i \times k_{i+1}}$ is the *incidence matrix* for delay edges F_i. More specifically, $\mathbf{F}_i[j, j'] = 1$ indicates that there is a delay transition from the j-th vertex in $\mathcal{G}_i$ to the j'-th vertex in $\mathcal{G}_{i+1}$; 0 otherwise.

Let us explain these equations. Eq. (7.18) is obtained from Eq. (7.6), where $\mathbf{D}_i(\Delta c_i - x)$ indicates the probability to delay until the "end" of region i, and $\mathbf{F}_i \vec{U}_{i+1}(0)$ denotes the probability to continue in $\mathcal{G}_{i+1}$ (at relative time 0). In a similar way, Eq. (7.16) and Eq. (7.17) are obtained from Eq. (7.7); the former reflects the case where clock x is not reset, while the latter considers the reset of x (and returning to $\mathcal{G}_0$).

CASE 2: $i = m$. $\vec{U}_m(x)$ is simplified as follows:

$$\vec{U}_m(x) = \int_0^\infty \hat{\mathbf{M}}_m(\tau)\vec{U}_m(x+\tau)d\tau + \vec{1}_F + \int_0^\infty \mathbf{B}_m(\tau)d\tau \cdot \vec{U}_0(0) \ , \quad (7.19)$$

where $\hat{\mathbf{M}}_m(\tau)[v,\cdot] = \mathbf{M}_m(\tau)[v,\cdot]$ for $v \notin V_F$, 0 otherwise. $\vec{1}_F$ is a vector such that $\vec{1}_F[v] = 1$ if $v \in V_F$, 0 otherwise. We note that $\vec{1}_F$ stems from the second clause of Eq. (7.8), and $\hat{\mathbf{M}}_m$ is obtained by setting the corresponding elements of $\mathbf{M}_m$ to 0. Also note that as the last subgraph $\mathcal{G}_m$ involves infinite regions, it has no delay transitions.

Before solving the system of integral equations (7.16)-(7.19), we first make the following observations:

(i) Due to the fact that inside $\mathcal{G}_i$ there are only Markovian jumps with neither resets nor delay transitions, $\mathcal{G}_i$ with (V_i, Λ_i, M_i) forms a CTMC $\mathcal{C}_i$, say. For each $\mathcal{G}_i$ we define an *augmented* CTMC $\mathcal{C}_i^a$ with state space $V_i \cup V_0$, such that all V_0-vertices are made absorbing in $\mathcal{C}_i^a$. The edges connecting V_i to V_0 are kept and all the edges inside $\mathcal{C}_0$ are removed. The augmented CTMC is used to calculate the probability to start from a vertex in $\mathcal{G}_i$ and take a reset edge in a certain time.

(ii) Given any CTMC $\mathcal{C}$ with k states and rate matrix $\mathbf{P} \cdot \mathbf{E}$, the matrix $\mathbf{\Pi}(x)$ is given by:

$$\mathbf{\Pi}(x) = \int_0^x \mathbf{M}(\tau)\mathbf{\Pi}(x-\tau)d\tau + \mathbf{D}(x) \ . \tag{7.20}$$

Intuitively, $\mathbf{\Pi}(t)[j, j']$ indicates the probability to start from vertex j and reach j' at time t.

The following proposition states the close relationship between $\mathbf{\Pi}(x)$ and the transient probability vector:

Proposition 7.4.1 Given a CTMC $\mathcal{C}$ with initial distribution α, rate matrix $\mathbf{P}\cdot\mathbf{E}$ and $\mathbf{\Pi}(t)$, $\vec{\pi}(t)$ satisfies the following two equations:

$$\vec{\pi}(t) = \alpha \cdot \mathbf{\Pi}(t) \ , \tag{7.21}$$

$$\frac{d\vec{\pi}(t)}{dt} = \vec{\pi}(t) \cdot \mathbf{Q} \ , \tag{7.22}$$

where $\mathbf{Q} = \mathbf{P}\cdot\mathbf{E} - \mathbf{E}$ is the infinitesimal generator.

Proof: The transition probability matrix $\mathbf{\Pi}(t)$ for a CTMC $\mathcal{C}$ with state space S is denoted by the following system of integral equations:

$$\mathbf{\Pi}(t) = \int_0^t \mathbf{M}(\tau)\mathbf{\Pi}(t-\tau)d\tau + \mathbf{D}(t) \ , \tag{7.23}$$

where $\mathbf{M}(\tau) = \mathbf{D}(\tau)\cdot\mathbf{P}\cdot\mathbf{E}$. Now we define, for the CTMC $\mathcal{C}$, a stochastic process $X(t)$. The probability $\Pr(X(t+\Delta t) = s_j)$ to be in state s_j at time $t + \Delta t$ can be defined as:

$$\Pr(X(t+\Delta t) = s_j) = \sum_{s_i \in S} \Pr(X(t) = s_i) \cdot \Pr(X(t+\Delta t) = s_j | X(t) = s_i) \ .$$

We can define $\Pr(X(t+\Delta t) = s_j)$ in the vector form as follows:

$$\vec{\pi}(t+\Delta t) = \vec{\pi}(t)\mathbf{\Phi}(t, t+\Delta t) \ ,$$

where

$$\vec{\pi}(t) = [\Pr(X(t) = s_1), \cdots, \Pr(X(t) = s_n)] \ ,$$

and

$$\mathbf{\Phi}(t, t+\Delta t)[i, j] = \Pr(X(t+\Delta t) = s_j | X(t) = s_i) \ .$$

As the stochastic process $X(t)$ is time-homogeneous, we have that

$$\Pr(X(t+\Delta t) = s_j | X(t) = s_i) = \Pr(X(\Delta t) = s_j | X(0) = s_i) \ ,$$

which means that $\mathbf{\Phi}(t, t+\Delta t) = \mathbf{\Phi}(0, \Delta t)$. As $\Pr(X(\Delta t) = s_j | X(0) = s_i)$ denotes the transition probability to go from state s_i to state s_j at time Δt, we have that $\mathbf{\Phi}(0, \Delta t) = \mathbf{\Pi}(\Delta t)$, which results in the equation:

$$\vec{\pi}(t+\Delta t) = \vec{\pi}(t)\mathbf{\Pi}(\Delta t) \ . \tag{7.24}$$

Now we transform Eq. (7.24) as follows:

$$\begin{array}{ll} & \vec{\pi}(t+\Delta t) = \vec{\pi}(t)\mathbf{\Pi}(\Delta t) \\ \Longrightarrow & \vec{\pi}(t+\Delta t) - \vec{\pi}(t) = \vec{\pi}(t)\mathbf{\Pi}(\Delta t) - \vec{\pi}(t) = \vec{\pi}(t)(\mathbf{\Pi}(\Delta t) - \mathbf{I}) \\ \Longrightarrow & \dfrac{d\vec{\pi}(t)}{dt} = \lim\limits_{\Delta t \to 0} \dfrac{\vec{\pi}(t+\Delta t) - \vec{\pi}(t)}{\Delta t} = \vec{\pi}(t) \lim\limits_{\Delta t \to 0} \dfrac{\mathbf{\Pi}(\Delta t) - \mathbf{I}}{\Delta t} \ . \end{array}$$

Now the task is to compute $\lim_{\Delta t \to 0} \dfrac{\mathbf{\Pi}(\Delta t) - \mathbf{I}}{\Delta t}$. For this we rewrite the right-hand limit as:

$$\lim_{\Delta t \to 0} \frac{1}{\Delta t} \int_0^{\Delta t} \mathbf{M}(\tau)\mathbf{\Pi}(\Delta t - \tau)d\tau + \lim_{\Delta t \to 0} \frac{1}{\Delta t} \left(\mathbf{D}(\Delta t) - \mathbf{I}\right) \ .$$

The $\lim_{\Delta t \to 0} \frac{1}{\Delta t} \int_0^{\Delta t} \mathbf{M}(\tau)\mathbf{\Pi}(\Delta t - \tau)d\tau$ is of the type $\frac{0}{0}$. Note that

$$\frac{d}{d\Delta t}\left(\int_0^{\Delta t} \mathbf{M}(\tau)\mathbf{\Pi}(\Delta t - \tau)d\tau\right) = \mathbf{M}(\Delta t)\mathbf{\Pi}(0) + \int_0^{\Delta t} \mathbf{M}(\tau)\frac{\partial}{\partial \Delta t}\mathbf{\Pi}(\Delta t - \tau)d\tau$$

and $\mathbf{D}(0) = \mathbf{\Pi}(0) = \mathbf{I}$, by Eq. (7.23). By l'Hôspital rule, we obtain:

$$\begin{array}{ll} & \lim\limits_{\Delta t \to 0} \dfrac{1}{\Delta t} \displaystyle\int_0^{\Delta t} \mathbf{M}(\tau)\mathbf{\Pi}(\Delta t - \tau)d\tau \\ = & \lim\limits_{\Delta t \to 0} \left(\mathbf{M}(\Delta t)\mathbf{\Pi}(0) + \displaystyle\int_0^{\Delta t} \mathbf{M}(\tau)\frac{\partial}{\partial \Delta t}\mathbf{\Pi}(\Delta t - \tau)d\tau\right) \\ = & \mathbf{M}(0)\mathbf{\Pi}(0) \\ = & \mathbf{P}{\cdot}\mathbf{E} \ . \end{array}$$

The $\lim_{\Delta t \to 0} \frac{1}{\Delta t} \left(\mathbf{D}(\Delta t) - \mathbf{I}\right)$ is of the type $\frac{0}{0}$. Note that

$$\frac{d}{d\Delta t}\left(\mathbf{D}(\Delta t) - \mathbf{I}\right) = -\mathbf{E}\mathbf{D}(\Delta t) \ .$$

Therefore by l'Hôspital rule, we obtain $\lim_{\Delta t \to 0} \frac{1}{\Delta t} \left(\mathbf{D}(\Delta t) - \mathbf{I}\right) = -\mathbf{E}$ and

$$\lim_{\Delta t \to 0} \frac{\mathbf{\Pi}(\Delta t) - \mathbf{I}}{\Delta t} = \mathbf{P}{\cdot}\mathbf{E} - \mathbf{E} = \mathbf{Q} \ ,$$

where $\mathbf{Q}$ is the infinitesimal generator of the CTMC $\mathcal{C}$. As a result we obtain:

$$\frac{d\vec{\pi}(t)}{dt} = \vec{\pi}(t) \lim_{\Delta t \to 0} \frac{\mathbf{\Pi}(\Delta t) - \mathbf{I}}{\Delta t} = \vec{\pi}(t)\mathbf{Q} \ .$$

Combining with Eq. (7.24) we obtain:

$$\vec{\pi}(t) = \alpha \cdot \mathbf{\Pi}(t) \quad \text{and} \quad \frac{d\vec{\pi}(t)}{dt} = \vec{\pi}(t) \cdot \mathbf{Q} \ .$$

■

Remark 7.4.2 Prop. 7.4.1 has been observed in [BHHK03] and plays an essential role in developing the efficient algorithm. However, to the best of our knowledge, a rigorous proof is missing in the literature.

$\vec{\pi}(t)$ is the *transient probability vector* with $\vec{\pi}(t)[s]$ indicating the probability to be in state s at time t given the initial probability distribution α. Eq. (7.22) is the celebrated forward Chapman-Kolmogorov equation. According to this proposition, solving the integral equation $\mathbf{\Pi}(t)$ boils down to selecting the appropriate initial distribution vector α and solving the system of ODEs in Eq. (7.22), which can be done very efficiently using the *uniformization technique*.

Prior to exposing in the next theorem how to solve the system of integral equations by solving a system of *linear* equations, we define $\bar{\mathbf{\Pi}}_i^a \in \mathbb{R}^{k_i \times k_0}$ for an augmented CTMC $\mathcal{C}_i^a$ to be part of $\mathbf{\Pi}_i^a$, where $\bar{\mathbf{\Pi}}_i^a$ only keeps the probabilities starting from V_i and ending in V_0. Actually,

$$\mathbf{\Pi}_i^a(x) = \left(\begin{array}{c|c} \mathbf{\Pi}_i(x) & \bar{\mathbf{\Pi}}_i^a(x) \\ \hline \mathbf{0} & \mathbf{I} \end{array}\right) \ ,$$

where $\mathbf{0} \in \mathbb{R}^{k_0 \times k_i}$ is the matrix with all elements zero, and $\mathbf{I} \in \mathbb{R}^{k_0 \times k_0}$ is the identity matrix.

Theorem 7.4.3 For subgraph $\mathcal{G}_i$ of $\mathcal{G}$ with k_i states, it holds for $0 \leq i < m$ that:

$$\vec{U}_i(0) = \mathbf{\Pi}_i(\Delta c_i) \cdot \mathbf{F}_i \vec{U}_{i+1}(0) + \bar{\mathbf{\Pi}}_i^a(\Delta c_i) \cdot \vec{U}_0(0) \ , \tag{7.25}$$

where $\mathbf{\Pi}_i(\Delta c_i)$ and $\bar{\mathbf{\Pi}}_i^a(\Delta c_i)$ are for CTMC $\mathcal{C}_i$ and the augmented CTMC $\mathcal{C}_i^a$, respectively. For the case $i = m$, it holds that:

$$\vec{U}_m(0) = \hat{\mathbf{P}}_i \cdot \vec{U}_m(0) + \vec{1}_F + \hat{\mathbf{B}}_m \cdot \vec{U}_0(0) \ , \tag{7.26}$$

where $\hat{\mathbf{P}}_i(v, v') = \mathbf{P}_i(v, v')$ if $v \notin V_F$; 0 otherwise, and $\hat{\mathbf{B}}_m = \int_0^\infty \mathbf{B}_m(\tau) d\tau$.

Proof: We first deal with the case $i < m$. If in $\mathcal{G}_i$ there exists some backward edge, namely, for some j, j', $B_i(x)[j, j'] \neq 0$, then we shall consider the *augmented* CTMC $\mathcal{C}_i^a$ with $k_i^a = k_i + k_0$ states. In view of this, the augmented integral equation $\vec{U}_i^a(x)$ is defined as:

$$\vec{U}_i^a(x) = \int_0^{\Delta c_i - x} \mathbf{M}_i^a(\tau) \vec{U}_i^a(x + \tau) d\tau + \mathbf{D}_i^a(\Delta c_i - x) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \ ,$$

where $\vec{U}_i^a(x) = \left(\begin{array}{c} \vec{U}_i(x) \\ \hline \vec{U}_i'(x) \end{array}\right) \in \mathbb{R}^{k_i^a \times 1}$, $\vec{U}_i'(x) \in \mathbb{R}^{k_0 \times 1}$ is the vector representing the reachability probability for the augmented states in $\mathcal{G}_i$, $\mathbf{F}_i^a = \left(\begin{array}{c|c} \mathbf{F}_i' & \mathbf{B}_i' \end{array}\right) \in \mathbb{R}^{k_i^a \times (k_{i+1} + k_0)}$ such that $\mathbf{F}_i' = \left(\begin{array}{c} \mathbf{F}_i \\ \hline \mathbf{0} \end{array}\right) \in \mathbb{R}^{k_i^a \times k_{i+1}}$ is the incidence matrix for delay edges, and $\mathbf{B}_i' = \left(\begin{array}{c} \mathbf{0} \\ \hline \mathbf{I} \end{array}\right) \in \mathbb{R}^{k_i^a \times k_0}$, $\vec{\hat{U}}_i(0) = \left(\begin{array}{c} \vec{U}_{i+1}(0) \\ \hline \vec{U}_0(0) \end{array}\right) \in \mathbb{R}^{(k_{i+1} + k_0) \times 1}$.

First, we prove the following equation:

$$\vec{U}_i^a(x) = \mathbf{\Pi}_i^a(\Delta c_i - x) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \ ,$$

where

$$\mathbf{\Pi}_i^a(x) = \int_0^x \mathbf{M}_i^a(\tau)\mathbf{\Pi}_i^a(x-\tau)d\tau + \mathbf{D}_i^a(x) \ . \tag{7.27}$$

Set $c_{i,x} = \Delta c_i - x$. We consider the iterations of the solution of the following system of integral equations:

$$\begin{aligned} \vec{U}_i^{a,(0)}(x) &= \vec{0} \\ \vec{U}_i^{a,(j+1)}(x) &= \int_0^{c_{i,x}} \mathbf{M}_i^a(\tau)\vec{U}_i^{a,(j)}(x+\tau)d\tau + \mathbf{D}_i^a(c_{i,x}) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \ , \end{aligned}$$

and

$$\begin{aligned} \mathbf{\Pi}_i^{a,(0)}(c_{i,x}) &= \mathbf{0} \\ \mathbf{\Pi}_i^{a,(j+1)}(c_{i,x}) &= \int_0^{c_{i,x}} \mathbf{M}_i^a(\tau)\mathbf{\Pi}_i^{a,(j)}(c_{i,x}-\tau)d\tau + \mathbf{D}_i^a(c_{i,x}) \ . \end{aligned}$$

By induction on j, we prove the following equation:

$$\vec{U}_i^{a,(j)}(x) = \mathbf{\Pi}_i^{a,(j)}(c_{i,x}) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \ . \tag{7.28}$$

- Base case: $\vec{U}_i^{a,(0)}(x) = \vec{0}$ and $\mathbf{\Pi}_i^{a,(0)}(c_{i,x}) = \mathbf{0}$. The equation clearly holds.
- Inductive case: By the induction hypothesis,

  $$\vec{U}_i^{a,(j)}(x) = \mathbf{\Pi}_i^{a,(j)}(c_{i,x}) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \ .$$

  It follows that

  $$\begin{aligned} &\vec{U}_i^{a,(j+1)}(x) \\ &= \int_0^{c_{i,x}} \mathbf{M}_i^a(\tau)\vec{U}_i^{a,(j)}(x+\tau)d\tau + \mathbf{D}_i^a(c_{i,x}) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \\ &= \int_0^{c_{i,x}} \mathbf{M}_i^a(\tau)\mathbf{\Pi}_i^{a,(j)}(c_{i,x}-\tau) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0)d\tau + \mathbf{D}_i^a(c_{i,x}) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \\ &= \left(\int_0^{c_{i,x}} \mathbf{M}_i^a(\tau)\mathbf{\Pi}_i^{a,(j)}(c_{i,x}-\tau)d\tau + \mathbf{D}_i^a(c_{i,x}) \right) \cdot \mathbf{F}_i^a \vec{\hat{U}}_i(0) \\ &= \mathbf{\Pi}_i^{a,(j+1)}(c_{i,x}) \cdot \mathbf{F}_i \vec{\hat{U}}_i(0) \ . \end{aligned}$$

  This completes the induction.

