

Real and stochastic time in process algebras for performance evaluation

Citation for published version (APA):

Markovski, J. (2008). *Real and stochastic time in process algebras for performance evaluation*. [Phd Thesis 1 (Research TU/e / Graduation TU/e), Mathematics and Computer Science]. Technische Universiteit Eindhoven. <https://doi.org/10.6100/IR637756>

DOI:

[10.6100/IR637756](https://doi.org/10.6100/IR637756)

Document status and date:

Published: 01/01/2008

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Real and Stochastic Time
in Process Algebras
for Performance Evaluation

by Jasen Markovski

©Jasen Markovski
IPA Dissertation Series 2008-26
Typeset using L^AT_EX2e
Printed by University Press Facilities, Eindhoven
Cover design by Jasen Markovski, adaptation by Paul Verspaget

A catalogue record is available
from the Eindhoven University of Technology Library
ISBN: 978-90-386-1394-9



The work in this thesis has been carried out under the auspices of the research school IPA (Institute for Programming research and Algorithmics).



The author was employed at the Eindhoven University of Technology, supported by the Dutch BSIK/BRICKS project AFM 3.2.

Real and Stochastic Time in Process Algebras for Performance Evaluation

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
Rector Magnificus, prof.dr.ir. C.J. van Duijn, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op donderdag 2 oktober 2008 om 16.00 uur

door

Jasen Markovski

geboren te Skopje, Macedonië

Dit proefschrift is goedgekeurd door de promotor:

prof.dr. J.C.M. Baeten

Copromotor:

dr. E.P. de Vink

Preface

This thesis is the final result of four years of research done in the Formal Methods Group at Eindhoven University of Technology in The Netherlands. First of all, I would like to thank my supervisor, professor Jos Baeten, for giving me a position in the group. He has provided me with a great deal of support, as well as tolerance, understanding, and flexibility as much as a supervisor can give. I am grateful for the trust that was given to me in the beginning of my research and the freedom to follow my own path at the later stages. I am also thankful for his endurance of the many ‘fiery’ discussions and the knowledge that he transferred to me.

I owe a lot to my co-supervisor Erik de Vink. He always had my best interest in mind, offering a helping hand every time I needed it. I gained expertise in writing papers under his guidance and he was always willing to share his experiences and teach me the tricks of the trade. I thank him for always being there for me and giving me the kind of endless support that any PhD student hopes for.

Many results presented in this thesis are a product of joint work. Nikola Trčka was involved in most of it as one of my closest co-workers and a great friend. In the beginning, we had long discussions, in which he was unselfishly transferring all of his knowledge to me. In the past four years, he always shared his ideas, open to criticism, and promptly sharing his own considering my inventions as well. He crushed many of my theories, never giving up in building new ones. The quality of my research and my expertise as a researcher would never be on this level if it were not for him.

I have learned a lot about writing, being more precise, and expressing myself better and clearer from Bas Luttik. I thank him for always finding the time and energy to read and comment on everything that Nikola and I put in front of him.

I would also like to thank Ana Sokolova for her support in and out of the workplace. I learned a lot from her both in her classes in Skopje as a student, as well as during my stay in the Netherlands, where we were involved in the

same research. Here, I continued to learn from her thoroughness, attention to detail, and ease of expression.

One year ago, Sonja Georgievska, a former colleague of mine, and Suzana Andova, her co-supervisor, joined our group. Almost immediately we began collaborating, which resulted in some interesting research. Sonja also greatly contributes to the never-ending discussions together with Nikola, and her input was always valued.

I thank the members of the reading committee professor Koos Rooda, professor Joost-Pieter Katoen, Manuel Núñez, and Pedro D’Argenio for reviewing the manuscript and giving me valuable comments that improved the quality of this thesis. I also thank professor Holger Hermanns, professor Joost-Pieter Katoen, and professor Tom Henzinger for inviting me to visit and present my work. I also thank professor Frank de Boer for having me in the BSIK/BRICKS AFM 3.2 project, which funded my research. Additionally, I thank professor Koos Rooda for offering me a post-doc position in his group, which I gladly accepted.

I thank my colleagues at the Formal Methods Group for contributing to a relaxed and productive working atmosphere. Special thanks goes to Simona Orzan as my part-time office-mate for almost three years during which she politely endured my discussions with Nikola and the “q-doi” stuff. I also thank miss Joosten for always being supportive, helpful, and ready to laugh. Later, Astrid Volkers came in her position and provided a cheerful company, which I enjoyed pretty much. I thank Tijn Borghuis for leading several interesting IPA days and for inviting me to talk there. I thank Ruurd Kuiper for his support during his Java classes. I also managed to learn a lot from Jing Pan, who exposes me to her different points of view. I would also like to thank Mohammad Mousavi for always finding time to answer my questions. I thank Walter van Niftrik for his support to my research during his master project.

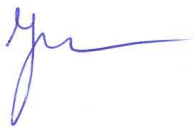
My stay in the Netherlands would have not been as pleasant if it were not for the many new and old friends. They know who they are and that they are really treasured, but still I want to mention some of them. Ana has helped me a lot in the beginning and provided me with more than just useful advice. George has become a great friend of mine and he introduced me to one of my favorite sports. So, I thank him for the many bruises and some exhilarating experiences. I was lucky to also have some of my old friends here in Eindhoven and I thank Bate Žare and Src for all the time we spent together. With Nikola and Marija I had many pleasant gatherings and their company and advice is always welcome. A group of Spanish people made my life very interesting and playful and I thank them for their amusing

company, especially Zlato, Irene, Emily, and Emma. Sonja and Starski came here one year ago and make my life here even more delightful. I also enjoy each of the Nataša's parties and Nadezhda's recipes. Grga and Nataša have also been great company and I wish them the best in their new home.

I cannot forget about my friends and family back home in Macedonia. They have always made my trips there memorable, making me feel like I almost never left home. Some of them came to visit, some of them I hope soon will. I thank Mire for always being there, never giving up on me, nor judging me. Special thanks goes to Kum and Nevestička who always found time in their very busy schedules. I also thank Dac for always keeping in touch, although I was not always at my best behavior. Jiggy and Šatana provided an enjoyable company, very often until very late in the evening. Suzi has always proven to me that all things are possible if you put your mind to it. Topče has been a great friend, always providing delightful company and conversation. Pileto Šareno always pointed me in the right direction of the best cafés and restaurants. Cecolina has always provided me with good advice, having in mind my best interest. I also thank Goce, Boki, Dečki, and Vesna for staying in touch despite their hectic obligations. Running out of space and time I apologize for not mentioning the rest of you. I hope you know you are much appreciated.

Endless amount of gratitude goes to Meri, who managed to make my life interesting, eventful, warm, comfortable, and full of love and sunshine, even on the cloudiest of days.

Finally, I want to express my deepest gratitude and appreciation for my parents Smile and Slavica, and my sister Jasminka, who have always been there to support me. Without their love, support, compassion, selfless sacrifice, and vision I would have never become the person that I am.



Jasen Markovski

Eindhoven, August 14th, 2008

Summary

Real and Stochastic Time in Process Algebras for Performance Evaluation

Process algebras are formalisms for abstract modeling of systems for the purpose of qualitative verification and quantitative evaluation. The purpose of verification is to show that the system behaves correctly, e.g., it does not contain a deadlock or a state with some desired property is eventually going to be reached. The quantitative or performance evaluation part gives an approximation how well the system will behave, e.g., the average time of a message to get through is 10 time units or the utilization (percentage of time that something is used) of some machine is 23.5 percent.

Originally, process algebras were only developed for qualitative modeling, but gradually they have been extended with time, probabilities, and Markovian (exponential) and generally-distributed stochastic time. The extensions up to stochastic time typically conservatively extended previous well-established theories. However, mostly due to the nature of the underlying (non-)Markovian performance models, the stochastic process algebras were built from scratch. These extensions were carried out as orthogonal extensions of untimed process theories with exponential delays or stochastic clocks. The underlying performance model is obtained by abstracting from the qualitative behavior using some weak behavioral equivalence.