Clearly,

$$\mathbf{\Pi}_i^a(c_{i,x}) = \lim_{j\to\infty} \mathbf{\Pi}_i^{a,(j)}(c_{i,x}) \quad \text{and} \quad \vec{U}_i^a(x) = \lim_{j\to\infty} \vec{U}_i^{a,(j)}(x) \ .$$

By taking the limit as $j \to \infty$ on both sides of Eq. (7.28), we obtain

$$\vec{U}_i^a(x) = \mathbf{\Pi}_i^a(c_{i,x}) \cdot \mathbf{F}_i^a \hat{\vec{U}}_i(0) \ .$$

Let $x = 0$. We obtain (recall that $c_{i,x} = \Delta c_i - x$)

$$\vec{U}_i^a(0) = \mathbf{\Pi}_i^a(\Delta c_i) \cdot \mathbf{F}_i^a \hat{\vec{U}}_i(0) \ .$$

We can also write the above relation as:

$$\begin{aligned}
&\left(\frac{\vec{U}_i(0)}{\vec{U}_i'(0)}\right) = \mathbf{\Pi}_i^a(\Delta c_i) \left(\begin{array}{c|c} \mathbf{F}_i' & \mathbf{B}_i' \end{array}\right) \left(\frac{\vec{U}_{i+1}(0)}{\vec{U}_0(0)}\right) \\
&= \left(\begin{array}{c|c} \mathbf{\Pi}_i(\Delta c_i) & \bar{\mathbf{\Pi}}_i^a(\Delta c_i) \\ \hline \mathbf{0} & \mathbf{I} \end{array}\right) \left(\begin{array}{c|c} \mathbf{F}_i & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{I} \end{array}\right) \left(\frac{\vec{U}_{i+1}(0)}{\vec{U}_0(0)}\right) \\
&= \left(\begin{array}{c|c} \mathbf{\Pi}_i(\Delta c_i)\mathbf{F}_i & \bar{\mathbf{\Pi}}_i^a(\Delta c_i) \\ \hline \mathbf{0} & \mathbf{I} \end{array}\right) \left(\frac{\vec{U}_{i+1}(0)}{\vec{U}_0(0)}\right) \\
&= \left(\frac{\mathbf{\Pi}_i(\Delta c_i)\mathbf{F}_i\vec{U}_{i+1}(0) + \bar{\mathbf{\Pi}}_i^a(\Delta c_i)\vec{U}_0(0)}{\vec{U}_0(0)}\right) \ .
\end{aligned}$$

As a result we can represent $\vec{U}_i(0)$ in the following matrix form

$$\vec{U}_i(0) = \mathbf{\Pi}_i(\Delta c_i)\mathbf{F}_i\vec{U}_{i+1}(0) + \bar{\mathbf{\Pi}}_i^a(\Delta c_i)\vec{U}_0(0)$$

by noting that $\mathbf{\Pi}_i$ is formed by the first k_i rows and columns of matrix $\mathbf{\Pi}_i^a$, and $\bar{\mathbf{\Pi}}_i^a$ is formed by the first k_i rows and the last $k_i^a - k_i$ columns of $\mathbf{\Pi}_i^a$.

For the case $i = m$, i.e., the last graph $\mathcal{G}_m$, the region size is infinite, therefore delay transitions do not exist. The vector $\vec{U}_m(x+\tau)$ in $\int_0^\infty \hat{\mathbf{M}}_m(\tau)\vec{U}_m(x+\tau)d\tau$ does not depend on entering time x, therefore we can take it out of the integral. As a result we obtain $\int_0^\infty \hat{\mathbf{M}}_m(\tau)d\tau \cdot \vec{U}_m(0)$. Moreover, $\int_0^\infty \hat{\mathbf{M}}_m(\tau)d\tau$ boils down to $\hat{\mathbf{P}}_m$ and $\int_0^\infty \mathbf{B}_m(\tau)d\tau$ to $\hat{\mathbf{B}}_m$. Also we add the vector $\vec{1}_F$ to ensure that the probability to start from a state in V_F is one (see Eq. (7.8)). ■

Since the coefficients of the linear equations are all known, solving the system of linear equations yields $\vec{U}_0(0)$, which contains the probability $Prob_{v_0}(0)$ of reaching V_F from initial vertex v_0.

Now we explain how Eq. (7.25) is derived from Eq. (7.16)-(7.18). The term $\mathbf{\Pi}_i(\Delta c_i) \cdot \mathbf{F}_i\vec{U}_{i+1}(0)$ is for the delay transitions, where $\mathbf{F}_i$ specifies how the delay transitions are connected between $\mathcal{G}_i$ and $\mathcal{G}_{i+1}$. The term $\bar{\mathbf{\Pi}}_i^a(\Delta c_i) \cdot \vec{U}_0(0)$ is for Markovian transitions with reset. $\bar{\mathbf{\Pi}}_i^a(\Delta c_i)$ in the augmented CTMC $\mathcal{C}_i^a$ specifies the probabilities to first take transitions inside $\mathcal{G}_i$ and then a one-step Markovian transition back to $\mathcal{G}_0$. Eq. (7.26) is derived from Eq. (7.19). Since it is the last region and time goes to infinity, the time to enter the region is irrelevant (thus set to 0). Thus $\int_0^\infty \hat{\mathbf{M}}_i(\tau)d\tau$ boils down to $\hat{\mathbf{P}}_i$. In fact, the Markovian jump probability inside $\mathcal{G}_m$ can be taken from the embedded DTMC of $\mathcal{C}_m$, which is $\hat{\mathbf{P}}_i$.

Example 7.4.4 For the single-clock DMTA in Fig. 7.3(a), we show how to compute the reachability probability $Prob((v_0,0),(v_5,\cdot))$ on the region graph $\mathcal{G}$ (cf. Fig. 7.3(b)), which has been partitioned into subgraphs $\mathcal{G}_0$, $\mathcal{G}_1$ and $\mathcal{G}_2$. We mainly show how to obtain the system of linear equations according to Thm. 7.4.3. We first collect the matrices for subgraphs $\mathcal{G}_0$, $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively.

The matrices for $\mathcal{G}_0$ are given as

$$\mathbf{M}_0(x) = \begin{pmatrix} 0 & 1{\cdot}r_0{\cdot}e^{-r_0x} & 0 \\ 0.5{\cdot}r_1{\cdot}e^{-r_1x} & 0 & 0.2{\cdot}r_1{\cdot}e^{-r_1x} \\ 0 & 0 & 0 \end{pmatrix} \quad \mathbf{F}_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

The matrices for $\mathcal{G}_1$ are given as

$$\mathbf{M}_1(x) = \begin{pmatrix} 0 & r_0{\cdot}e^{-r_0x} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & r_2{\cdot}e^{-r_2x} \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad \mathbf{F}_1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \mathbf{B}_1 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{M}_1^a(x) = \begin{pmatrix} 0 & r_0{\cdot}e^{-r_0x} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5{\cdot}r_1{\cdot}e^{-r_1x} & 0 & 0.2{\cdot}r_1{\cdot}e^{-r_1x} \\ 0 & 0 & 0 & r_2{\cdot}e^{-r_2x} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & r_0{\cdot}e^{-r_0x} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & r_1{\cdot}e^{-r_1x} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & r_2{\cdot}e^{-r_2x} \end{pmatrix}$$

The matrices for $\mathcal{G}_2$ are given as

$$\hat{\mathbf{M}}_2(x) = \begin{pmatrix} 0 & r_2{\cdot}e^{-r_2x} \\ 0 & 0 \end{pmatrix} \quad \hat{\mathbf{P}}_2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

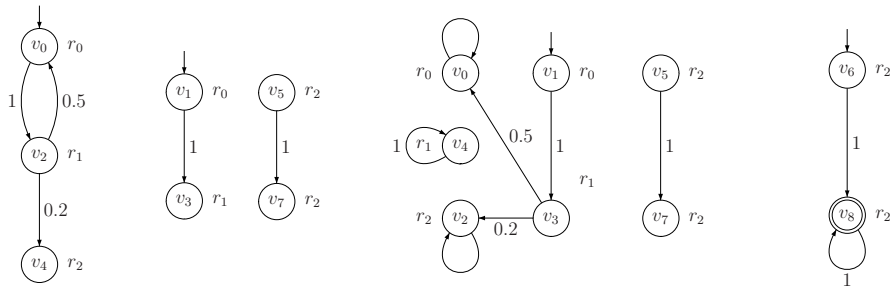
To obtain the system of linear equations, we need:

$$\mathbf{\Pi}_0(1) = \begin{pmatrix} p_{00} & p_{02} & p_{04} \\ p_{20} & p_{22} & p_{24} \\ p_{40} & p_{42} & p_{44} \end{pmatrix} \mathbf{\Pi}_1(1) = \begin{pmatrix} p_{11} & p_{13} & p_{15} & p_{17} \\ p_{31} & p_{33} & p_{35} & p_{37} \\ p_{51} & p_{53} & p_{55} & p_{57} \\ p_{71} & p_{73} & p_{75} & p_{77} \end{pmatrix} \bar{\mathbf{\Pi}}_1^a(1) = \begin{pmatrix} \bar{p}_{10} & \bar{p}_{12} & \bar{p}_{14} \\ \bar{p}_{30} & \bar{p}_{32} & \bar{p}_{32} \\ \bar{p}_{50} & \bar{p}_{52} & \bar{p}_{54} \\ \bar{p}_{70} & \bar{p}_{72} & \bar{p}_{74} \end{pmatrix}$$

All elements in these $\mathbf{\Pi}$-matrices can be computed by the transient probability in the corresponding CTMCs $\mathcal{C}_0$, $\mathcal{C}_1$ and $\mathcal{C}_1^a$ (cf. Fig. 7.7). The obtained system of linear equations is thus

$$\begin{bmatrix} u_0 \\ u_2 \\ u_4 \end{bmatrix} = \begin{pmatrix} p_{00} & p_{02} & p_{04} \\ p_{20} & p_{22} & p_{24} \\ p_{40} & p_{42} & p_{44} \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{bmatrix} u_1 \\ u_3 \\ u_5 \\ u_7 \end{bmatrix}$$

$$\begin{bmatrix} u_1 \\ u_3 \\ u_5 \\ u_7 \end{bmatrix} = \begin{pmatrix} p_{11} & p_{13} & p_{15} & p_{17} \\ p_{31} & p_{33} & p_{35} & p_{37} \\ p_{51} & p_{53} & p_{55} & p_{57} \\ p_{71} & p_{73} & p_{75} & p_{77} \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot \begin{bmatrix} u_6 \\ u_8 \end{bmatrix} + \begin{pmatrix} \bar{p}_{10} & \bar{p}_{12} & \bar{p}_{14} \\ \bar{p}_{30} & \bar{p}_{32} & \bar{p}_{32} \\ \bar{p}_{50} & \bar{p}_{52} & \bar{p}_{54} \\ \bar{p}_{70} & \bar{p}_{72} & \bar{p}_{74} \end{pmatrix} \cdot \begin{bmatrix} u_0 \\ u_1 \\ u_3 \end{bmatrix}$$



(a) $\mathcal{C}_0$ (b) $\mathcal{C}_1$ (c) $\mathcal{C}_1^a$ (d) $\mathcal{C}_2$

Figure 7.7: CTMCs

$$\begin{bmatrix} u_6 \\ u_8 \end{bmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \cdot \begin{bmatrix} u_6 \\ u_8 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

This system of linear equations can be solved easily.

Remark 7.4.5 We note that for two-clock DTA which yield two-clock DMTA, the approach given in this section fails in general. In the single-clock case, the reset guarantees to jump to $\mathcal{G}_0(0)$, and the delay to $\mathcal{G}_{i+1}(0)$ when it is in $\mathcal{G}_i$. However, in the two-clock case, after delay or reset generally only one clock has a fixed value, while the value of the other one is not determined.

The time-complexity of computing the reachability probability in the single-clock DTA case is $\mathcal{O}(m \cdot |S|^2 \cdot |Loc|^2 \cdot \lambda \cdot \Delta c + m^3 \cdot |S|^3 \cdot |Loc|^3)$, where m is the number of constants appearing in DTA, $|S|$ is the number of states in the CTMC, $|Loc|$ is the number of locations in the DTA, λ is the maximal exit rate in the CTMC and $\Delta c = \max_{0 \leq i < m}\{c_{i+1} - c_i\}$. The first term $m \cdot |S|^2 \cdot |Loc|^2 \cdot \lambda \cdot \Delta c$ is due to the uniformization technique for computing transient distribution; and the second term $m^3 \cdot |S|^3 \cdot |Loc|^3$ is the time complexity for solving a system of linear equations with $\mathcal{O}(m \cdot |S| \cdot |Loc|)$ variables. Note that from this analysis we can conclude that a PTIME procedure is obtained. However, this does *not* imply that the problem of model checking single-clock DTA is in PTIME, as our procedure involves computation over real numbers in general and thus can only be done in an approximated way .

7.5 Conclusion

We considered the quantitative verification of a CTMC $\mathcal{C}$ against a DTA $\mathcal{A}$. As a key result, we obtained that computing the probability of $\mathcal{C} \models \mathcal{A}$ can be reduced to computing reachability probabilities in PDPs. For single-clock DTA, this reduces to solving a system of linear equations yielding an equivalent, though simpler, characterization than in [DHS09].

Many problems remain open. How to solve the obtained system of integral equations and ODEs and their (theoretical) complexity (in terms of both Turing computation model and Blum-Shub-Smale computation model [BSS89, BCSS98]) are clearly interesting topics of further investigation; providing took support is also in the plan. Moreover, note that for DTA, the fairness issue (e.g. Büchi accepting condition) has not been addressed in this chapter (we essentially considered the reachability), which deserves future exploration. Other great challenges include non-deterministic TA and M(I)TL model checking.

Chapter 8

Probabilistic Time-abstracting Bisimulations for PTA

8.1 Introduction

In this chapter, we study *probabilistic timed automata* (PTA) [KNSS02], which are a probabilistic extension of timed automata (TA). As in TA, in the research on PTA, the notion of *region graphs* plays an essential role, see e.g. [KNSS02]. However, it is well-recognized that albeit being a very useful tool for theoretical purposes, the region graph is too large to be of any practical interest: its size is exponential in the number of clocks of the system as well as in the size of the constants in the time constraints. To overcome this explosion, inspired by [TY01], we propose *probabilistic time-abstracting bisimulations* (PTABs) for PTA, where the passage of arbitrary time is abstracted by a τ-transition. To deal with the probabilistic aspect, we follow a standard approach due to [LS91]. This equivalence is usually much coarser than the region equivalence, therefore, in practice, it induces a (potentially) much smaller state space partition. In particular, the region equivalence constitutes a (very fine) probabilistic time-abstracting bisimulation. The *bisimulation quotient* is a finite-state Markov decision process (MDP), where the states are equivalence classes over *symbolic states* (sets of states) with either τ or discrete probabilistic transitions.

PTAB is particularly useful when the desired properties do not involve time constraints. Such properties are very common in practice, i.e., safety, reachability, etc. Those properties can be captured well by the *probabilistic computation tree logic* (PCTL, [HJ94]), which is proven to be preserved under PTABs. In this case, the existing algorithms and tools for MDPs with respect to PCTL [BdA95, KNP04] can thus be applied for PTA.

To obtain a minimal PTAB quotient, our algorithm works in the *partition-refinement* fashion [PT87, KS90]. We start from an initial partition that respects state labeling, and proceed by refining each block till it contains only bisimilar states. Due to the fact that PTA involve an interweaving of time, nondeterminism and probability distributions, the minimization has thus to deal with the following difficulties: When taking a τ-transition, it must guarantee that time

traverses continuously, e.g., time cannot jump from 0 to 2 without traversing 1. Thus, we introduce the *timed predecessor* set as a splitter, as in [TY01]. Moreover, since a discrete transition results in one or more probability distributions, the splitter of only one block in [TY01] is, however, not applicable. Our algorithm instead adopts the idea of the *mutual-refine* technique in [BEM00], which maintains a state partition and a distribution partition. In each refinement iteration, the distribution partition is used to refine a state partition and vice versa. The algorithm in [BEM00], unfortunately, cannot be applied in our setting in a straightforward way, as in that case *untimed* probabilistic models (e.g. MDPs, probabilistic automata) are addressed and thus the set of target distributions is always fixed, while in our case, since the number of symbolic states in a block might grow with the iteration when time comes into play, both the symbolic state space and the distribution set vary in each round, let alone their partitions. To solve this problem, an *Expand* operator is introduced, recalculating the symbolic states in a block as well as the distribution set before the mutual-refine technique is applied. This algorithm is symbolic, namely, equivalence classes are symbolic states and set-theoretic operators are used to compute the set of predecessor states of a symbolic state.

Related work. Besides [TY01], for (non-probabilistic) timed systems, time-abstracting bisimulation is also studied in [LY97] in the setting of real-time process calculi. [BEM00, DHS03] present algorithms for probabilistic bisimulation and simulation of discrete probabilistic systems. [LMST03] investigates weak probabilistic bisimulation for PTA with a decision procedure. That algorithm is region based, which is tried to be avoided in the current chapter. [KNSS02] presents a comprehensive exposition for PTA and model checking algorithms for PTA. [KNSW07] gives a symbolic algorithm for model checking, however, the problem of computing time-abstracting bisimulations (or generally computing bisimulations) is not considered there.

Structure of the chapter. Section 8.2 presents basic definitions regarding PTA. Section 8.3 defines the PTAB. Section 8.4 presents the bisimulation minimization algorithm and constitutes the core of this chapter. Section 8.5 shows that the bisimulation preserves PCTL formulae. This chapter is concluded in Section 8.6.

8.2 Preliminaries

The basic notions regarding probability distributions, clock variables and clock constraints can be found in Section 2.2.

8.2.1 Probabilistic Timed Automata

Definition 8.2.1 (Probabilistic timed automata [KNSS02]) A *probabilistic timed automaton* (PTA) is a tuple $\mathcal{G} = (Loc, \mathcal{X}, \ell_0, L, Inv, \rightsquigarrow)$ where:

- Loc is a finite set of *locations* with $\ell_0 \in Loc$ the initial location;
- $\mathcal{X}$ is a set of clocks;
- $L : Loc \to 2^{\mathrm{AP}}$ is a labeling function for the locations, where AP denotes a finite set of atomic propositions;
- $\rightsquigarrow \subseteq Loc \times \mathcal{B}(\mathcal{X}) \times Distr(2^{\mathcal{X}} \times Loc)$ is an edge relation; and
- $Inv : Loc \to \mathcal{B}(\mathcal{X})$ is an invariant-assignment function.

Recall that $\mathcal{B}(\mathcal{X})$ is the set of clock constrains, as defined in Section 2.2.3. For simplicity, we require that the invariants and enabling conditions in the transitions are subject to the following conventions. We note that (1) and (2) are used to ensure the soundness of the semantic model (cf. Def. 8.2.6), while (3) is a common assumption in the literature.

1. If in some state of $\mathcal{G}$, allowing any amount of time to elapse would violate the invariant of the current location, then the guard of at least one discrete transition is satisfied. Namely, in each state (ℓ, ν), if $\nu + t \not\models Inv(\ell)$ for any $t \in \mathbb{R}_{>0}$, then there must exist some edge from ℓ with the guard g such that $\nu \models g$;
2. It is never possible to perform a discrete transition to a node for which the invariant is not satisfied by the current values of the clocks; and
3. All invariants are *downward-closed* in the sense that for any $d \in \mathbb{R}_{\geq 0}, \nu + d \models Inv(\ell)$ implies that $\nu \models Inv(\ell)$.

The labeling function L associates to each location ℓ a set of atomic propositions that are valid in ℓ. The system starts in location ℓ_0 with all its clocks initialized to 0. The values of all the clocks increase uniformly with time. We refer to $\ell \overset{g}{\rightsquigarrow} \eta$ as a *transition*, where the *guard* g is a clock constraint on the clocks of $\mathcal{G}$ and η is a distribution over the (X, ℓ) pairs with $X \subseteq \mathcal{X}$ a set of clocks to be reset and ℓ the successor location. The intuition is that the PTA $\mathcal{G}$ can move from location ℓ to location ℓ' via two phases. In the first phase, a distribution η is nondeterministically chosen when g holds. In the second phase, a successor location ℓ' is probabilistically chosen according to $\eta(X, \ell')$, where the clocks in X should be reset when entering ℓ'. The function Inv assigns to each ℓ a location invariant that constrains the amount of time that may be spent in ℓ. In other words, location ℓ must be left before the invariant $Inv(\ell)$ becomes invalid. If there is no outgoing discrete transition enabled, the corresponding state has a (discrete) *deadlock*.

Example 8.2.2 Fig. 8.1 is an example PTA, where from ℓ_1 there are two distributions (or transitions), and thus the PTA is nondeterministic. The transitions to ℓ_3 and ℓ_4 share the same guard $x > 1$, since they belong to the same distribution. The transition to ℓ_3 resets the clock $\{x\}$. The labeling on ℓ_1 and ℓ_3 is $\{a\}$, $\emptyset$ otherwise.

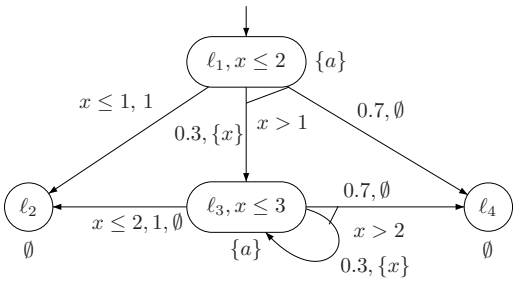


Figure 8.1: An example PTA

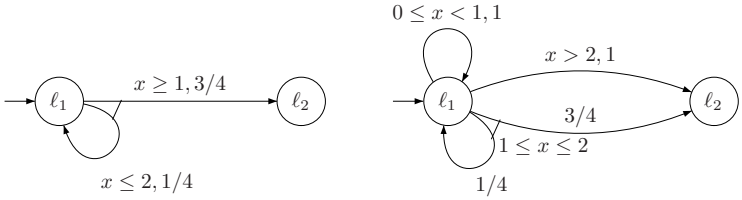


Figure 8.2: The encoding to a one-clock-constraint model

Remark 8.2.3 According to the syntax , one transition of a PTA is associated with a single clock constraint. This requirement is intuitive and reasonable since the more-than-one clock constraint case, see e.g., [Bea03, LMST03], can be encoded by adding more distributions. To give an example, the left-hand automaton in Fig. 8.2 is a PTA with two clock constraints in one distribution. This can be encoded by the PTA on the right in Fig. 8.2. It goes as follows: When $0 \leq x < 1$, the transition from ℓ_1 to ℓ_2 is not enabled. Thus the only possible transition is the self-loop on ℓ_1, which is normalized to probability 1. When $1 \leq x \leq 2$, both transitions are enabled, and their probabilities remain the same. The $x > 2$ case is similar as $0 \leq x < 1$.

8.2.2 Probabilistic Timed Structures

The semantics of a timed automaton is an (infinite) timed transition system. The semantics of a PTA is provided by a *probabilistic timed structure*, which is essentially an infinite MDP.

Definition 8.2.4 A *probabilistic timed structure* (PTS) $\mathcal{M}$ is a *labeled Markov decision process* $(S, \mathit{Steps}, L, s_0)$ where:

- S is a set of states with $s_0 \in S$ being the initial state;
- $\mathit{Steps} : S \to 2^{\mathbb{R}_{\geq 0} \times \mathit{Distr}(S)}$ is a function that assigns to each state $s \in S$ a set of pairs (t, μ) where $t \in \mathbb{R}_{\geq 0}$ and $\mu \in \mathit{Distr}(S)$; and

- $L : S \to 2^{\mathrm{AP}}$ is a state labeling function.

$\mathit{Steps}(s)$ is the set of transitions that can be nondeterministically chosen in state s. The transition labels are of the form (t, μ) where t is the *duration* of the transition and μ is the probability distribution over the successor states. $s \xrightarrow{t,\mu} s'$ means that after t time units have elapsed, a transition is fired from s to s' with probability $\mu(s')$.

Paths. Paths in a PTS arise by resolving both the nondeterministic and probabilistic choices. A path of the PTS $\mathcal{M} = (S, \mathit{Steps}, L, s_0)$ is a finite or infinite sequence:

$$\omega = s_0 \xrightarrow{t_0,\mu_0} s_1 \xrightarrow{t_1,\mu_1} s_2 \xrightarrow{t_2,\mu_2} \ldots$$

where $s_i \in S$, $(t_i, \mu_i) \in \mathit{Steps}(s_i)$ and $\mu_i(s_{i+1}) > 0$ for all $0 \leq i \leq |\omega|$, where $|\omega|$ is the number of transitions in ω. For $0 \leq i \leq |\omega|$, $\omega[i]$ denotes the $(i{+}1)$-th state of ω (i.e., $\omega[i] = s_i$) and $\mathit{step}(\omega, i)$ (or $\mathit{step}(\omega[i])$) denotes the $(i{+}1)$-th transition (t_i, μ_i). A finite path ω ends in a state, denoted by $\mathit{last}(\omega)$. We write $\mathit{Paths}^\star$ to denote the set of finite paths and $\mathit{Paths}^\star(s)$ the set of finite paths that start from state s. Paths^ω and $\mathit{Paths}^\omega(s)$ are the counterparts for infinite paths.

Definition 8.2.5 A *scheduler* of a PTS $\mathcal{M} = (S, \mathit{Steps}, L, s_0)$ is a function $\mathfrak{G}$ mapping every finite path ω of $\mathcal{M}$ to a pair (t, μ) such that $\mathfrak{G}(\omega) \in \mathit{Steps}(\mathit{last}(\omega))$. Let $\mathfrak{W}$ be the set of all schedulers of $\mathcal{M}$.

A scheduler resolves the nondeterminism by choosing a probability distribution based on the process executed so far. Formally, if a PTS is guided by scheduler $\mathfrak{G}$ and has the finite path ω as its *history*, then it will be in state s in the next step with probability $\mu(s)$ after t time units, where $\mathfrak{G}(\omega) = (t, \mu)$. We write $\mathit{Paths}^\omega_{\mathfrak{G}}$ to denote the set of infinite paths induced by a given scheduler $\mathfrak{G}$, i.e., $\mathit{Paths}^\omega_{\mathfrak{G}} = \{\omega \in \mathit{Paths}^\omega \mid \mathfrak{G}(\omega{\downarrow_i}) = \mathit{step}(\mathit{last}(\omega{\downarrow_i})) \text{ for } i \geq 0\}$, where $\omega{\downarrow_i}$ returns the prefix of ω up to length i. $\mathit{Paths}^\omega_{\mathfrak{G}}(s)$ is defined as $\mathit{Paths}^\omega_{\mathfrak{G}} \cap \mathit{Paths}^\omega(s)$.

Scheduler $\mathfrak{G}$ on PTS $\mathcal{M}$ induces a discrete-time Markov chain (DTMC) $\mathcal{M}^{\mathfrak{G}}$, where the nondeterminism has been resolved. Each state in $\mathcal{M}^{\mathfrak{G}}$ is a finite path fragment ω in $\mathcal{M}$. Formally $\mathcal{M}^{\mathfrak{G}} = (\mathit{Paths}^\star_{\mathfrak{G}}, \mathbf{P}^{\mathfrak{G}})$ is a DTMC where

$$\mathbf{P}^{\mathfrak{G}}(\omega, \omega') = \begin{cases} \mu(s) & \text{if } \mathfrak{G}(\omega) = (t, \mu) \text{ and } \omega' = \omega \xrightarrow{t,\mu} s \\ 0 & \text{otherwise .} \end{cases} .$$

The definition of probability space of $\mathcal{M}^{\mathfrak{G}}$ can be found in Section 2.2.2.

8.2.3 Obtaining a PTS from a PTA

Any PTA can be interpreted as a PTS. Due to the continuous nature of clocks, these underlying PTSs in general have infinitely many states (even uncountably many), and are infinitely branching. PTA can thus be considered as a *finite* description of infinite PTSs.

Given a PTA $\mathcal{G} = (Loc, \mathcal{X}, \ell_0, L, Inv, \rightsquigarrow)$, a *state* of $\mathcal{G}$ is a pair (ℓ, ν), where $\ell \in Loc$ is a location and ν is a valuation satisfying the invariant of ℓ.

Definition 8.2.6 (PTS semantics of a PTA) Let $\mathcal{G} = (Loc, \mathcal{X}, \ell_0, L, Inv, \rightsquigarrow)$ be a PTA. The PTS of $\mathcal{G}$ is $\mathcal{M}_\mathcal{G} = (S, Steps, L', s_0)$ with:

- $S = \{(\ell, \nu) \mid \nu \models Inv(\ell), \ell \in Loc\}$ with $s_0 = (\ell_0, \vec{0})$;
- $L'((\ell, \nu)) = L(\ell)$;
- *Steps* assigns to each state in S a set of transitions, which are defined in two ways. Namely, for each state $(\ell, \nu) \in S$,
  - *discrete transition*: $(\ell, \nu) \xrightarrow{0,\mu} (\ell', \nu')$, if the following conditions hold:
    1. $\exists$ transition $\ell \overset{g}{\rightsquigarrow} \eta$ in $\mathcal{G}$ with $\eta(\ell', X) > 0$;
    2. $\nu \models g$;
    3. $\nu' = \nu[X := 0]$; and
    4. $\mu(\ell', \nu') = \sum_{X \subseteq \mathcal{X}, \nu' = \nu[X:=0]} \eta(X, \ell')$.

    Usually, we simply write $(\ell, \nu) \rightarrow \mu$.
  - *delay transition*: $(\ell, \nu) \xrightarrow{d,\mathbf{1}} (\ell, \nu + d)$ for all $0 \leq d \leq t$, if $\nu + d \models Inv(\ell)$.
    Note that $\mathbf{1}$ indicates that the probability distribution is $\mu^1_{(\ell, \nu+d)}$.
    Usually, we simply write $(\ell, \nu) \xrightarrow{d} (\ell, \nu + d)$.

Formally, for any state (ℓ, ν) in $\mathcal{M}_\mathcal{G}$, we say that (ℓ, ν) has a (discrete) deadlock if $Steps((\ell, \nu))$ does *not* contain any discrete transition.

Symbolic states. We define symbolic states which are used for the effective representation and manipulation of the infinite state space of PTS. Generally, a symbolic state denotes a set of states of $\mathcal{M}_\mathcal{G}$.

In a nutshell, a *zone* [HNSY94] $Z \subseteq \mathbb{R}^\mathcal{X}$ of $\mathcal{X}$ is a set of valuations which satisfy a conjunction of clock constraints. Formally, the zone for the constraint g is $Z = \{\nu \mid \nu \models g, x \in \mathcal{X}\}$. Geometrically, a zone is a polyhedron (note that in this chapter we do not require a zone to be convex). A *symbolic state* $\mathbf{S}$ is a set of states whose clock evaluations form the same zone. Strictly speaking, $\mathbf{S}$ is a set of pairs of *location* and *zone*, namely, it is of the form (ℓ, Z). The union of all symbolic states is the state space S.

8.3 Probabilistic Time-abstracting Bisimulations

In order to refine the dense state space as much as possible, we adopt the time-abstracting bisimulation [TY01] for state space minimization, which abstracts from the quantitative aspect of time: we know that *some* time passes, but not how much. We first introduce a technical definition:

Definition 8.3.1 $\mu, \mu' \in \mathit{Distr}(S)$ are equivalent with respect to equivalence $\mathcal{R}$ on S, written $\mu \equiv_{\mathcal{R}} \mu'$, if $\forall U \in S/\mathcal{R}.\ \mu(U)=\mu'(U)$.

Definition 8.3.2 (Probabilistic time-abstracting bisimulation) Let $\mathcal{G}$ be a PTA and $\mathcal{M}_{\mathcal{G}} = (S, \mathit{Steps}, L', s_0)$ the PTS of $\mathcal{G}$. A *probabilistic time-abstracting bisimulation* (PTAB) for $\mathcal{G}$ is an equivalence relation $\mathcal{R}$ over the state space S of $\mathcal{M}_{\mathcal{G}}$ such that for all states $(s_1, s_2) \in \mathcal{R}$, the following conditions hold:

- $L'(s_1) = L'(s_2)$;
- If $s_1 \xrightarrow{t_1} s_1'$, for some $t_1 \in \mathbb{R}_{>0}$, then there exist $t_2 \in \mathbb{R}_{>0}$ and $s_2' \in S$ such that $s_2 \xrightarrow{t_2} s_2'$ and $(s_1', s_2') \in \mathcal{R}$; and
- If $s_1 \to \mu_1$, for some $\mu_1 \in \mathit{Distr}(S)$, then there exists some $\mu_2 \in \mathit{Distr}(S)$ such that $s_2 \to \mu_2$ and $\mu_1 \equiv_{\mathcal{R}} \mu_2$.

s_1 and s_2 are *probabilistic time-abstracting bisimilar*, denoted by $s_1 \sim s_2$, if $(s_1, s_2) \in \mathcal{R}$ for some PTAB $\mathcal{R}$.

We use τ-transitions to abstract the exact time passage away, formally, $s \xrightarrow{\tau} s'$ iff $\exists t \in \mathbb{R}_{>0}.s \xrightarrow{t} s'$.

It is straightforward to verify that $\sim$ is an equivalence. The rest of Section 8.3 is devoted to showing the properties of PTAB.

8.3.1 Region Equivalences

In the following, we first recall the definition and properties of the region equivalence [AD94, ACD93], which is essential in turning the infinite state space of a PTS into a *finite* quotient. Later we will show a similar result as in [TY01] that the region equivalence for PTS is in fact a PTAB.

Given a PTA $\mathcal{G}$, let $c = c_{\max}(\mathcal{G})$, the largest integer constant among all the clock constraints and invariants in $\mathcal{G}$. Two clock evaluations ν and ν' are *region equivalent* (with respect to c), denoted by $\nu \cong \nu'$, iff they satisfy:

- $\forall x \in \mathcal{X}$, either $\lfloor \nu(x) \rfloor = \lfloor \nu'(x) \rfloor$ or both $\nu(x) > c$ and $\nu'(x) > c$; and
- $\forall x, y \in \mathcal{X}$, either $\lfloor \nu(x) - \nu(y) \rfloor = \lfloor \nu'(x) - \nu'(y) \rfloor$ or both $\lfloor \nu(x) - \nu(y) \rfloor > c$ and $\lfloor \nu'(x) - \nu'(y) \rfloor > c$.

Note that $\lfloor r \rfloor$ is the floor function that returns, for $r \in \mathbb{R}$, the maximal integer that is at most r. The equivalence classes induced by $\cong$ are called *regions*. The region equivalence can be lifted to states of PTA such that $(\ell, \nu) \cong (\ell', \nu')$ if $\ell = \ell'$ and $\nu \cong \nu'$. The region equivalence enjoys the following properties:

Lemma 8.3.3 For valuations $\nu, \nu' \in \mathbb{R}^{\mathcal{X}}$ with $\nu \cong \nu'$:

1. for any zone Z, $\nu \in Z$ iff $\nu' \in Z$;

2. for any set of clocks $X \subseteq \mathcal{X}$, $\nu[X := 0] \cong \nu'[X := 0]$; and
3. $\forall d \geq 0 \exists d' \geq 0.\ \nu + d \cong \nu' + d'$.

Theorem 8.3.4 The region equivalence is a PTAB, i.e., $\cong \subseteq \sim$.

Proof: Let $(\ell, \nu), (\ell, \nu')$ be two states in PTS $\mathcal{M} = (S, \mathit{Steps}, L', s_0)$ such that $(\ell, \nu) \cong (\ell, \nu')$.

- (Labels) Since $L'((\ell, \nu)) = L(\ell)$, where L is the labeling function in the corresponding PTA $\mathcal{G}$, we have $L'((\ell, \nu)) = L'((\ell, \nu'))$.
- (Timed transition) Let $(\ell, \nu) \xrightarrow{d} (\ell, \nu + d)$ be a timed transition. Due to Lem. 8.3.3(3), there exists a $d' \geq 0$ such that $\nu + d \cong \nu' + d'$. We have that $\nu', \nu' + d' \models \mathit{Inv}(\ell)$ since $\nu, \nu + d \models \mathit{Inv}(\ell)$. For any $d'' < d'$, $\nu + d'' \models \mathit{Inv}(\ell)$, by the downward-closedness of $\mathit{Inv}(\ell)$. Thus $(\ell, \nu') \xrightarrow{d'} (\ell, \nu' + d')$ is also a timed transition.
- (Probabilistic transition) Let $(\ell, \nu) \rightarrow \mu$ be a probabilistic transition. μ is chosen by some scheduler $\mathfrak{G}$. As $\mathfrak{G}$ can only select enabled transitions, $\mu \models g$. Let $Z = \{\nu \mid \nu \models g\}$, clearly $\nu \in Z$. Since $(\ell, \nu) \cong (\ell, \nu')$, due to Lem. 8.3.3(1), $\nu' \in Z$, which means that $\nu' \models g$, thus μ is also enabled in (ℓ, ν'). Therefore, we can construct a scheduler $\mathfrak{G}'$ which chooses the same distribution as $\mathfrak{G}$. Since $\mu \equiv_{\cong} \mu$ due to Lem. 8.3.3(2), $(\ell, \nu') \rightarrow \mu$ is also a probabilistic transition.

The region equivalence satisfies all the conditions of being a PTAB, therefore $\cong \subseteq \sim$. ■

The above theorem asserts that the region equivalence is a (probably very refined) PTAB. Note that the converse does not hold in general. It can be the case that $(\ell, \nu) \sim (\ell', \nu')$ where $\ell \neq \ell'$ (see Ex. 8.4.5), however, $(\ell, \nu) \ncong (\ell', \nu')$.