The thesis investigates several issues: (1) What is the relationship between discrete real and generally-distributed stochastic time in the process theories? (2) Is it possible, and if so, how, to extend timed process theories with stochastic time? (3) Reversely, is it possible, and if so, how, to embed discrete real time in generally distributed process theories? Additionally, (4) is the abstraction using the weak behavioral equivalence in Markovian process theories (and other modeling formalisms as well) performance preserving, and is such an approach compositional? In the end, (5) how can we do performance analysis using discrete-time and probabilistic choices?

The contents of the thesis is as follows. First, we introduce the central concept of a race condition that defines the interaction between stochastic timed delays. We introduce a new type of race condition, which enables the synchronization of stochastic delays with the same sample as in timed process theories. This gives the basis for the notion of a timed delay in a racing context, which models the expiration of stochastic delays. In this new setting, we define a strong bisimulation relation that deals with the (probabilistic) race condition on a symbolic level. Next, we show how to derive stochastic delays as guarded recursive specification involving timed delays in a racing context and we derive a ground-complete stochastic-time process theory. Then, we take the opposite viewpoint and we develop a stochastic process theory from scratch, relying on the same interpretation of the race condition. We embed real time in the stochastic-time setting by using context-sensitive interpolation, a restricted notion of time additivity. Afterwards, we turn to Markovian process theories and we show compositionality of the Markov reward chains with fast and silent transitions with respect to lumping-based and reduction-based aggregation methods. These methods can be used to show preservation of performance measures when eliminating probabilistic choices and non-deterministic silent steps in Markovian process theories. Then, we specify the underlying model of probabilistic timed process theories as a discrete-time probabilistic reward graph and we show its transformation to a discrete-time Markov reward chain. The approach is illustrated by extending the environment of the modeling language χ . The developed theories are illustrated by specifying a version of the concurrent alternating bit protocol and analyzing it in the χ toolset.

Contents

1	Introduction	1
1.1	Describing a Testing System	1
1.2	Formal Methods and Performance Evaluation	10
1.3	Process Algebras	12
1.4	Timed and Probabilistic Extensions	13
1.5	Markovian Time Extensions	14
1.6	Extensions with Generally-Distributed Stochastic Time	16
1.7	Outline and Contribution	20
2	Race Condition	25
2.1	Racing Stochastic Delays	26
2.2	Stochastic Delay Prefix	28
2.3	Dependent and Independent Race Condition	29
2.4	Timed Delays in a Racing Context	31
2.5	Design Choices	34
2.6	Summary	35
3	Process Theory TCP^{drst}	37
3.1	Racing Timed Transition Schemes	37
3.2	Probabilistic Timed Transition Systems	39
3.3	Bisimulation Relation	41
3.4	Signature	43
3.5	Auxiliary Operations	44
3.6	Naming Conflicts	46
3.7	Structural Operational Semantics	47
3.8	α -conversion	51
3.9	Term Model	57
3.10	Summary	59

4	Equational Theory	61
4.1	Renaming of Independent Delays	61
4.2	Dependence Scope	63
4.3	Alternative Composition	64
4.4	Renaming of Independent Delays	67
4.5	Encapsulation	69
4.6	Parallel Composition	71
4.7	Maximal Progress	74
4.8	Head Normal Form	75
4.9	Ground Completeness	76
4.10	Guarded Recursive Specifications	77
4.11	Summary	79
5	Process Theory $\text{DTCP}_{\text{rec}}^{\text{dst}}$	81
5.1	Delayable Action Prefix and Delayable Deadlock	81
5.2	Stochastic Delay Prefix	82
5.3	Interaction between the Prefix Operators	84
5.4	Signature	86
5.5	Dependence Scope and Encapsulation	87
5.6	Alternative Composition	88
5.7	α -conversion	93
5.8	Parallel Composition	93
5.9	Maximal Progress	95
5.10	Head Normal Form	96
5.11	Race-Complete Process Specifications	98
5.12	The $G/G/1/\infty$ Queue	99
5.13	Summary	101
6	Extending Real Time with Stochastic Time	103
6.1	Overview of Stochastic Bisimulation Relations	103
6.2	Extending Real Time with Stochastic Time	104
6.3	Context-Sensitive Interpolation	106
6.4	Stochastic Process Theory $\text{TCP}_{\text{rec}}^{\text{st}}$	108
6.5	Stochastic Transition Schemes	110
6.6	Bisimulation	113
6.7	Structural Operational Semantics	114
6.8	Expansion of the Parallel Composition	120
6.9	Embedding Real Time as Dirac Stochastic Time	121
6.10	Summary	122