The following result shows that deadlocks are preserved by $\sim$.

Proposition 8.3.5 If $(\ell, \nu) \sim (\ell', \nu')$, then (ℓ, ν) has a deadlock iff (ℓ', ν') has a deadlock.

Evidently, the converse does not hold.

Remark 8.3.6 Another desired property of timed systems is *timelock-free*. For TA, generally timelocks are states violating the time-progress requirement. Formally, a state s is a *timelock* if all infinite runs starting from s are zeno. A TA is timelock-free if none of its reachable states is a timelock. Similar notions can be adapted to PTA in an expected way: a state s is a *timelock* in PTA if with probability 1, all infinite runs starting from s are zeno; and the notion of timelock-free for PTA is defined accordingly. However, unfortunately, time-abstracting bisimulations do *not* preserve non-zenoness (or time-divergence) or timelockness. As

Figure 8.3: Counterexample regarding timelock from [TY01]

a matter of fact, this problem already occurs for (non-probabilistic) TA, as in [TY01], page 41, figure 11, a counterexample is given explicitly, which we reproduce as in Fig. 8.3. It is easy to see that $(\ell_1, 0) \sim (\ell_2, 0)$. However, $(\ell_1, 0)$ is a timelock while $(\ell_2, 0)$ is not. (We note that this is not surprising, since time-abstracting bisimulations are insensitive to exact delays.) We also mention that to handle the time-divergence problem, [TY01] basically makes the well-known *strongly non-zeno* assumption, namely, a minimal amount of time passes in every cyclic execution of the TA. We will not discuss this in depth since somehow it is impertinent to the main issue addressed in the current chapter.

8.4 Minimization of PTA

Having defined the PTAB, an immediate question is: how to compute it, since one of the crucial steps of exploiting PTAB for verification is to generate the quotient of the given PTA. A simple answer might be, taking the region graph, since the region equivalence is a PTAB! However, as pointed in [ACD93], since the number of regions grows exponentially with the number of clocks in the PTA, the finite region equivalence quotient is too large to be of any practical interest. In fact [KNSS02] shows that the number of regions for PTA is even higher than for TA. In other words, for the sake of efficiency, we are interested in the *minimal* quotient, namely, the one corresponding to the *coarsest* bisimulation. In what follows, we will propose an algorithm to compute the quotient of a PTS with respect to the coarsest PTAB, which combines the algorithm in [TY01] for TA and the algorithm in [BEM00] for MDPs.

8.4.1 Partition Refinement

Prior to presenting our algorithm, let us first recall how the minimization algorithm from [KS90, PT87] works for finite (non-probabilistic, untimed) labeled transition systems (LTSs). The algorithm relies on the *partition-refinement* technique [PT87]. Roughly speaking, the state space S is partitioned in *blocks*, i.e., pairwise disjoint sets of states. Starting from an initial partition Π_0 where, e.g., all equally-labeled states form a block, the algorithm successively refines these blocks such that ultimately each block contains only bisimilar states. The refinement is based on the fact that a bisimulation induces a pre-stable partition. Formally, given a partition of states Π and blocks $C_1, C_2 \in \Pi$, C_1 is *pre-stable* with respect to C_2 if $C_1 \subseteq \mathsf{pred}(C_2)$ or $C_1 \cap \mathsf{pred}(C_2) = \emptyset$, where $\mathsf{pred}(C)$ is the

set of direct predecessors of all the states in C. If C_1 is not stable with respect to C_2, then C_1 can further be partitioned into two sub-blocks $C_1 \cap \mathsf{pred}(C_2)$ and $C_1 \setminus \mathsf{pred}(C_2)$. In this case, C_2 is a *splitter* of C_1. Π is *pre-stable* if all its blocks are pairwise pre-stable. The main sketch of the algorithm presented in Algo. 1, albeit simple, is the essence of partition refinement.

Algorithm 1 The general partition-refinement algorithm

Require: The LTS, the initial partition Π_0
Ensure: The partition Π under the coarsest bisimulation
1: $\Pi := \Pi_0$;
2: **while** $(\exists C_1, C_2 \in \Pi,\ C_1$ is not stable with respect to $C_2)$ **do**
3: $\quad \Pi_{C_1} := \{C_1 \cap \mathsf{pred}(C_2), C_1 \setminus \mathsf{pred}(C_2)\}$;
4: $\quad \Pi := (\Pi \setminus \{C_1\}) \cup \Pi_{C_1}$;
5: **end while**
6: **return** Π;

The scheme can be adapted to infinite state spaces, assuming that they admit effective representations of blocks and decision procedures for computing intersection, set-difference and predecessors of blocks, and testing whether a block is empty. For termination, it must be ensured that a pre-stable partition always exists. In [TY01], such an adaption is given for TA to compute time-abstracting bisimulation, since the state space of TA falls in this category. In particular, two symbolic operations, i.e. time-pred and disc-pred, are introduced, which return the set of all discrete-predecessors and time-predecessors of states respectively. For details, see [TY01], Figure 15.

8.4.2 Bisimulation Quotienting Algorithm

In this section, we shall move further, taking probabilistic transitions into account. This is not trivial, since the infinite states (caused by time) and probabilistic transitions are closely interweaved, thus the set pred should be replaced by the *discrete predecessors* discpred and the *timed predecessors* timepred in a proper way.

The set of timed predecessors splits a block where a discontinuity on time occurs when taking a timed transition. This is captured by the *time-refinement operator* (see Def. 8.4.1). Besides, due to Prop. 8.3.5, a state having a deadlock must be in a different block than a state that does not suffer from a deadlock. This suggests a first *discrete-refinement operator* (see Def. 8.4.2).

For discrete predecessors, since a probability distribution rather than a state is associated with a transition, successively dividing a block by a single-block splitter does not suffice. Instead, we adapt the *mutual-refine* algorithm proposed in [BEM00]. The algorithm maintains a distribution partition in addition to a state partition, and in each iteration refines one partition by the other and vice versa, till both partitions stabilize. However, this algorithm cannot be directly applied in our case, since a block might expand in a new partition as the number of symbolic states in it may grow. Consequently, in a new partition,

it is possible that the distribution set differs from the one in the last iteration, and obviously the old distribution partition is obsolete. The *Expand operator* (see Def. 8.4.3) thus recalculates the symbolic states, the distribution set as well as the distribution partition, and as a final step in one iteration, a state partition is refined by the a second *discrete-refinement operator* (see Def. 8.4.4) using the newest distribution partition.

The algorithm is presented in Algo. 2. A detailed explanation follows.

Algorithm 2 The partition-refinement algorithm for PTA

Require: The PTA $\mathcal{G}$ and PTS $\mathcal{M}_{\mathcal{G}} = (S, \mathit{Steps}, L', s_0)$
Ensure: The partition Π under the coarsest PTAB
1: Initialization: Get the initial partition, $\Pi := \Pi_{\mathrm{AP}}$;
2: Partition Π according to $\mathit{Refine}_d^1(\Pi, \nabla)$.
3: **Repeat**
4: PHASE I – **Refine Π by discrete transitions:**
5: Choose some block $C \in \Pi$,
6: $\mathit{Expand}(C, \Pi)$;
7: Update the distribution set Distr';
8: Compute the equivalence class Distr'/Π;
9: Choose some $M \in \mathit{Distr}'/\Pi$;
10: $\Pi := \mathit{Refine}_d^2(\Pi, M)$;
11: PHASE II – **Refine Π by time delays:**
12: Choose some block $C \in \Pi$;
13: $\Pi := \mathit{Refine}_t(\Pi, C)$;
14: **until** Π does not change.
15: **return** Π;

Determining the initial partition

The initial partition of states $\Pi_{\mathrm{AP}} = S/\mathcal{R}_{\mathrm{AP}}$ is the AP-partition of S, where $\mathcal{R}_{\mathrm{AP}} = \{(s_1, s_2) \in S \times S \mid L(s_1) = L(s_2)\}$. Initially, the zone of symbolic state (ℓ, ν) is $\mathit{Inv}(\ell)$, thus on the symbolic state level,

$$\Pi_{\mathrm{AP}} = \big\{\{([\ell]_{\mathcal{R}_{\mathrm{AP}}}, \mathit{Inv}(\ell))\} \mid \ell \in \mathit{Loc}\big\} \ .$$

Refining partitions

In the rest of this section, we will concentrate on how to refine an existing partition. For reference convenience, given a PTA, we designate to each transition (leading to a distribution) a unique action name, and for each location ℓ, we denote $\nabla(\ell)$ as the set of all outgoing transitions from ℓ, which is ranged over by $\alpha, \beta, \cdots$. Let $\nabla = \bigcup_{\ell \in \mathit{Loc}} \nabla(\ell)$. Moreover, for each transition α, g_α and μ_α are the corresponding guard and the resulting distribution, respectively. As an example, there are four uniquely labeled transitions in the PTA in Fig. 8.1.

As we have two types of transitions, there are two types of splitters as well. For the timed transitions, the time-splitter is a block C_2, and the time-refinement operator is as follows:

Definition 8.4.1 (The time-refinement operator) Let Π be a partition of S and $C_1, C_2 \in \Pi$. Then:

$$\mathit{Refine}_t(C_1, C_2) = \{C_1 \cap \mathsf{timepred}(C_2), C_1 \setminus \mathsf{timepred}(C_2)\} \setminus \{\emptyset\} \ ,$$

where $\mathsf{timepred}(S) = \{s \mid \exists s' \in S, t \in \mathbb{R}_{>0}, s \xrightarrow{t} s'\}$.

We define $\mathit{Refine}_t(\Pi, C_2) = \bigcup_{C_1 \in \Pi} \mathit{Refine}_t(C_1, C_2)$.

This corresponds to PHASE II (line 11-13) in Algo. 2.

For the discrete transitions, the split consists of two steps. The first step is to differentiate the symbolic states that can fire a discrete transition from those that cannot. In this step, a splitter is the action set ∇, which refines a block as follows:

Definition 8.4.2 (The 1^{st} discrete-refinement operator) Let Π be a partition of S, ∇ be the action set, and $C \in \Pi$. Then:

$$\mathit{Refine}_d^1(C, \nabla) = \{C^+, C^-\} \setminus \{\emptyset\} \ ,$$

where $C^+ = \{(\ell, Z) \mid \exists \alpha \in \nabla(\ell), Z \subseteq g_\alpha \text{ and } (\ell, Z') \in C \text{ for some } Z'\}$ and $C^- = \{(\ell, Z) \mid \forall \alpha \in \nabla(\ell), Z \cap g_\alpha = \emptyset \text{ and } (\ell, Z') \in C \text{ for some } Z'\}$.

We define $\mathit{Refine}_d^1(\Pi, \nabla) = \bigcup_{C \in \Pi} \mathit{Refine}_d^1(C, \nabla)$.

All symbolic states in C^+ have an enabled discrete transition whereas any of them in C^- does not. Actually, C^- is the set of states that have a deadlock. This is used in line 2 of the algorithm. This operator has only to be performed once, because the further refinement won't change the fact of having a discrete transition. Thus, we place it in the initialization.

As the second step, we can further partition C^+ according to the distributions. Suppose the current partition $\Pi = \{C_1, ..., C_n\}$, $n \in \mathbb{N}$. For any block C_i, we can write $C_i = \{(\ell_i^1, Y_i^1), \ldots, (\ell_i^q, Y_i^q)\}$ with $C_i = \bigcup_{1 \le j \le q} (\ell_i^j, Y_i^j)$, and for any $1 \le h \ne k \le q$, $\ell_i^h \ne \ell_i^k$. For index $1 \le h \le m$ and $\alpha \in \nabla(\ell_i^h)$ such that $Y_i^h \subseteq g_\alpha$, we hope to derive the distributions induced by α.

However, it is possible that the resulting symbolic state of a transition bestrides different blocks, which leads to the problem that the probability $\mu(C)$ to a block C is not well defined. For instance, Fig. 8.4(a) illustrates a distribution from $\mathbf{S}_0 = (\ell, Z)$ to $\mathbf{S}_1 = (\ell_1, Z)$ and $\mathbf{S}_2 = (\ell_2, Z)$, where $\mathbf{S}_0 \in C_0$, $\mathbf{S}_2 \in C_2$ but $\mathbf{S}_1$ scatters in C_{11}, C_{12} and C_{13}, as in Fig. 8.4(b). Note that $\{Z_{11}, Z_{12}, Z_{13}\}$ is a partition of Z. The problem is that $\mu(C_{1i})$ can *not* be defined for $1 \le i \le 3$.

To solve this problem, we have to split a *symbolic state* in such a way that each sub-symbolic state has well-defined probabilistic transitions over the partition Π, as in Fig. 8.4(c). As a result of this split, the number of blocks stays

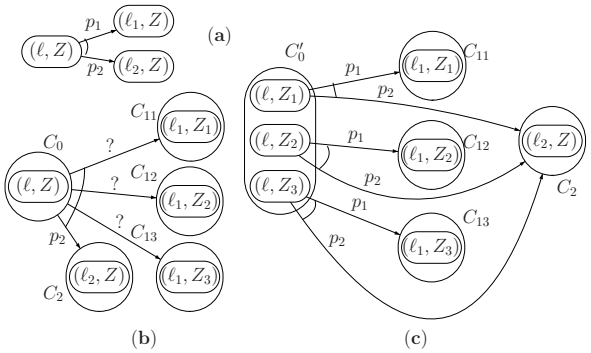


Figure 8.4: The motivation of *Expand*

the same, but the symbolic state space expands in terms of transitions. In the following, we define the *Expand* operator in a formal way.

For symbolic state $\mathbf{S} = (\ell, Z)$ and action α,

$$Supp(\mu_\alpha) = \{(\ell_1, X_1), \ldots, (\ell_m, X_m)\}$$

with probabilities $p_1, ..., p_m$, respectively, where $X_i \subseteq \mathcal{X}$ is the reset clock set, and p_i is the associated probability with $\sum_{1 \le i \le m} p_i = 1$. For successor (ℓ_j, X_j, p_j), the resulting symbolic state is $\mathbf{S}_j = (\ell_j, Z[X_j := 0])$. In the following, we will split Z into such a partition $\mathcal{Z} = \{Z_1, \ldots, Z_f\}$ that for any (sub)symbolic state (ℓ, Z') of $\mathbf{S}$, i.e., $Z' \in \mathcal{Z}$, each of its successor state is located only in one block. For $C_k \in \{C_1, ..., C_n\}$, define

$$Z_j^k = \left\{(\ell, \nu) \mid \nu \in Z,\ (\ell_j, \nu[X_j := 0]) \in C_k\right\} \ .$$

It is possible that $Z_j^k = \emptyset$. For each successor $1 \le j \le m$, $\{Z_j^1, Z_j^2, \ldots, Z_j^n\}$ is a partition of Z. We have the following partitions:

$$\begin{array}{ll} \text{For 1-st successor :} & \{Z_1^1, \ldots, Z_1^k, \ldots, Z_1^n\} \\ & \vdots \\ \text{For } j\text{-th successor :} & \{Z_j^1, \ldots, Z_j^k, \ldots, Z_j^n\} \\ & \vdots \\ \text{For } m\text{-th successor :} & \{Z_m^1, \ldots, Z_m^k, \ldots, Z_m^n\} \quad . \end{array}$$

We define

$$Z_{\vec{k}} = \bigcap_{1 \le j \le m} Z_j^{\vec{k}[j]} \ ,$$

where for each j, $1 \le \vec{k}[j] \le n$. $Z_j^{\vec{k}[j]}$ denotes choosing the $\vec{k}[j]$-th element in the j-th row, where $\vec{k}$ is a vector of indices. Stated in words, $Z_{\vec{k}}$ is obtained

by taking the intersection of one arbitrary element from each row in the above "matrix". It is not difficult to see that

$$\{Z_{\vec{k}} \mid 1 \leq \vec{k}[j] \leq n, 1 \leq j \leq m\} \setminus \{\emptyset\}$$

is a partition of Z, and in the worst case, this partition may contain n^m blocks.

For each $Z_{\vec{k}}$, since $Z_{\vec{k}} \subseteq Z_j^{\vec{k}[j]}$ for $1 \leq j \leq m$, it must be the case that $(\ell_j, Z_{\vec{k}}[X_j := 0]) \subseteq C_{\vec{k}[j]}$ for each $1 \leq j \leq m$. Hence, the probability from the symbolic state $(\ell, Z_{\vec{k}})$ to C_i for $1 \leq i \leq n$ is obtained by adding the nonzero probabilities in the i-th column:

$$\Pr\big((\ell, Z_{\vec{k}}), \alpha, C_i\big) = \sum_{1 \leq j \leq m, \vec{k}[j]=i} p_j \ .$$

In what follows, we merge those $Z_{\vec{k}}$ and $Z_{\vec{k}'}$ such that for each $C_i \in \Pi$, $\Pr\big((\ell, Z_{\vec{k}}), \alpha, C_i\big) = \Pr\big((\ell, Z_{\vec{k}'}), \alpha, C_i\big)$. As a result, the partition $\mathcal{Z} = \{Z_1, ..., Z_f\}$ is obtained. And the expansion operator expands a block with (possibly) more refined symbolic states as follows:

Definition 8.4.3 (The Expand operator) Let Π be a partition of S, $\alpha \in \nabla$, $C \in \Pi$ and $\mathbf{S} = (\ell, Z) \in C$. Then:

$$\mathit{Expand}(\mathbf{S}, \alpha, \Pi) = \{(\ell, Z_i) \mid 1 \leq i \leq f\} \ .$$

$\mathit{Expand}(C, \Pi) = \bigcup_{\mathbf{S} \in C, \alpha \in \nabla} \mathit{Expand}(\mathbf{S}, \alpha, \Pi)$.

Note that for each sub-symbolic state $\mathbf{T}$ of $\mathbf{S}$ in $\mathit{Expand}(\mathbf{S}, \alpha, \Pi)$, $\Pr(\mathbf{T}, \alpha, C_k)$ is well-defined. Let us denote $\mu_{\mathbf{T},\alpha}$ as the distribution over Π from $\mathbf{T}$ via action α. Now the distribution set is updated as $\mathit{Distr}' = \{\mu_{\mathbf{T},\alpha} \mid \mathbf{T} \in \mathit{Expand}(C, \Pi)$ with $\mathbf{T} = (\ell, Y)$ for some ℓ, Y and $\alpha \in \nabla(\ell)\}$. The distribution partition on Distr' over Π, denoted by Distr'/Π, can thus be updated consequentially, based on the following fact: Let $M \in \mathit{Distr}'/\Pi$, then $\forall \mu, \mu' \in M$, $\mu(C) = \mu'(C)$ for any $C \in \Pi$. As the mutual-refine technique, the state partition can in turn be refined by the distribution partition as follows:

Definition 8.4.4 (The 2^{nd} discrete-refinement operator) Let Π be a partition of S with $C \in \Pi$, $C' = \mathit{Expand}(C, \Pi)$ and $M \in \mathit{Distr}'/\Pi$. Then:

$$\mathit{Refine}_d^2(C, M) = \{C_M, C \setminus C_M\} \setminus \{\emptyset\} \ ,$$

where $C_M = \{\mathbf{T} \mid \mu_{\mathbf{T},\alpha} \in M$ and $\mathbf{T} = (\ell, Y), \alpha \in \nabla(\ell)\}$.