7	Aggregation Methods for Markov Reward Chains with Fast and Silent Transitions	123
7.1	Extended Markovian Models	128
7.2	Aggregation Methods	134
7.3	Relational Properties	144
7.4	Parallel Composition and Compositionality	146
7.5	Summary	154
8	Analyzing the Concurrent Alternating Bit Protocol	155
8.1	The Language χ	155
8.2	Discrete-Time Probabilistic Reward Graphs	157
8.3	The Concurrent Alternating Bit Protocol	173
8.4	Specification and Analysis in χ	175
8.5	Summary	180
9	Conclusions and Future Work	181
	Bibliography	185
	Curriculum Vitae	195

Chapter 1

Introduction

In this thesis we deal with timed and stochastic specifications of complex systems for the purpose of verification and performance analysis. Although initially targeted at the analysis of software-intensive systems, the techniques developed are applicable to a wide range of timed and stochastic distributed systems. The process theories developed in this thesis are inherently of a technical nature. A little of the reader's patience is required to digest the unavoidable overhead preceding the presentation of results.

We begin by informally presenting the topics explored in this thesis by means of an example. We aim to provide the reader outside the fields of formal methods and performance analysis with a better insight into the matters investigated in the sequel. Then, we give an overview of the topics of interest by chronologically discussing the timed and stochastic extensions of process algebras. We finish the introduction by sketching the structure of the thesis, underlying the main results and contributions, as well as listing the supporting publications.

1.1 Describing a Testing System

We start off with modeling a simple testing system using paradigms from formal methods and performance analysis. Using this example we point out the key issues discussed in this thesis using an informal language, terminology, and notation.

We begin by describing the qualitative behavior of the testing system, i.e., the activities or actions that an observer of this system might be interested in. The system is depicted in Figure 1.1.

We observe the testing system in isolation in the sense that we do not model the whole environment, i.e., we do not care how the products are

produced (although we do take into account the temporal and probabilistic properties of the arrival process) and what happens with the defective or approved products. We clearly separate the system in four components, each one with its own purpose: (1) “Arrival of products”, which sends the products for testing, (2) “Product tester”, which receives the product, begins its testing, and determines whether the product is defective or approved, (3) “Receiver of defective products”, which consumes the defective products, and (4) “Receiver of approved products”, which consumes the approved products.