We define $\mathit{Refine}_d^2(\Pi, M) = (\Pi \setminus \{C\}) \cup \mathit{Refine}_d^2(C, M)$.

The above steps correspond to PHASE I, line 4-10 in Algo. 2.

Example 8.4.5 The bisimulation quotient of the PTA in Fig. 8.1 is shown in Fig. 8.5. There are four equivalence classes, with the above three labeled with

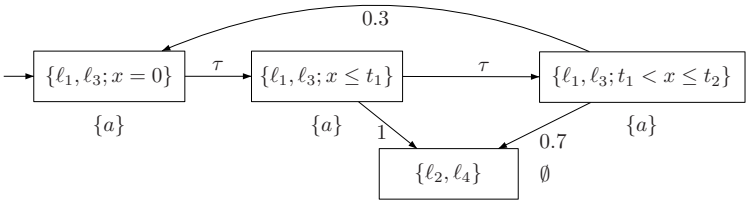


Figure 8.5: The bisimulation quotient

$\{a\}$ and the one below, the sinking state, with $\emptyset$. The label τ denotes that some time passes during the transition. The intuition is that from ℓ_1 or ℓ_3 it is possible to go to a state within a given period of time (the first τ), where either it takes a discrete transition to the sinking state, or it stays (taking the second τ) for some time till it can take a transition back resetting its clock with probability 0.3, or it goes to the sinking state with probability 0.7. t_1 and t_2 are arbitrary time points, which have been abstracted from the original model.

Correctness and termination. It is not difficult to see that by any of the *Refine* operators, we may obtain some new blocks, where for any two states in different blocks, they are not bisimilar and these blocks are disjoint. The correctness of the algorithm follows from standard correctness arguments of the standard partition-refinement scheme [PT87, KS90] and its probabilistic adaption [BEM00]. The definition of $\mathit{Refine}_t(\Pi, C)$ clearly yields a finer partition, which deals with timed transitions. For the two operators for discrete transitions, the first one, $\mathit{Refine}^1_d(\Pi, \nabla)$ (where ∇ is an action set), distinguishes the states with discrete transitions and it also yields a finer partition. The second one, $\mathit{Refine}^1_d(\Pi, M)$ (where M is an equivalence class with respect to the probabilistic distributions), is standard in computing probabilistic bisimulations (see [BEM00, Lem. 4.4]). The correctness of novel *Expand* operator follows directly from its definition.

Termination is ensured by Thm. 8.3.4. Namely, in the worst case, the algorithm will generate the partition induced by the region equivalence.

Complexity. We analyze the complexity of the algorithm briefly. Since in the worst case, the region equivalence will be obtained, our algorithm needs to refine exponentially many times to reach the fixpoint, and thus it is an EXPTIME algorithm. On the other hand, it is not very hard to show, by adapting the constructions of [CY92, LS00, LS07], that for PTA with at least three clocks, the EXPTIME lower-bound can be obtained. The key observation is that, as shown in [CY92, LS00], given a TA, the reachability problem is PSPACE-complete when the constants occurring in the guards of the TA belong to $\{0, 1\}$ or when the number of clocks is at least three (with relatively large constants to which these clocks are compared). Both of them turn out to be the source of the complexity

and conceptually they can be traded for each other. When the probability is involved, as shown in [LS07], the complexity lower-bound (of almost-sure probabilistic reachability, which is a natural variant of reachability in the probabilistic setting) climbs up to EXPTIME-complete (note that one can trade the number of clocks for the number of components, according to [LS00]). Moreover, one can easily use PTAB to "encode" the probabilistic reachability property (see also [JLS08] for relevant work). Hence the claimed EXPTIME lower-bound. Nevertheless, we note that (1) for PTA with only one clock, we can show that the algorithm only needs polynomial time to reach the fixpoint. (This is due to the fact that for one-clock TA, one can take larger granularity on the occurring constants, and thus obtain a region graph which is only of polynomial size, see [LMS04, JLS08].) Thus in this case, we can get a polynomial-time algorithm; (2) In practice, usually, PTA have a much coarser partition than the one induced by the region equivalence, and thus our algorithm is expected to perform pretty well in this case. Undoubtedly, (2) has to be confirmed by the experimental data, which is left as the future work. (The current chapter is only devoted to the theoretical development.) We only note here that the results of [TY01] for (non-probabilistic) TA are encouraging.

8.5 Verification of Branching-time Properties

The logic PCTL. In this section we prove that PTAB preserves branching-time properties specified in *probabilistic CTL* (PCTL, [HJ94]). We first briefly introduce the syntax and semantics of PCTL.

$$\Phi ::= \mathrm{tt} \mid a \mid \neg\Phi \mid \Phi \wedge \Phi \mid \exists\mathbb{P}_{\trianglelefteq p}(\phi) \ ,$$

where $p \in [0,1]$ is a probability, $a \in \mathrm{AP}$, $\trianglelefteq \in \{<, \leq, >, \geq\}$ and ϕ is a path formula defined as:

$$\phi ::= \Phi \,\mathsf{U}\, \Phi \mid \Phi \,\mathsf{W}\, \Phi \ .$$

Semantics. The semantics of PCTL is defined by a satisfaction relation, denoted $\models$, which is characterized as the least relation over the state in a PTS $\mathcal{M} = (S, \mathit{Steps}, L, s_0)$ (infinite paths in $\mathcal{M}$, respectively) and the state formulae (path formulae) satisfying:

$s \models \mathrm{tt}$		
$s \models a$	iff	$a \in L(s)$
$s \models \neg\Phi$	iff	not $(s \models \Phi)$
$s \models \Phi \wedge \Psi$	iff	$s \models \Phi$ and $s \models \Psi$
$s \models \exists\mathbb{P}\trianglelefteq p(\phi)$	iff	for some scheduler $\mathfrak{S} \in \mathfrak{W}$, $\mathit{Prob}(s,\phi) \trianglelefteq p$ in the DTMC $\mathcal{D}^{\mathfrak{S}}$, where $\mathit{Prob}(s,\phi) = \Pr\{\sigma \in \mathit{Paths}(s) \mid \sigma \models \phi\}$.

$$\begin{array}{rll} \rho \models \Phi \,\mathsf{U}\, \Psi & \text{iff} & \exists\, i.\big(\rho[i] \models \Psi \wedge \forall\, 0 \leqslant j < i.\, \rho[j] \models \Phi\big), \\ \rho \models \Phi \,\mathsf{W}\, \Psi & \text{iff} & \text{either } \rho \models \Phi \,\mathsf{U}^{\leqslant \Psi} \text{ or } \forall i.\rho[i] \models \Phi \ . \end{array}$$

The bisimulation relation can be lifted to paths in the following way:

Lemma 8.5.1 (Bisimulation on paths) Let $s \sim s'$. Then for each (finite or infinite) path $\omega = s_0 \xrightarrow{t_0,\mu_0} s_1 \xrightarrow{t_1,\mu_1} s_2 \cdots \in Paths(s)$, there exists a path $\omega' = s'_0 \xrightarrow{t'_0,\mu'_0} s_1 \xrightarrow{t'_1,\mu'_1} s'_2 \cdots \in Paths(s')$ of the same length such that for each $i \geq 0$, $s_i \sim s'_i$ and $\mu_i \equiv_\sim \mu'_i$.

Theorem 8.5.2 Let $\mathcal{G}$ be a PTA and $\sim$ be a PTAB on $\mathcal{G}$. For any PCTL formula Φ, $s \sim s'$ implies that $s \models \Phi$ iff $s' \models \Phi$.

Proof: The proof is by induction on the structure of Φ. The base case is trivial, since if $s \sim s'$, then $L(s) = L(s')$. The interesting inductive cases are for $\Phi = \mathbb{P}_{\trianglelefteq p}(\phi)$, where $\phi = \Psi_1 \,\mathsf{U}\, \Psi_2$. Assume that $s \models \Phi$. Then there exists a scheduler $\mathfrak{G} : Paths^* \to \mathbb{R} \times Distr(S)$ such that $Paths^{\mathfrak{G}}(s, \Psi_1 \mathsf{U} \Psi_2) = \{\omega \in Paths^{\mathfrak{G}}(s) \mid \exists i \geq 0.\ \omega(i,t_i) \models \Psi_2 \wedge \forall 0 \leq j < i,\ t < t_j.\ \omega(j,t) \models \Psi_1\}$ and $\Pr(Paths^{\mathfrak{G}}(s, \Psi_1 \mathsf{U} \Psi_2)) \trianglelefteq p$.

Assume $\omega \in Paths^{\mathfrak{G}}(s, \Psi_1 \mathsf{U} \Psi_2)$. According to Lem. 8.5.1, there must exist a probabilistic time-abstracting bisimilar path $\omega' \in Paths^{\mathfrak{G}'}(s', \Psi_1 \mathsf{U} \Psi_2)$, and vice versa. We can thus construct a scheduler $\mathfrak{G}' : Paths^* \to \mathbb{R} \times Distr(S)$ as follows: for $\omega \in Paths^*(s)$ and its bisimilar path $\omega' \in Paths^*(s')$, if $\mathfrak{G}(\omega) = (\mu, t)$, then $\mathfrak{G}'(\omega') = (\mu', t')$ and $\mu \equiv_\sim \mu'$.

It remains to show that $\Pr(Paths^{\mathfrak{G}}(s, \Psi_1 \mathsf{U} \Psi_2)) = \Pr(Paths^{\mathfrak{G}'}(s', \Psi_1 \mathsf{U} \Psi_2))$. Due to the fact that for each $\Psi_1 \mathsf{U} \Psi_2$ path, the probability distributions determined by $\mathfrak{G}$ and $\mathfrak{G}'$ are equivalent, the probability measures of the two sets of paths coincide. ■

The above theorem states that a PCTL formula is preserved by PTABs, which indicates that all the PCTL properties can be checked on the quotient PTSs. Thus all the existing techniques, algorithms, and tools for finite MDPs can be applied. We remark that the reverse direction of the theorem, however, does not hold in general.

8.6 Conclusion

We have investigated PTABs for PTA. This equivalence usually provides a much coarser partition than tradition region equivalence and preserves PCTL. We provided a non-trivial adaptation of the traditional partition-refinement algorithm to compute the quotient under PTAB.

As for future work, experimental evaluation of the efficiency of the proposed algorithm is to be carried out. Moreover, combined probabilistic time-abstracting bisimulations (where a transition can be simulated by a convex combination of transitions), *weak* probabilistic time-abstracting bisimulations, and logical characterizations of PTABare interesting topics which deserve further investigation. Furthermore, abstract-refinement and counterexample generation problems for PTA are in the plan.

Bibliography

[AB06] Rajeev Alur and Mikhail Bernadsky. Bounded model checking for GSMP models of stochastic real-time systems. In João P. Hespanha and Ashish Tiwari, editors, *HSCC*, volume 3927 of *Lecture Notes in Computer Science*, pages 19–33. Springer, 2006.

[ABKM09] Eric Allender, Peter Bürgisser, Johan Kjeldgaard-Pedersen, and Peter Bro Miltersen. On the complexity of numerical analysis. *SIAM J. Comput.*, 38(5):1987–2006, 2009.

[ACD91a] Rajeev Alur, Costas Courcoubetis, and David L. Dill. Model-checking for probabilistic real-time systems (extended abstract). In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez-Artalejo, editors, *ICALP*, volume 510 of *Lecture Notes in Computer Science*, pages 115–126. Springer, 1991.

[ACD91b] Rajeev Alur, Costas Courcoubetis, and David L. Dill. Verifying automata specifications of probabilistic real-time systems. In de Bakker et al. [dBHdRR92], pages 28–44.

[ACD93] Rajeev Alur, Costas Courcoubetis, and David L. Dill. Model-checking in dense real-time. *Inf. Comput.*, 104(1):2–34, 1993.

[Ace03] Luca Aceto. Some of my favourite results in classic process algebra. *Bulletin of the EATCS*, 81:90–108, 2003.

[ACFI06] Luca Aceto, Taolue Chen, Wan Fokkink, and Anna Ingólfsdóttir. On the axiomatizability of priority. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 2006.

[ACFI08] Luca Aceto, Taolue Chen, Wan Fokkink, and Anna Ingólfsdóttir. On the axiomatisability of priority. *Mathematical Structures in Computer Science*, 18(1):5–28, 2008.

[ACHH92] Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, and Pei-Hsin Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In Robert L.

Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, editors, *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*, pages 209–229. Springer, 1992.

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theor. Comput. Sci.*, 126(2):183–235, 1994.

[AD00] Robert B. Ash and Catherine A. Doléans-Dade. *Probability and Measure Theory*. Academic Press, 2000.

[AFH96] Rajeev Alur, Tomás Feder, and Thomas A. Henzinger. The benefits of relaxing punctuality. *J. ACM*, 43(1):116–146, 1996.

[AFI07] Luca Aceto, Wan Fokkink, and Anna Ingólfsdóttir. Ready to preorder: Get your BCCSP axiomatization for free! In Till Mossakowski, Ugo Montanari, and Magne Haveraaen, editors, *CALCO*, volume 4624 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2007.

[AFI08] Luca Aceto, Wan Fokkink, and Anna Ingólfsdóttir. A cancellation theorem for BCCSP. *Fundam. Inform.*, 88(1-2):1–21, 2008.

[AFIL05] Luca Aceto, Wan Fokkink, Anna Ingólfsdóttir, and Bas Luttik. CCS with Hennessy's merge has no finite-equational axiomatization. *Theor. Comput. Sci.*, 330(3):377–405, 2005.

[AFIL09] Luca Aceto, Wan Fokkink, Anna Ingólfsdóttir, and Bas Luttik. A finite equational base for CCS with left merge and communication merge. *ACM Trans. Comput. Log.*, 10(1), 2009.

[AFIM08] Luca Aceto, Wan Fokkink, Anna Ingólfsdóttir, and Mohammad Reza Mousavi. Lifting non-finite axiomatizability results to extensions of process algebras. In Giorgio Ausiello, Juhani Karhumäki, Giancarlo Mauri, and C.-H. Luke Ong, editors, *IFIP TCS*, volume 273 of *IFIP*, pages 301–316. Springer, 2008.

[AFIN06] Luca Aceto, Wan Fokkink, Anna Ingólfsdóttir, and Sumit Nain. Bisimilarity is not finitely based over BPA with interrupt. *Theor. Comput. Sci.*, 366(1-2):60–81, 2006.

[AFV01] Luca Aceto, Wan Fokkink, and Chris Verhoef. Structural operational semantics. In Jan Bergstra, Alban Ponse, and Scott Smolka, editors, *Handbook of Process Algebra*, pages 197–292. Elsevier Science, 2001.

[AFvGI04] Luca Aceto, Wan Fokkink, Rob J. van Glabbeek, and Anna Ingólfsdóttir. Nested semantics over finite trees are equationally hard. *Inf. Comput.*, 191(2):203–232, 2004.

[AH91] Rajeev Alur and Thomas A. Henzinger. Logics and models of real time: A survey. In de Bakker et al. [dBHdRR92], pages 74–106.

[AH93] Rajeev Alur and Thomas A. Henzinger. Real-time logics: Complexity and expressiveness. *Inf. Comput.*, 104(1):35–77, 1993.

[AH94] Rajeev Alur and Thomas A. Henzinger. A really temporal logic. *J. ACM*, 41(1):181–204, 1994.

[AH97] Rajeev Alur and Thomas A. Henzinger. Real-time system = discrete system + clock variables. *STTT*, 1(1-2):86–109, 1997.

[AI07] Luca Aceto and Anna Ingólfsdóttir. The saga of the axiomatization of parallel composition. In Caires and Vasconcelos [CV07], pages 2–16.

[AI08] Luca Aceto and Anna Ingólfsdóttir. On the expressibility of priority. *Inf. Process. Lett.*, 109(1):83–85, 2008.

[AILS07] Luca Aceto, Anna Ingólfsdóttir, Kim Larsen, and Jiri Srba. *Reactive Systems: Modelling, Specification and Verification*. Cambridge University Press, 2007.

[Alu99] Rajeev Alur. Timed automata. In Nicolas Halbwachs and Doron Peled, editors, *CAV*, volume 1633 of *Lecture Notes in Computer Science*, pages 8–22. Springer, 1999.

[AM04] Rajeev Alur and P. Madhusudan. Decision problems for timed automata: A survey. In Marco Bernardo and Flavio Corradini, editors, *SFM*, volume 3185 of *Lecture Notes in Computer Science*, pages 1–24. Springer, 2004.

[ASSB00] Adnan Aziz, Kumud Sanwal, Vigyan Singhal, and Robert K. Brayton. Model-checking continous-time Markov chains. *ACM Trans. Comput. Log.*, 1(1):162–170, 2000.

[AW95] George B. Arfken and Hans J. Weber. *Mathematical Methods for Physicists (4th ed.)*. Academic Press, 1995.

[BA07] Mikhail Bernadsky and Rajeev Alur. Symbolic analysis for GSMP models with one stateful clock. In Bemporad et al. [BBB07b], pages 90–103.

[Bae90] Jos C. M. Baeten, editor. *Applications of Process Algebra. Cambridge Tracts in Theoretical Computer Science (No. 17)*. Cambridge University Press, 1990.

[Bae05] Jos C. M. Baeten. A brief history of process algebra. *Theor. Comput. Sci.*, 335(2-3):131–146, 2005.

[BB00] Jos C. M. Baeten and Jan A. Bergstra. Mode transfer in process algebra. Technical report, CSR00–01, Eindhoven University of Technology, 2000.

[BBB+07a] Christel Baier, Nathalie Bertrand, Patricia Bouyer, Thomas Brihaye, and Marcus Größer. Probabilistic and topological semantics for timed automata. In Vikraman Arvind and Sanjiva Prasad, editors, *FSTTCS*, volume 4855 of *Lecture Notes in Computer Science*, pages 179–191. Springer, 2007.

[BBB07b] Alberto Bemporad, Antonio Bicchi, and Giorgio C. Buttazzo, editors. *Hybrid Systems: Computation and Control, 10th International Workshop, HSCC 2007, Pisa, Italy, April 3-5, 2007, Proceedings*, volume 4416 of *Lecture Notes in Computer Science*. Springer, 2007.

[BBB+08] Christel Baier, Nathalie Bertrand, Patricia Bouyer, Thomas Brihaye, and Marcus Größer. Almost-sure model checking of infinite paths in one-clock timed automata. In *LICS*, pages 217–226. IEEE Computer Society, 2008.