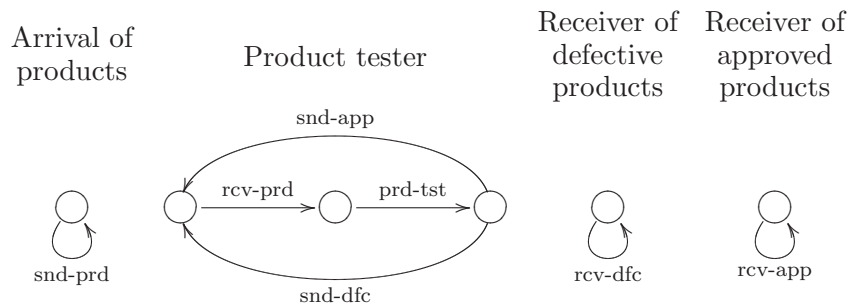


Figure 1.1: Qualitative description of the components of the testing system

In general, we visualize models using graphs or transition systems in which the states denote points with different behavior and the outgoing transitions depict the activities. We consider the leftmost state as the starting state of each graph. For example, in the component “Arrival of products” there is only one outgoing transition labeled “snd-prd” denoting that a product has arrived and it is ready to be sent further. This component has only one state, meaning that it is just responsible for delivering products in the system. The following component “Product tester”, first has to receive a product in order to begin its operation as given by the transition “rcv-prd”. The activities “snd-prd” and “rcv-prd” are synchronizing, meaning that when the former component sends a product to the latter there is a synchronized communication between the components. Later, we denote this synchronization activity as “prd-snt”. After the product has been received, the tester begins the testing of the product (as given by the transition “prd-tst”) and makes a choice whether the product is defective or approved as depicted by the two outgoing transitions “snd-dfc” and “snd-app”, respectively. Note that there is no quantification on the way that the choice is made, i.e., it is made nondeterministically. In this situation we also say that

the system is underspecified.

The components of the system can be merged to give the observable behavior of the testing system as depicted at the left-hand side in Figure 1.2.

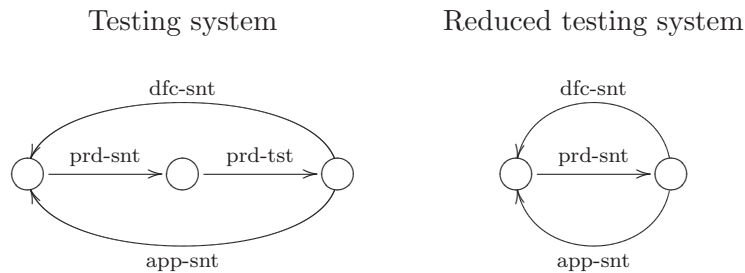


Figure 1.2: Qualitative description of the testing system

The arriving products are now sent to the tester by synchronizing the sending and receiving activity of the product from the component “Arrival of products” to the component “Product tester”, which is denoted by the ‘synchronizing’ transition “*prd-snt*”. Afterwards, the product is being tested and either defective or approved products are ‘communicated’ to the corresponding receiver, denoted by the synchronization activities “*dfc-snt*” and “*app-snt*”, respectively.

For the purpose of verification of the correct (observable) functioning of the testing system, we sometimes wish to ‘abstract’ from the internal workings of the system. At the right-hand side of Figure 1.2 we depict the reduced model of the testing system. Here, we do not care about the actual activities involving the testing of the product. Instead, we treat the tester as a black box, assuming that the testing is done in a proper manner. From the observable behavior of the system we can now ensure that the products that go inside the tester eventually come out labeled either as defective or approved.

Next, we proceed by quantifying the temporal aspects of the system. For example, if we wish to specify that the products arrive every three units of time, then we can extend the specification of the component “Arrival of products” as depicted in Figure 1.3. The time delays are represented by transitions labeled by a number that represents the duration of the delay. We extend the specification of the component “Product tester” as well. Let us assume that finding a defective product takes two units of time. Approved products need to be labeled, for which we assign additional two units of

time, amounting to four units of time required for the testing of approved products.

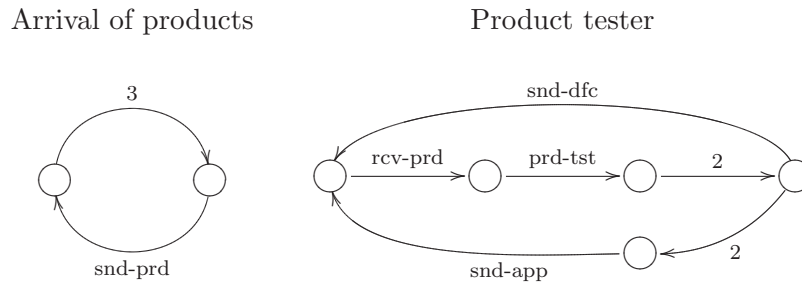


Figure 1.3: Timed description of the components “Arrival of products” and “Product tester”