[BBBM08] Nathalie Bertrand, Patricia Bouyer, Thomas Brihaye, and Nicolas Markey. Quantitative model-checking of one-clock timed automata under probabilistic semantics. In *QEST*, pages 55–64. IEEE Computer Society, 2008.

[BBK86] Jos C. M. Baeten, Jan A. Bergstra, and Jan Willem Klop. Syntax and defining equations for an interrupt mechanism in process algebra. *Fundam. Inform.*, IX(2):127–168, 1986.

[BC05] Patricia Bouyer and Fabrice Chevalier. On conciseness of extensions of timed automata. *Journal of Automata, Languages and Combinatorics*, 10(4):393–405, 2005.

[BCH+07] Christel Baier, Lucia Cloth, Boudewijn R. Haverkort, Matthias Kuntz, and Markus Siegle. Model checking Markov chains with actions and state labels. *IEEE Trans. Software Eng.*, 33(4):209–224, 2007.

[BCJ09] Jasper Berendsen, Taolue Chen, and David N. Jansen. Undecidability of cost-bounded reachability in priced probabilistic timed automata. In Jianer Chen and S. Barry Cooper, editors, *TAMC*, volume 5532 of *Lecture Notes in Computer Science*, pages 128–137. Springer, 2009.

[BCSS98] Lenore Blum, Felipe Cucker, Michael Shub, and Steve Smale. *Compexity and Real Computation*. Springer-Verlag, 1998.

[BdA95] Andrea Bianco and Luca de Alfaro. Model checking of probabalistic and nondeterministic systems. In Thiagarajan [Thi95], pages 499–513.

[Bea03] Danièle Beauquier. On probabilistic timed automata. *Theor. Comput. Sci.*, 292(1):65–84, 2003.

[BEM00] Christel Baier, Bettina Engelen, and Mila E. Majster-Cederbaum. Deciding bisimilarity and similarity for probabilistic processes. *J. Comput. Syst. Sci.*, 60(1):187–231, 2000.

[Ber85] Jan A. Bergstra. Put and get, primitives for synchronous unreliable message passing. Technical report, Logic Group Preprint Series 3, Utrecht University, Department of Philosophy, 1985.

[BFN03] Stefan Blom, Wan Fokkink, and Sumit Nain. On the axiomatizability of ready traces, ready simulation, and failure traces. In Jos C. M. Baeten, Jan Karel Lenstra, Joachim Parrow, and Gerhard J. Woeginger, editors, *ICALP*, volume 2719 of *Lecture Notes in Computer Science*, pages 109–118. Springer, 2003.

[BHH+04] Christel Baier, Boudewijn R. Haverkort, Holger Hermanns, Joost-Pieter Katoen, and Markus Siegle, editors. *Validation of Stochastic Systems - A Guide to Current Research*, volume 2925 of *Lecture Notes in Computer Science*. Springer, 2004.

[BHHK03] Christel Baier, Boudewijn R. Haverkort, Holger Hermanns, and Joost-Pieter Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Trans. Software Eng.*, 29(6):524–541, 2003.

[BHR84] Stephen D. Brookes, C. A. R. Hoare, and A. W. Roscoe. A theory of communicating sequential processes. *J. ACM*, 31(3):560–599, 1984.

[Bil95] Patrick Billingsley. *Probability and Measure*. Wiley-Interscience, New York, 1995.

[BIM95] Bard Bloom, Sorin Istrail, and Albert R. Meyer. Bisimulation can't be traced. *J. ACM*, 42(1):232–268, 1995.

[BJK06] Jasper Berendsen, David N. Jansen, and Joost-Pieter Katoen. Probably on time and within budget: On reachability in priced probabilistic timed automata. In *QEST*, pages 311–322. IEEE Computer Society, 2006.

[BK82] Jan A. Bergstra and Jan Willem Klop. Fixed point semantics in process algebras. Report IW 206, Mathematical Centre, Amsterdam, 1982.

[BK84] Jan A. Bergstra and Jan Willem Klop. Process algebra for synchronous communication. *Information and Control*, 60(1-3):109–137, 1984.

[BK08] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. MIT Press, 2008.

[BL08] Patricia Bouyer and François Laroussinie. Model checking timed automata. In Stephan Merz and Nicolas Navet, editors, *Modeling and Verification of Real-Time Systems*, pages 111–140. ISTE Ltd. – John Wiley & Sons, Ltd., 2008.

[BLY96] Ahmed Bouajjani, Yassine Lakhnech, and Sergio Yovine. Model-checking for extended timed temporal logics. In Bengt Jonsson and Joachim Parrow, editors, *FTRTFT*, volume 1135 of *Lecture Notes in Computer Science*, pages 306–326. Springer, 1996.

[Bou04] Patricia Bouyer. Forward analysis of updatable timed automata. *Formal Methods in System Design*, 24(3):281–320, 2004.

[Bou09] Patricia Bouyer. Model-checking timed temporal logics. In Carlos Areces and Stéphane Demri, editors, *Proceedings of the 4th Workshop on Methods for Modalities (M4M-5)*, Electronic Notes in Theoretical Computer Science, Cachan, France, 2009. Elsevier Science Publishers. To appear.

[BPDG98] Béatrice Bérard, Antoine Petit, Volker Diekert, and Paul Gastin. Characterization of the expressive power of silent transitions in timed automata. *Fundam. Inform.*, 36(2-3):145–182, 1998.

[BPS01] Jan A. Bergstra, Alban Ponse, and Scott Smolka, editors. *Handbook of Process Algebra*. North-Holland, 2001.

[Bri85] Ed Brinksma. A tutorial on LOTOS. In Michel Diaz, editor, *PSTV*, pages 171–194. North-Holland, 1985.

[BS81] Stanley Burris and H. P. Sankappanavar. *A Course in Universal Algebra*. Springer-Verlag, New York, 1981.

[BSS89] Lenore Blum, Michael Shub, and Steve Smale. On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. *Bull. Am. Math. Soc.*, 21(1):1–46, 1989.

[BV95] Jos C.M. Baeten and Chris Verhoef. Concrete process algebra. In S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of logic in computer science (vol. 4): semantic modelling*, pages 149–268. Oxford University Press, 1995.

[BW90] Jos C. M. Baeten and W.P. Weijland. *Process Algebra, Cambridge Tracts in Theoretical Computer Science 18*. Cambridge University Press, 1990.

[BY03] Johan Bengtsson and Wang Yi. Timed automata: Semantics, algorithms and tools. In Jörg Desel, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Lectures on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*, pages 87–124. Springer, 2003.

[CD88] Oswaldo L.V. Costa and Mark H. A. Davis. Approximations for optimal stopping of a piecewise-deterministic process. *Math. Control Signals Systems*, 1(2):123–146, 1988.

[CES86] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Program. Lang. Syst.*, 8(2):244–263, 1986.

[CF06] Taolue Chen and Wan Fokkink. On finite alphabets and infinite bases III: Simulation. In Christel Baier and Holger Hermanns, editors, *CONCUR*, volume 4137 of *Lecture Notes in Computer Science*, pages 421–434. Springer, 2006.

[CF08] Taolue Chen and Wan Fokkink. On the axiomatizability of impossible futures: Preorder versus equivalence. In *LICS*, pages 156–165. IEEE Computer Society, 2008.

[CFLN08] Taolue Chen, Wan Fokkink, Bas Luttik, and Sumit Nain. On finite alphabets and infinite bases. *Inf. Comput.*, 206(5):492–519, 2008.

[CFN06] Taolue Chen, Wan Fokkink, and Sumit Nain. On finite alphabets and infinite bases II: Completed and ready simulation. In Luca Aceto and Anna Ingólfsdóttir, editors, *FoSSaCS*, volume 3921 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2006.

[CFvG08] Taolue Chen, Wan Fokkink, and Rob J. van Glabbeek. Ready to preorder: The case of weak process semantics. *Inf. Process. Lett.*, 109(2):104–111, 2008.

[CFvG09] Taolue Chen, Wan Fokkink, and Rob J. van Glabbeek. On finite bases for weak semantics: Failures versus impossible futures. In Mogens Nielsen, Antonín Kucera, Peter Bro Miltersen, Catuscia Palamidessi, Petr Tuma, and Frank D. Valencia, editors, *SOFSEM*, volume 5404 of *Lecture Notes in Computer Science*, pages 167–180. Springer, 2009.

[CGP99] Edmund M. Clarke, Orna Grumberg, and Doron Peled. *Model Checking*. MIT Press, 1999.

[CH90] Rance Cleaveland and Matthew Hennessy. Priorities in process algebras. *Inf. Comput.*, 87(1/2):58–77, 1990.

[CHK08] Taolue Chen, Tingting Han, and Joost-Pieter Katoen. Time-abstracting bisimulation for probabilistic timed automata. In *TASE*, pages 177–184. IEEE Computer Society, 2008.

[CHKM09a] Taolue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre. LTL model checking of time-inhomogeneous Markov chains. In *ATVA*. Springer, 2009. To appear in *Lecture Notes in Computer Science*.

[CHKM09b] Taolue Chen, Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre. Quantitative model checking of continuous-time Markov chains against timed automata specifications. In *LICS*. IEEE Computer Society, 2009. To appear. Also aviailable as technical report AIB-2009-02, RWTH Aachen University, Germany.

[CLN01] Rance Cleaveland, Gerald Luttgen, and V. Natarajan. Priorities in process algebra. In Jan Bergstra, Alban Ponse, and Scott Smolka, editors, *Handbook of Process Algebra*, pages 711–765. Elsevier Science, 2001.

[CLN07] Rance Cleaveland, Gerald Lüttgen, and V. Natarajan. Priority and abstraction in process algebra. *Inf. Comput.*, 205(9):1426–1458, 2007.

[CLNS96] Rance Cleaveland, Gerald Lüttgen, V. Natarajan, and Steve Sims. Priorities for modeling and verifying distributed systems. In Tiziana Margaria and Bernhard Steffen, editors, *TACAS*, volume 1055 of *Lecture Notes in Computer Science*, pages 278–297. Springer, 1996.

[Cor91] C. Corduneanu. *Integral Equations and Applications*. Cambridge University Press, 1991.

[CPvdPW07] Taolue Chen, Bas Ploeger, Jaco van de Pol, and Tim A. C. Willemse. Equivalence checking for infinite systems using parameterized boolean equation systems. In Caires and Vasconcelos [CV07], pages 120–135.

[CSKN05] Stefano Cattani, Roberto Segala, Marta Z. Kwiatkowska, and Gethin Norman. Stochastic transition systems for continuous state spaces and non-determinism. In Vladimiro Sassone, editor, *FoSSaCS*, volume 3441 of *Lecture Notes in Computer Science*, pages 125–139. Springer, 2005.

[CV07] Luís Caires and Vasco Thudichum Vasconcelos, editors. *CONCUR 2007 - Concurrency Theory, 18th International Conference, CONCUR 2007, Lisbon, Portugal, September 3-8, 2007,*

Proceedings, volume 4703 of *Lecture Notes in Computer Science*. Springer, 2007.

[CvdPW08] Taolue Chen, Jaco van de Pol, and Yanjing Wang. PDL over accelerated labeled transition systems. In *TASE*, pages 193–200. IEEE Computer Society, 2008.

[CW95] Juanito Camilleri and Glynn Winskel. CCS with priority choice. *Inf. Comput.*, 116(1):26–37, 1995.

[CY92] Costas Courcoubetis and Mihalis Yannakakis. Minimum and maximum delay problems in real-time systems. *Formal Methods in System Design*, 1(4):385–415, 1992.

[CY95] Costas Courcoubetis and Mihalis Yannakakis. The complexity of probabilistic verification. *J. ACM*, 42(4):857–907, 1995.

[Dav84] Mark H. A. Davis. Piecewise-deterministic Markov processes: A general class of non-diffusion stochastic models. *Journal of the Royal Statistical Society (B)*, 46(3):353–388, 1984.

[Dav93] Mark H. A. Davis. *Markov Models and Optimization*. Chapman and Hall, 1993.

[DB95] Ashvin Dsouza and Bard Bloom. On the expressive power of CCS. In Thiagarajan [Thi95], pages 309–323.

[dBHdRR92] J. W. de Bakker, Cornelis Huizing, Willem P. de Roever, and Grzegorz Rozenberg, editors. *Real-Time: Theory in Practice, REX Workshop, Mook, The Netherlands, June 3-7, 1991, Proceedings*, volume 600 of *Lecture Notes in Computer Science*. Springer, 1992.

[DEP02] Josee Desharnais, Abbas Edalat, and Prakash Panangaden. Bisimulation for labelled markov processes. *Inf. Comput.*, 179(2):163–193, 2002.

[dFG07] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. Simulations up-to and canonical preorders: (extended abstract). *Electr. Notes Theor. Comput. Sci.*, 192(1):13–28, 2007.

[dFG09] David de Frutos-Escrig and Carlos Gregorio-Rodríguez. (Bi)simulations up-to characterise process semantics. *Inf. Comput.*, 207(2):146–170, 2009.

[dFGP08a] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. Coinductive characterisations reveal nice relations between preorders and equivalences. *Electr. Notes Theor. Comput. Sci.*, 212:149–162, 2008.

[dFGP08b] David de Frutos-Escrig, Carlos Gregorio-Rodríguez, and Miguel Palomino. Ready to preorder: an algebraic and general proof. *J. Log. Algebr. Program.*, 2008.

[DHS03] Salem Derisavi, Holger Hermanns, and William H. Sanders. Optimal state-space lumping in Markov chains. *Inf. Process. Lett.*, 87(6):309–315, 2003.

[DHS09] Susanna Donatelli, Serge Haddad, and Jeremy Sproston. Model checking timed and stochastic properties with CSL^{TA}. *IEEE Trans. Software Eng.*, 35(2):224–240, 2009.

[DW02] Klaus Denecke and Shelly L. Wismath. *Universal Algebra and Applications in Theoretical Computer Science*. CRC/C&H, 2002.

[FL00] Wan Fokkink and Bas Luttik. An ω-complete equational specification of interleaving. In Ugo Montanari, José D. P. Rolim, and Emo Welzl, editors, *ICALP*, volume 1853 of *Lecture Notes in Computer Science*, pages 729–743. Springer, 2000.

[FN04] Wan Fokkink and Sumit Nain. On finite alphabets and infinite bases: From ready pairs to possible worlds. In Igor Walukiewicz, editor, *FoSSaCS*, volume 2987 of *Lecture Notes in Computer Science*, pages 182–194. Springer, 2004.

[FN05] Wan Fokkink and Sumit Nain. A finite basis for failure semantics. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 755–765. Springer, 2005.

[Fok00] Wan Fokkink. *Introduction to Process Algebra. EATCS Texts in Theoretical Computer Science*. Springer, 2000.

[Fok07] Wan Fokkink. *Modelling Distributed Systems. EATCS Texts in Theoretical Computer Science*. Springer, 2007.

[GR01] Jan Friso Groote and Michel A. Reniers. Glgebraic process verification. In Jan Bergstra, Alban Ponse, and Scott Smolka, editors, *Handbook of Process Algebra*, pages 1151–1208. Elsevier Science, 2001.

[Gro90] Jan Friso Groote. A new strategy for proving ω-completeness applied to process algebra. In Jos C. M. Baeten and Jan Willem Klop, editors, *CONCUR*, volume 458 of *Lecture Notes in Computer Science*, pages 314–331. Springer, 1990.

[Gur90] R. Gurevic. Equational theory of positive numbers with exponentiation is not finitely axiomatizable. *Ann. Pure Appl. Logic*, 49(1):1–30, 1990.

[GV08] Orna Grumberg and Helmut Veith, editors. *25 Years of Model Checking - History, Achievements, Perspectives*, volume 5000 of *Lecture Notes in Computer Science*. Springer, 2008.

[Hav00] Boudewijn R. Haverkort. Markovian models for performance and dependability evaluation. In Ed Brinksma, Holger Hermanns, and Joost-Pieter Katoen, editors, *European Educational Forum: School on Formal Methods and Performance Analysis*, volume 2090 of *Lecture Notes in Computer Science*, pages 38–83. Springer, 2000.

[Hee86] Jan Heering. Partial evaluation and ω-completeness of algebraic specifications. *Theor. Comput. Sci.*, 43:149–167, 1986.

[Hen77] L. Henkin. The logic of equality. *American Mathematical Monthly*, 84(8):597–612, 1977.

[Hen88] Matthew Hennessy. *Algebraic Theory of Processes*. MIT Press, Cambridge, MA, 1988.

[Hen96] Thomas A. Henzinger. The theory of hybrid automata. In *LICS*, pages 278–292, 1996.

[Hen98] Thomas A. Henzinger. It's about time: Real-time logics reviewed. In Davide Sangiorgi and Robert de Simone, editors, *CONCUR*, volume 1466 of *Lecture Notes in Computer Science*, pages 439–454. Springer, 1998.

[HJ94] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Asp. Comput.*, 6(5):512–535, 1994.

[HKNP06] Andrew Hinton, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. PRISM: A tool for automatic verification of probabilistic systems. In Holger Hermanns and Jens Palsberg, editors, *TACAS*, volume 3920 of *Lecture Notes in Computer Science*, pages 441–444. Springer, 2006.

[HKPV98] Thomas A. Henzinger, Peter W. Kopke, Anuj Puri, and Pravin Varaiya. What's decidable about hybrid automata? *J. Comput. Syst. Sci.*, 57(1):94–124, 1998.

[HM85] Matthew Hennessy and Robin Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985.

[HNSY94] Thomas A. Henzinger, Xavier Nicollin, Joseph Sifakis, and Sergio Yovine. Symbolic model checking for real-time systems. *Inf. Comput.*, 111(2):193–244, 1994.

[Hoa85] C.A.R Hoare. *Communicating Sequential Process*. Prentice Hall, 1985.

[ISO87] ISO. *Information processing systems – open systems interconnection – LOTOS – a formal description technique based on the temporal ordering of observational behaviour* ISO/TC97/SC21/N DIS8807, 1987.

[JLS08] Marcin Jurdzinski, François Laroussinie, and Jeremy Sproston. Model checking probabilistic timed automata with one or two clocks. *CoRR*, abs/0809.0060, 2008.

[Kat08a] Joost-Pieter Katoen. Perspectives in probabilistic verification. In *TASE*, pages 3–10. IEEE Computer Society, 2008.

[Kat08b] Joost-Pieter Katoen. Quantitative evaluation in embedded system design: Trends in modeling and analysis techniques. In *DATE*, pages 86–87. IEEE, 2008.

[Kel76] Robert M. Keller. Formal verification of parallel programs. *Commun. ACM*, 19(7):371–384, 1976.

[KKZ05] Joost-Pieter Katoen, Maneesh Khattri, and Ivan S. Zapreev. A Markov reward model checker. In *QEST*, pages 243–244. IEEE Computer Society, 2005.

[KNP04] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Probabilistic symbolic model checking with PRISM: a hybrid approach. *STTT*, 6(2):128–142, 2004.

[KNPS08] Marta Z. Kwiatkowska, Gethin Norman, Dave Parker, and Jeremy Sproston. *Modeling and Verification of Real-Time Systems: Formalisms and Software Tools*, chapter Verification of Real-Time Probabilistic Systems, pages 249–288. John Wiley & Sons, 2008.

[KNS03] Marta Z. Kwiatkowska, Gethin Norman, and Jeremy Sproston. Probabilistic model checking of deadline properties in the IEEE 1394 Firewire root contention protocol. *Formal Asp. Comput.*, 14(3):295–318, 2003.

[KNSS00] Marta Z. Kwiatkowska, Gethin Norman, Roberto Segala, and Jeremy Sproston. Verifying quantitative properties of continuous probabilistic timed automata. In Catuscia Palamidessi, editor, *CONCUR*, volume 1877 of *Lecture Notes in Computer Science*, pages 123–137. Springer, 2000.

[KNSS02] Marta Z. Kwiatkowska, Gethin Norman, Roberto Segala, and Jeremy Sproston. Automatic verification of real-time systems with discrete probability distributions. *Theor. Comput. Sci.*, 282(1):101–150, 2002.

[KNSW07] Marta Z. Kwiatkowska, Gethin Norman, Jeremy Sproston, and Fuzhi Wang. Symbolic model checking for probabilistic timed automata. *Inf. Comput.*, 205(7):1027–1077, 2007.

[Koy90] Ron Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.

[KS90] Paris C. Kanellakis and Scott A. Smolka. CCS expressions, finite state processes, and three problems of equivalence. *Journal of Information and Computation*, 86(1):43–68, 1990.

[KSK76] John G. Kemeny, J. Laurie Snell, and Anthony W. Knapp. *Denumerable Markov Chains (Graduate Texts in Mathematics)*. Springer, 1976.

[Lin95] Huimin Lin. PAM: A process algebra manipulator. *Formal Methods in System Design*, 7(3):243–259, 1995.

[LL85] Suzanne M. Lenhart and Yu-Chung Liao. Integro-differential equations associated with optimal stopping time of a piecewise-deterministic process. *Stochastics*, 15(3):183–207, 1985.

[LLT90] Azeddine Lazrek, Pierre Lescanne, and Jean-Jacques Thiel. Tools for proving inductive equalities, relative completeness, and ω-completeness. *Inf. Comput.*, 84(1):47–70, 1990.

[LLW95] François Laroussinie, Kim Guldstrand Larsen, and Carsten Weise. From timed automata to logic - and back. In Jirí Wiedermann and Petr Hájek, editors, *MFCS*, volume 969 of *Lecture Notes in Computer Science*, pages 529–539. Springer, 1995.

[LMS04] François Laroussinie, Nicolas Markey, and Ph. Schnoebelen. Model checking timed automata with one or two clocks. In *CONCUR*, pages 387–401, 2004.

[LMST03] Ruggero Lanotte, Andrea Maggiolo-Schettini, and Angelo Troina. Weak bisimulation for probabilistic timed automata and applications to security. In *SEFM*, pages 34–43. IEEE Computer Society, 2003.

[LS91] Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Inf. Comput.*, 94(1):1–28, 1991.

[LS00] Francesca Levi and Davide Sangiorgi. Controlling interference in ambients. In *POPL*, pages 352–364, 2000.

[LS07] François Laroussinie and Jeremy Sproston. State explosion in almost-sure probabilistic reachability. *Inf. Process. Lett.*, 102(6):236–241, 2007.

[Lut06] Bas Luttik. What is algebraic in process theory. *Bulletin of the EATCS*, 88:66–83, 2006.

[LY91] Suzanne M. Lenhart and Naoki Yamada. Perron's method for viscosity solutions associated with piecewise-deterministic processes. *Funkcialaj Ekvacioj*, 34:173–186, 1991.

[LY97] Kim Guldstrand Larsen and Wang Yi. Time-abstracted bisimulation: Implicit specifications and decidability. *Inf. Comput.*, 134(2):75–101, 1997.

[Lyn51] R. Lyndon. Identities in two-valued calculi. *Transactions of the American Mathematical Society*, 71:457–465, 1951.

[Mau91] Sjouke Mauw. *PSF – A Process Specification Formalism*. PhD thesis, University of Amsterdam, December 1991.

[McK96] R. McKenzie. Tarski's finite basis problem is undecidable. *Journal of Algebra and Computation*, 6(1):49–104, 1996.

[Mil80] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980.

[Mil84] Robin Milner. A complete inference system for a class of regular behaviours. *J. Comput. Syst. Sci.*, 28(3):439–466, 1984.

[Mil89a] Robin Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[Mil89b] Robin Milner. A complete axiomatisation for observational congruence of finite-state behaviors. *Inf. Comput.*, 81(2):227–247, 1989.

[Mil90] Robin Milner. Operational and algebraic semantics of concurrent processes. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 1201–1242. 1990.

[ML07] Sayan Mitra and Nancy A. Lynch. Trace-based semantics for probabilistic timed I/O automata. In Bemporad et al. [BBB07b], pages 718–722.

[MMT87] R. McKenzie, G. McNulty, and W. Taylor. *Algebras, Varieties, Lattices*. Wadsworth & Brooks/Cole, 1987.

[Mol89] F. Moller. *Axioms for Concurrency*. PhD thesis, Department of Computer Science, University of Edinburgh, July 1989. Report CST-59-89. Also published as ECS-LFCS-89-84.

[Mol90a] Faron Moller. The importance of the left merge operator in process algebras. In Mike Paterson, editor, *ICALP*, volume 443 of *Lecture Notes in Computer Science*, pages 752–764. Springer, 1990.

[Mol90b] Faron Moller. The nonexistence of finite axiomatisations for CCS congruences. In *LICS*, pages 142–153. IEEE Computer Society, 1990.

[MPW92a] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, I. *Inf. Comput.*, 100(1):1–40, 1992.

[MPW92b] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, II. *Inf. Comput.*, 100(1):41–77, 1992.

[MTHM97] Robin Milner, Mads Tofte, Robert Harper, and David MacQueen. *The Definition of Standard ML (Revised)*. MIT Press, 1997.

[Mur65] V.L. Murskiĭ. The existence in the three-valued logic of a closed class with a finite basis having no finite complete system of identities. *Doklady Akademii Nauk SSSR*, 163:815–818, 1965. In Russian.

[Mur75] V.L. Murskiĭ. The existence of a finite basis of identities, and other properties of "almost all" finite algebras. *Problemy Kibernetiki*, 30:43–56, 1975. In Russian.

[NH84] Rocco De Nicola and Matthew Hennessy. Testing equivalences for processes. *Theor. Comput. Sci.*, 34:83–133, 1984.

[OW08] Joël Ouaknine and James Worrell. Some recent results in metric temporal logic. In Franck Cassez and Claude Jard, editors, *FORMATS*, volume 5215 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2008.

[Pan01] Prakash Panangaden. Measure and probability for concurrency theorists. *Theor. Comput. Sci.*, 253(2):287–309, 2001.

[Par81] David Park. Concurrency and automata on infinite sequences. In Peter Deussen, editor, *Theoretical Computer Science*, volume 104 of *Lecture Notes in Computer Science*, pages 167–183. Springer, 1981.

[Phi87] Iain Phillips. Refusal testing. *Theor. Comput. Sci.*, 50:241–284, 1987.

[Phi08] Iain Phillips. CCS with priority guards. *J. Log. Algebr. Program.*, 75(1):139–165, 2008.

[Plo74] Gordon D. Plotkin. The lambda-calculus is ω-incomplete. *J. Symb. Log.*, 39(2):313–317, 1974.

[Plo04a] Gordon D. Plotkin. The origins of structural operational semantics. *J. Log. Algebr. Program.*, 60-61:3–15, 2004.

[Plo04b] Gordon D. Plotkin. A structural approach to operational semantics. *J. Log. Algebr. Program.*, 60-61:17–139, 2004.

[Pnu77] A. Pnueli. The temporal logic of programs. In *FOCS*, pages 46–57, Los Alamitos, CA, 1977. Computer Society Press.

[PT87] Robert Paige and Robert Endre Tarjan. Three partition refinement algorithms. *SIAM J. Comput.*, 16(6):973–989, 1987.

[Put94] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, New York, 1994.

[RB81] William C. Rounds and Stephen D. Brookes. Possible futures, acceptances, refusals, and communicating processes. In *FOCS*, pages 140–149. IEEE, 1981.

[RV07] Arend Rensink and Walter Vogler. Fair testing. *Inf. Comput.*, 205(2):125–198, 2007.

[Sew97] Peter Sewell. Nonaxiomatisability of equivalences over finite state processes. 90(1–3):163–191, 15 December 1997.

[Spr04] Jeremy Sproston. Model checking for probabilistic timed systems. In Baier et al. [BHH$^+$04], pages 189–229.

[Sto02] Mariëlle Stoelinga. An introduction to probabilistic automata. *Bulletin of the EATCS*, 78:176–198, 2002.

[Thi95] P. S. Thiagarajan, editor. *Foundations of Software Technology and Theoretical Computer Science, 15th Conference, Bangalore, India, December 18-20, 1995, Proceedings*, volume 1026 of *Lecture Notes in Computer Science*. Springer, 1995.

[Tho97] Wolfgang Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, volume III*, pages 389–455. Springer, New York, 1997.

[TY01] Stavros Tripakis and Sergio Yovine. Analysis of timed systems using time-abstracting bisimulations. *Formal Methods in System Design*, 18(1):25–68, 2001.

[Var85] Moshe Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *FOCS*, pages 327–338, 1985.

[vG90] Rob J. van Glabbeek. The linear time-branching time spectrum (extended abstract). In Jos C. M. Baeten and Jan Willem Klop, editors, *CONCUR*, volume 458 of *Lecture Notes in Computer Science*, pages 278–297. Springer, 1990.

[vG93a] Rob J. van Glabbeek. A complete axiomatization for branching bisimulation congruence of finite-state behaviours. In Andrzej M. Borzyszkowski and Stefan Sokolowski, editors, *MFCS*, volume 711 of *Lecture Notes in Computer Science*, pages 473–484. Springer, 1993.

[vG93b] Rob J. van Glabbeek. The linear time - branching time spectrum II. In Eike Best, editor, *CONCUR*, volume 715 of *Lecture Notes in Computer Science*, pages 66–81. Springer, 1993.

[vG97] Rob J. van Glabbeek. Notes on the methodology of CCS and CSP. *Theor. Comput. Sci.*, 177(2):329–349, 1997.

[vG01] Rob J. van Glabbeek. The linear time-branching time spectrum I. In Jan Bergstra, Alban Ponse, and Scott Smolka, editors, *Handbook of Process Algebra*, pages 3–99. Elsevier Science, 2001.

[vGV06] Rob J. van Glabbeek and Marc Voorhoeve. Liveness, fairness and impossible futures. In Christel Baier and Holger Hermanns, editors, *CONCUR*, volume 4137 of *Lecture Notes in Computer Science*, pages 126–141. Springer, 2006.

[vGW96] Rob J. van Glabbeek and W. P. Weijland. Branching time and abstraction in bisimulation semantics. *J. ACM*, 43(3):555–600, 1996.

[VM01] Marc Voorhoeve and Sjouke Mauw. Impossible futures and determinism. *Inf. Process. Lett.*, 80(1):51–58, 2001.

[Vog92] Walter Vogler. *Modular Construction and Partial Order Semantics of Petri Nets*, volume 625 of *Lecture Notes in Computer Science*. Springer, 1992.

Summary

This dissertation consists of two parts.

Part I concentrates on the axiomatizability of process algebras, mostly on two basic process algebras BCCSP and BCCS.

Chapter 3 presents two meta-theorems regarding the axiomatizability. The first one concerns the relationship between preorders and equivalences. We show that the same algorithm proposed by Aceto *el al.* and de Frutos Escrig *et al.* for concrete semantics, which transforms an axiomatization for a preorder to the one for the corresponding equivalence, applies equally well to weak semantics. This makes it applicable to all 87 preorders surveyed in the "linear time – branching time spectrum II" that are at least as coarse as the ready simulation preorder. We also extend the scope of the algorithm to infinite processes, by adding recursion constants. The second meta-theorem concerns the relationship between concrete and weak semantics. For any semantics which is not finer than failures or impossible futures semantics, we provide an algorithm to transform an axiomatization for the concrete version to the one for the weak counterpart. As an application of this algorithm, we derive ground- and ω-complete axiomatizations for weak failure, weak completed trace, weak trace preorders.

Chapter 4 settles the remaining open questions regarding the existence of ω-complete axiomatizations in the setting of the process algebra BCCSP for all the semantics in the linear time – branching time spectrum I, either positively by giving a finite, sound and ground-complete axiomatization which turns out to be ω-complete, or negatively by proving that such a finite basis of the equational theory does not exist. We prove that in case of a finite alphabet with at least two actions, failure semantics affords a finite basis, while for ready simulation, completed simulation, simulation, possible worlds, ready trace, failure trace and ready semantics, such a finite basis does not exist. Completed simulation semantics also lacks a finite basis in case of an infinite alphabet of actions.

Chapter 5 investigates the (in)equational theories of concrete and weak impossible futures semantics over the process algebras BCCSP and BCCS. We present a finite, sound, ground-complete axiomatization for BCCSP modulo the concrete impossible futures preorder, which implies a finite, sound, ground-complete axiomatization for BCCS modulo the weak impossible futures preorder. By contrast, we prove that no finite, sound axiomatization for BCCS modulo the weak impossible futures equivalence is ground-complete, and this

negative result carries over to the concrete case. If the alphabet of actions is infinite, then the aforementioned ground-complete axiomatizations are shown to be ω-complete. However, if the alphabet is finite and nonempty, we prove that the inequational (resp. equational) theories of BCCSP and BCCS modulo the impossible futures preorder (resp. equivalence) lack such a finite basis. Finally, we show that the negative result regarding impossible futures equivalence extends to all n-nested impossible futures equivalences for $n \geq 2$, and to all n-nested impossible futures preorders for $n \geq 3$.

Chapter 6 studies the equational theory of bisimulation equivalence over the process algebra BCCSP_Θ, i.e., BCCSP extended with the priority operator Θ of Baeten *et al.* It is proven that, in the presence of an infinite set of actions, bisimulation equivalence has no finite, sound, ground-complete axiomatization over that language. This negative result applies even if the syntax is extended with an arbitrary collection of auxiliary operators, and motivates the study of axiomatizations using equations with action predicates as conditions. In the presence of an infinite set of actions, it is shown that, in general, bisimulation equivalence has no finite, sound, ground-complete axiomatization consisting of equations with action predicates as conditions over BCCSP_Θ. Finally, sufficient conditions on the priority structure over actions are identified that lead to a finite, sound, ground-complete axiomatization of bisimulation equivalence using equations with action predicates as conditions.

Part II concentrates on the verification of probabilistic real-time systems.

Chapter 7 studies the following problem: given a continuous-time Markov chain $\mathcal{C}$, and a linear real-time property provided as a deterministic timed automaton $\mathcal{A}$, what is the probability of the set of paths of $\mathcal{C}$ that are accepted by $\mathcal{A}$? It is shown that this set of paths is measurable and computing its probability can be reduced to computing the reachability probability in a piecewise deterministic Markov process. The reachability probability is characterized as the least solution of a system of integral equations and is shown to be approximated by solving a system of partial differential equations. For the special case of single-clock deterministic timed automata, the system of integral equations can be transformed into a system of linear equations where the coefficients are solutions of ordinary differential equations.

Chapter 8 focuses on probabilistic timed automata, an extension of timed automata with discrete probabilistic branchings. As the region construction of these automata often leads to an exponential blow-up over the size of original automata, reduction techniques are of the utmost importance. In this chapter, we investigate probabilistic time-abstracting bisimulation (PTaB), an equivalence notion that abstracts from exact time delays. PTaB is proven to preserve probabilistic computation tree logic. The region equivalence is a (very refined) PTaB. Furthermore, we provide a non-trivial adaptation of the traditional partition-refinement algorithm to compute the quotient under the PTaB. This algorithm is symbolic in the sense that equivalence classes are represented as polyhedra.

Nederlandse Samenvatting[1]

Klokken, Dobbelen en Processen

Het proefschrift bestaat uit twee delen.

Deel I concentreert zich op de axiomatizeerbaarheid van procesalgebra's, met name op twee basale procesalgebra's BCCSP en BCCS.

Hoofdstuk 3 bevat twee meta-stellingen betreffende axiomatizeerbaarheid. De eerste richt zich op de relatie tussen preorders and equivalenties. We tonen aan dat het algoritme van Aceto *el al.* en de Frutos Escrig *et al.* voor concrete semantieken, die een axiomatizering voor een preorder omzet in een axiomatizering voor de corresponderende equivalentie, ook van toepassing is op zwakke semantieken. Dit maakt het van toepassing op alle 87 preorders in het "lineaire tijd – splitsende tijd spectrum II" die minstens zo grof zijn als de ready simulation preorder. We breiden de scope van het algoritme ook uit tot oneindige processen, door toevoeging van recursie-constanten. De tweede meta-stelling betreft de relatie tussen concrete and zwakke semantieken. Voor iedere semantiek die niet fijner is dan failures of impossible futures semantiek, geven wij een algoritme om een axiomatizering voor de concrete versie om te zetten naar een axiomatizering voor de zwakke tegenhanger. Als een toepassing van dit algoritme leiden we grond- en ω-complete axiomatizeringen af voor weak failure, weak completed trace, en weak trace preorder.

Hoofdstuk 4 beantwoordt alle openstaande vragen wat betreft het bestaan van ω-complete axiomatizeringen in de context van de procesalgebra BCCSP, voor de semantieken in het "lineaire tijd – splitsende tijd spectrum I". De antwoorden zijn ofwel positief door een eindige, grond-complete axiomatizering te geven die ook ω-compleet is, ofwel negatief door te bewijzen dat een dergelijke eindige basis voor de equationele theorie niet bestaat. We bewijzen dat er bij een eindig alfabet van tenminste twee acties, een eindige basis is voor failure semantiek, terwijl voor ready simulation, completed simulation, simulation, possible worlds, ready trace, failure trace and ready semantiek, zo'n eindige basis niet bestaat. Completed simulation semantiek heeft ook geen eindige basis bij een oneindig alfabet van acties.