Here, an interesting phenomenon occurs: we do not see in the description of the component “Product tester” in Figure 1.3 a composite delay of four time units. We do see, however, a delay of two time units preceding a choice between the transition “snd-dfc”, which denotes testing of defective products, and a delay of two time units followed by the transition “snd-app”, which labels approved products. The description implies that the passage of time for testing defective or approved products is observed simultaneously. The passage of time by itself does not make a choice, but the activities of the system are the ones that make it. This temporal property is referred to as time determinism. Another implication from the above discussion is that two delays, each with duration of two units of time are considered as equivalent to one composite delay of four time units. This temporal property is referred to as time additivity. Time determinism and time additivity are the identifying properties of passage of time.

The timed behavior of the testing system is depicted in Figure 1.4.

Here, we see that both time determinism and time additivity play a role. The initial product comes into the system after three units of time, which are depicted as two successive delays of two and one time unit, respectively. If the tested product is defective, then the result is known in two time units, leaving a time-unit gap before the arrival of the next product. If the tested product is approved, then the testing operation costs four time units. Thus, the following arriving product has already waited an extra time unit in the component “Arrival of products” for synchronization with the component “Product tester”. This means that we implicitly assume that the activities

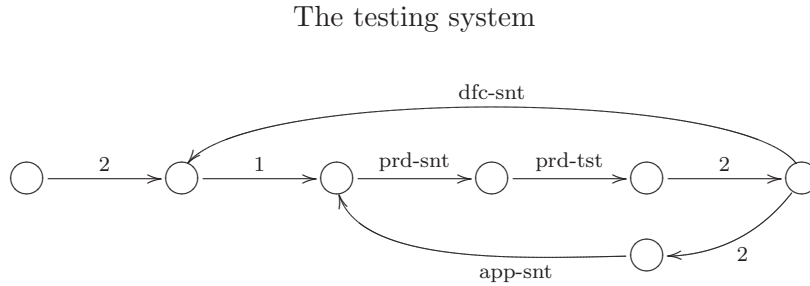


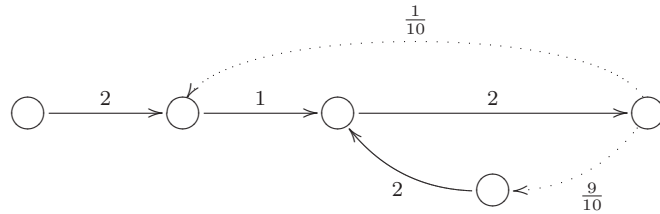
Figure 1.4: Timed description of the testing system

of the components (e.g., “snd-prd”, “rcv-dfc”) are delayable, i.e., they allow passage of time before the other component is ready to synchronize. The synchronization itself is assumed to happen instantly as there is no point in waiting, so, e.g., the activities “prd-snt” or “dfc-snt” happen as soon as possible and are deemed undelayable. This assumption is also known as the maximal progress of time. We also note that internal activities that have no synchronizing counterpart, like “prd-tst”, are also typically considered as undelayable.

Looking at the testing system from the perspective of performance analysis, the emphasis is put on the quantitative aspects of the system, instead of the qualitative ones. So, the transitions of the system that do not carry any quantitative information are superfluous and they do not exist in the specification. In Figure 1.5 we depict the testing system from Figure 1.4 suitable for performance analysis. The choice whether the product is defective or approved is now quantified by an explicit probabilistic choice (denoted by dotted arrows), which expresses that on average 9 out of 10 products are approved. It is assumed that the probabilistic choice is immediate, i.e., its resolution does not consume any time.