Hoofdstuk 5 onderzoekt de equationele theorieën voor concrete en zwakke impossible futures semantiek over de procesalgebra's BCCSP and BCCS. We

[1]Dutch Summary. Translated from the English summary by Wan Fokkink.

presenteren een eindige, grond-complete axiomatizering voor BCCSP modulo de concrete impossible futures preorder, wat een eindige, grond-complete axiomatizering impliceert voor BCCS modulo de zwakke impossible futures preorder. In contrast hiermee bewijzen we dat er geen eindige, grond-complete axiomatizering bestaat voor BCCS modulo de zwakke impossible futures equivalentie; dit negatieve resultaat is overdraagbaar naar het concrete geval. Indien het alfabet van acties eindig is, tonen we aan dat de voornoemde grond-complete axiomatizeringen ω-compleet zijn. Echter, indien het alfabet eindig is en niet leeg, dan bewijzen we dat de inequationele (resp. equationele) theorieën van BCCSP and BCCS modulo impossible futures preorder (resp. equivalentie) zo'n eindige basis ontberen. Tenslotte tonen we aan dat het negatieve resultaat betreffende impossible futures equivalentie ook van toepassing is op alle n-nested impossible futures equivalenties voor $n \geq 2$, and op alle n-nested impossible futures preorders voor $n \geq 3$.

Hoofdstuk 6 bestudeert de equationele theorie van bisimulatie equivalentie voor de procesalgebra BCCSP_Θ, oftewel, BCCSP uitgebreid met de prioriteitsoperator Θ van Baeten *et al.* Er wordt bewezen dat in aanwezigheid van een oneindig alfabet, bisimulatie equivalentie geen eindige, grond-complete axiomatizering heeft over deze procesalgebra. Dit negatieve resultaat is zelfs van toepassing als de syntax wordt uitgebreid met een willekeurige collectie hulpoperatoren, en motiveert de studie van axiomatizeringen die gebruik maken van vergelijkingen met actie-predicaten als condities. In aanwezigheid van een oneindig alfabet, wordt aangetoond dat bisimulatie equivalentie in het algemeen geen eindige, grond-complete axiomatizering heeft over BCCSP_Θ van vergelijkingen met actie-predicaten als condities. Tenslotte worden condities gegeven op de prioriteits-structuur over acties, die aanleiding geven tot een eindige, grond-complete axiomatizering van bisimulatie equivalentie, waarbij de vergelijkingen zijn voorzien van actie-predicaten als condities.

Deel II concentreert zich op de verificatie van probabilistische real-time systemen.

Hoofdstuk 7 bestudeert het volgende probleem: gegeven een continuous-time Markov-keten $\mathcal{C}$, en een lineaire real-time eigenschap weergegeven als een deterministische getimede automaat $\mathcal{A}$, wat is de kansgrootte van de verzameling paden van $\mathcal{C}$ die worden geaccepteerd door $\mathcal{A}$? Aangetoond wordt dat deze verzameling paden meetbaar is, en dat het berekenen van zijn kans kan worden gereduceerd tot het berekenen van de kans op bereikbaarheid in een stuksgewijs deterministisch Markov-proces. De kans op bereikbaarheid wordt gekarakterizeerd als de kleinste oplossing van een systeem van integrale vergelijkingen, en wordt benaderd door het oplossen van een systeem van partiële differentiaalvergelijkingen. Voor het speciale geval van enkele-klok deterministische getimede automaten, blijkt dat het systeem van integrale vergelijkingen kan worden getransformeerd naar een systeem van lineaire vergelijkingen waarin de coëfficiënten oplossingen zijn van gewone differentiaalvergelijkingen.

Hoofdstuk 8 richt zich op probabilistische getimede automaten, een uitbreiding van getimede automaten met discrete probabilistische splitsingen. Aangezien

de constructie van gebieden voor deze automaten vaak leidt tot een exponentiële toename van de omvang van de originele automaat, zijn reductietechnieken van het grootste belang. In dit hoofdstuk onderzoeken we probabilistische tijd-abstraherende bisimulatie (PTaB), een equivalentie-notie die abstraheert van precieze tijdsmomenten. Er wordt bewezen dat PTaB de probabilistische computation tree logica behoudt. De gebiedsequivalentie is een (zeer verfijnde) PTaB. We geven ook een niet-triviale aanpassing van het traditionele partitie-verfijningsalgoritme om het quotiënt onder de PTaBte berekenen. Dit algoritme is symbolisch in de zin dat equivalentie-klassen worden gerepresenteerd als polyhedra.

Curriculum Vitae

Taolue Chen was born on January 26th, 1980 in Taizhou, Jiangsu province, China. In 2002, he graduated with a bachelor degree in computer science from the Nanjing University in Nanjing, Jiangsu province, China. In 2005, he received his master degree, in computer science as well, from the same university, with a thesis on mobile process algebras, under the supervision of prof. dr. Jian Lu.

In July 2005, Chen started his PhD study in the Software Engineering Group 2 (Specification and Analysis of Embedded Systems) at the Centrum Wiskunde and Informatica (CWI) in Amsterdam, The Netherlands. Under the supervision of prof. dr. Wan Fokkink (Vrije Universiteit Amsterdam) and prof. dr. Jaco van de Pol (Universiteit of Twente), he worked on concurrency theory and formal methods, in particular, theory of process algebras and verification of probabilistic real-time systems. His work was carried out under the auspices of the BSIK/BRICKS project (Basic Research in Informatics for Creating the Knowledge Society). The present dissertation contains the results of this work.

Titles in the IPA Dissertation Series since 2005

E. Ábrahám. *An Assertional Proof System for Multithreaded Java -Theory and Tool Support-* . Faculty of Mathematics and Natural Sciences, UL. 2005-01

R. Ruimerman. *Modeling and Remodeling in Bone Tissue*. Faculty of Biomedical Engineering, TU/e. 2005-02

C.N. Chong. *Experiments in Rights Control - Expression and Enforcement*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-03

H. Gao. *Design and Verification of Lock-free Parallel Algorithms*. Faculty of Mathematics and Computing Sciences, RUG. 2005-04

H.M.A. van Beek. *Specification and Analysis of Internet Applications*. Faculty of Mathematics and Computer Science, TU/e. 2005-05

M.T. Ionita. *Scenario-Based System Architecting - A Systematic Approach to Developing Future-Proof System Architectures*. Faculty of Mathematics and Computing Sciences, TU/e. 2005-06

G. Lenzini. *Integration of Analysis Techniques in Security and Fault-Tolerance*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-07

I. Kurtev. *Adaptability of Model Transformations*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-08

T. Wolle. *Computational Aspects of Treewidth - Lower Bounds and Network Reliability*. Faculty of Science, UU. 2005-09

O. Tveretina. *Decision Procedures for Equality Logic with Uninterpreted Functions*. Faculty of Mathematics and Computer Science, TU/e. 2005-10

A.M.L. Liekens. *Evolution of Finite Populations in Dynamic Environments*. Faculty of Biomedical Engineering, TU/e. 2005-11

J. Eggermont. *Data Mining using Genetic Programming: Classification and Symbolic Regression*. Faculty of Mathematics and Natural Sciences, UL. 2005-12

B.J. Heeren. *Top Quality Type Error Messages*. Faculty of Science, UU. 2005-13

G.F. Frehse. *Compositional Verification of Hybrid Systems using Simulation Relations*. Faculty of Science, Mathematics and Computer Science, RU. 2005-14

M.R. Mousavi. *Structuring Structural Operational Semantics*. Faculty of Mathematics and Computer Science, TU/e. 2005-15

A. Sokolova. *Coalgebraic Analysis of Probabilistic Systems*. Faculty of Mathematics and Computer Science, TU/e. 2005-16

T. Gelsema. *Effective Models for the Structure of pi-Calculus Processes with Replication*. Faculty of Mathematics and Natural Sciences, UL. 2005-17

P. Zoeteweij. *Composing Constraint Solvers*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2005-18

J.J. Vinju. *Analysis and Transformation of Source Code by Parsing and Rewriting*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2005-19

M.Valero Espada. *Modal Abstraction and Replication of Processes with Data*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2005-20

A. Dijkstra. *Stepping through Haskell*. Faculty of Science, UU. 2005-21

Y.W. Law. *Key management and link-layer security of wireless sensor networks: energy-efficient attack and defense*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2005-22

E. Dolstra. *The Purely Functional Software Deployment Model*. Faculty of Science, UU. 2006-01

R.J. Corin. *Analysis Models for Security Protocols*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2006-02

P.R.A. Verbaan. *The Computational Complexity of Evolving Systems*. Faculty of Science, UU. 2006-03

K.L. Man and R.R.H. Schiffelers. *Formal Specification and Analysis of Hybrid Systems*. Faculty of Mathematics and Computer Science and Faculty of Mechanical Engineering, TU/e. 2006-04

M. Kyas. *Verifying OCL Specifications of UML Models: Tool Support and Compositionality*. Faculty of Mathematics and Natural Sciences, UL. 2006-05

M. Hendriks. *Model Checking Timed Automata - Techniques and Applications*. Faculty of Science, Mathematics and Computer Science, RU. 2006-06

J. Ketema. *Böhm-Like Trees for Rewriting*. Faculty of Sciences, VUA. 2006-07

C.-B. Breunesse. *On JML: topics in tool-assisted verification of JML programs*. Faculty of Science, Mathematics and Computer Science, RU. 2006-08

B. Markvoort. *Towards Hybrid Molecular Simulations*. Faculty of Biomedical Engineering, TU/e. 2006-09

S.G.R. Nijssen. *Mining Structured Data*. Faculty of Mathematics and Natural Sciences, UL. 2006-10

G. Russello. *Separation and Adaptation of Concerns in a Shared Data Space*. Faculty of Mathematics and Computer Science, TU/e. 2006-11

L. Cheung. *Reconciling Nondeterministic and Probabilistic Choices*. Faculty of Science, Mathematics and Computer Science, RU. 2006-12

B. Badban. *Verification techniques for Extensions of Equality Logic*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2006-13

A.J. Mooij. *Constructive formal methods and protocol standardization*. Faculty of Mathematics and Computer Science, TU/e. 2006-14

T. Krilavicius. *Hybrid Techniques for Hybrid Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2006-15

M.E. Warnier. *Language Based Security for Java and JML*. Faculty of

Science, Mathematics and Computer Science, RU. 2006-16

V. Sundramoorthy. *At Home In Service Discovery*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2006-17

B. Gebremichael. *Expressivity of Timed Automata Models*. Faculty of Science, Mathematics and Computer Science, RU. 2006-18

L.C.M. van Gool. *Formalising Interface Specifications*. Faculty of Mathematics and Computer Science, TU/e. 2006-19

C.J.F. Cremers. *Scyther - Semantics and Verification of Security Protocols*. Faculty of Mathematics and Computer Science, TU/e. 2006-20

J.V. Guillen Scholten. *Mobile Channels for Exogenous Coordination of Distributed Systems: Semantics, Implementation and Composition*. Faculty of Mathematics and Natural Sciences, UL. 2006-21

H.A. de Jong. *Flexible Heterogeneous Software Systems*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2007-01

N.K. Kavaldjiev. *A run-time reconfigurable Network-on-Chip for streaming DSP applications*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2007-02

M. van Veelen. *Considerations on Modeling for Early Detection of Abnormalities in Locally Autonomous Distributed Systems*. Faculty of Mathematics and Computing Sciences, RUG. 2007-03

T.D. Vu. *Semantics and Applications of Process and Program Algebra*. Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2007-04

L. Brandán Briones. *Theories for Model-based Testing: Real-time and Coverage*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2007-05

I. Loeb. *Natural Deduction: Sharing by Presentation*. Faculty of Science, Mathematics and Computer Science, RU. 2007-06

M.W.A. Streppel. *Multifunctional Geometric Data Structures*. Faculty of Mathematics and Computer Science, TU/e. 2007-07

N. Trčka. *Silent Steps in Transition Systems and Markov Chains*. Faculty of Mathematics and Computer Science, TU/e. 2007-08

R. Brinkman. *Searching in encrypted data*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2007-09

A. van Weelden. *Putting types to good use*. Faculty of Science, Mathematics and Computer Science, RU. 2007-10

J.A.R. Noppen. *Imperfect Information in Software Development Processes*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2007-11

R. Boumen. *Integration and Test plans for Complex Manufacturing Systems*. Faculty of Mechanical Engineering, TU/e. 2007-12

A.J. Wijs. *What to do Next?: Analysing and Optimising System Behaviour in Time*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2007-13

C.F.J. Lange. *Assessing and Improving the Quality of Modeling: A Series of Empirical Studies about the UML*. Faculty of Mathematics and Computer Science, TU/e. 2007-14

T. van der Storm. *Component-based Configuration, Integration and Delivery*. Faculty of Natural Sciences, Mathematics, and Computer Science,UvA. 2007-15

B.S. Graaf. *Model-Driven Evolution of Software Architectures*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2007-16

A.H.J. Mathijssen. *Logical Calculi for Reasoning with Binding*. Faculty of Mathematics and Computer Science, TU/e. 2007-17

D. Jarnikov. *QoS framework for Video Streaming in Home Networks*. Faculty of Mathematics and Computer Science, TU/e. 2007-18

M. A. Abam. *New Data Structures and Algorithms for Mobile Data*. Faculty of Mathematics and Computer Science, TU/e. 2007-19

W. Pieters. *La Volonté Machinale: Understanding the Electronic Voting Controversy*. Faculty of Science, Mathematics and Computer Science, RU. 2008-01

A.L. de Groot. *Practical Automaton Proofs in PVS*. Faculty of Science, Mathematics and Computer Science, RU. 2008-02

M. Bruntink. *Renovation of Idiomatic Crosscutting Concerns in Embedded Systems*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2008-03

A.M. Marin. *An Integrated System to Manage Crosscutting Concerns in Source Code*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2008-04

N.C.W.M. Braspenning. *Model-based Integration and Testing of High-tech Multi-disciplinary Systems*. Faculty of Mechanical Engineering, TU/e. 2008-05

M. Bravenboer. *Exercises in Free Syntax: Syntax Definition, Parsing, and Assimilation of Language Conglomerates*. Faculty of Science, UU. 2008-06

M. Torabi Dashti. *Keeping Fairness Alive: Design and Formal Verification of Optimistic Fair Exchange Protocols*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2008-07

I.S.M. de Jong. *Integration and Test Strategies for Complex Manufacturing Machines*. Faculty of Mechanical Engineering, TU/e. 2008-08

I. Hasuo. *Tracing Anonymity with Coalgebras*. Faculty of Science, Mathematics and Computer Science, RU. 2008-09

L.G.W.A. Cleophas. *Tree Algorithms: Two Taxonomies and a Toolkit*. Faculty of Mathematics and Computer Science, TU/e. 2008-10

I.S. Zapreev. *Model Checking Markov Chains: Techniques and Tools*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2008-11

M. Farshi. *A Theoretical and Experimental Study of Geometric Networks*. Faculty of Mathematics and Computer Science, TU/e. 2008-12

G. Gulesir. *Evolvable Behavior Specifications Using Context-Sensitive Wildcards*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2008-13

F.D. Garcia. *Formal and Computational Cryptography: Protocols, Hashes and Commitments*. Faculty of Science, Mathematics and Computer Science, RU. 2008-14

P. E. A. Dürr. *Resource-based Verification for Robust Composition of Aspects*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2008-15

E.M. Bortnik. *Formal Methods in Support of SMC Design*. Faculty of Mechanical Engineering, TU/e. 2008-16

R.H. Mak. *Design and Performance Analysis of Data-Independent Stream Processing Systems*. Faculty of Mathematics and Computer Science, TU/e. 2008-17

M. van der Horst. *Scalable Block Processing Algorithms*. Faculty of Mathematics and Computer Science, TU/e. 2008-18

C.M. Gray. *Algorithms for Fat Objects: Decompositions and Applications*. Faculty of Mathematics and Computer Science, TU/e. 2008-19

J.R. Calamé. *Testing Reactive Systems with Data - Enumerative Methods and Constraint Solving*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2008-20

E. Mumford. *Drawing Graphs for Cartographic Applications*. Faculty of Mathematics and Computer Science, TU/e. 2008-21

E.H. de Graaf. *Mining Semi-structured Data, Theoretical and Experimental Aspects of Pattern Evaluation*. Faculty of Mathematics and Natural Sciences, UL. 2008-22

R. Brijder. *Models of Natural Computation: Gene Assembly and Membrane Systems*. Faculty of Mathematics and Natural Sciences, UL. 2008-23

A. Koprowski. *Termination of Rewriting and Its Certification*. Faculty of Mathematics and Computer Science, TU/e. 2008-24

U. Khadim. *Process Algebras for Hybrid Systems: Comparison and Development*. Faculty of Mathematics and Computer Science, TU/e. 2008-25

J. Markovski. *Real and Stochastic Time in Process Algebras for Performance Evaluation*. Faculty of Mathematics and Computer Science, TU/e. 2008-26

H. Kastenberg. *Graph-Based Software Specification and Verification*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2008-27

I.R. Buhan. *Cryptographic Keys from Noisy Data Theory and Applications*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2008-28

R.S. Marin-Perianu. *Wireless Sensor Networks in Motion: Clustering Algorithms for Service Discovery and Provisioning*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2008-29

M.H.G. Verhoef. *Modeling and Validating Distributed Embedded Real-Time Control Systems*. Faculty

of Science, Mathematics and Computer Science, RU. 2009-01

M. de Mol. *Reasoning about Functional Programs: Sparkle, a proof assistant for Clean*. Faculty of Science, Mathematics and Computer Science, RU. 2009-02

M. Lormans. *Managing Requirements Evolution*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2009-03

M.P.W.J. van Osch. *Automated Model-based Testing of Hybrid Systems*. Faculty of Mathematics and Computer Science, TU/e. 2009-04

H. Sozer. *Architecting Fault-Tolerant Software Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-05

M.J. van Weerdenburg. *Efficient Rewriting Techniques*. Faculty of Mathematics and Computer Science, TU/e. 2009-06

H.H. Hansen. *Coalgebraic Modelling: Applications in Automata Theory and Modal Logic*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2009-07

A. Mesbah. *Analysis and Testing of Ajax-based Single-page Web Applications*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2009-08

A.L. Rodriguez Yakushev. *Towards Getting Generic Programming Ready for Prime Time*. Faculty of Science, UU. 2009-9

K.R. Olmos Joffré. *Strategies for Context Sensitive Program Transformation*. Faculty of Science, UU. 2009-10

J.A.G.M. van den Berg. *Reasoning about Java programs in PVS using JML*. Faculty of Science, Mathematics and Computer Science, RU. 2009-11

M.G. Khatib. *MEMS-Based Storage Devices. Integration in Energy-Constrained Mobile Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-12

S.G.M. Cornelissen. *Evaluating Dynamic Analysis Techniques for Program Comprehension*. Faculty of Electrical Engineering, Mathematics, and Computer Science, TUD. 2009-13

D. Bolzoni. *Revisiting Anomaly-based Network Intrusion Detection Systems*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-14

H.L. Jonker. *Security Matters: Privacy in Voting and Fairness in Digital Exchange*. Faculty of Mathematics and Computer Science, TU/e. 2009-15

M.R. Czenko. *TuLiP - Reshaping Trust Management*. Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2009-16

T. Chen. *Clocks, Dice and Processes*. Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2009-17