We note, however, that this model is incomplete in the sense that additional information in form of rewards or costs is required to specify the performance measures of interest. The rewards are numbers assigned to states that are used to form a meaningful weighted sum of the fraction of time that the system spends in each of its states. For example, if we wish to find out the long-run utilization of the tester, then reward 0 is assigned to the leftmost two states and reward 1 is assigned to the rightmost three states. The intuition behind such distribution of rewards is that the tester is employed in the rightmost three states, which can be deduced by com-

The testing system

**Figure 1.5:** Timed description suitable for performance analysis

paring the specifications in Figures 1.4 and 1.5. Thus, this distribution of rewards will collect the fraction of time that the system on average spends for testing of products. What is left to do is to compute the fraction of time that the process spends in each state and multiply it with the rewards, which amounts to a long-run utilization of $\frac{38}{39}$. We deal with this class of performance models in the last part of the thesis.

For a precise performance modeling, the deterministic delays of the timed description of Figure 1.5 are sometimes insufficient. The most general manner of approximating passage of time is by using generally-distributed stochastic time. The most prominent performance models with generally-distributed stochastic time are the generalized semi-Markov processes. They employ decreasing stochastic clocks that can sample from any probability distribution. In a state the clocks can be reset, which is denoted by the name of the clock in the state. When the clock is reset, it is assigned a value or sample and it immediately starts counting downwards. The expiration of the clock, i.e., its reaching zero, is denoted by an outgoing transition with its name. We note that when sampling from continuous distributions, the probability that two clocks expire at the same time is zero. On the left-hand side in Figure 1.6 we give a generalized semi-Markovian description of the testing system, assuming that the clocks cannot expire simultaneously.

The clock a is assigned to the arriving delay of the component “Arrival of products”, the clock d is assigned to the delay required to test a defective product, and the clock p is assigned to the delay required to test an approved product. Recall that in the timed specification the time delays were explicitly merged according to the principle of time determinism. When doing performance analysis the delays are typically represented as separate

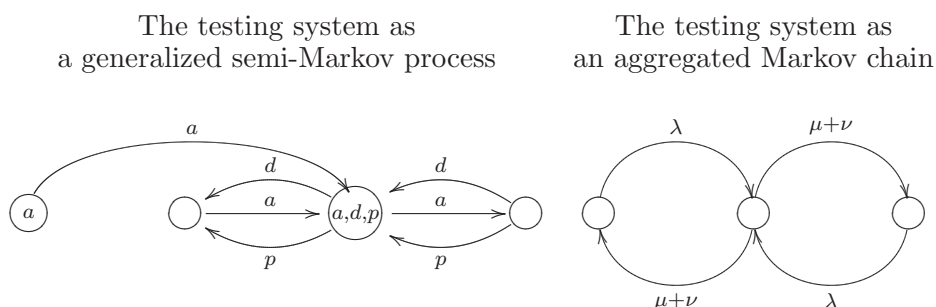


Figure 1.6: Descriptions of the testing system from a performance analytic point of view

constructs and their interaction is guided by a so-called race condition. The race condition states that the transitions guided by the (simultaneously-expiring) clock(s) with the smallest sample will be taken. Notably, the property of time determinism is preserved, although each clock can have a separate outgoing transition as depicted in Figure 1.6. A major part of this thesis is dedicated to the relationship between deterministic or real time and stochastic time and the preservation of the real-time properties in race condition semantics.

Coming back to our example, the testing system introduces the initial product after the expiration of a . In the next state, all clocks are reset, meaning that there is simultaneous passage of time for the arrival of the successive product, its testing as defective product, and its testing as an approved product. In this state there is a race, which can have only one winner, as for the sake of simplicity we assume that no two clocks can expire simultaneously. Then, two things can happen: (1) either the product is labeled defective or approved and sent to the corresponding receiver, which is denoted by the outgoing transitions labeled by d and p , respectively, or (2) a new product has arrived in the component "Arrival of products" and it is waiting to be received by the tester as depicted by the outgoing transition a . Note that in both cases there is no resetting of clocks as no new activities are started. In the former case, the system waits for the a clock to expire, i.e., it expects the successive product for testing, whereas in the latter case the new product is waiting for synchronization as the tester has to finish the current testing operation. We note that the starting state is unique, as it is the only state in which there is no simultaneous expiration of the three

clocks (cf. the starting states of the timed descriptions given in Figures 1.4 and 1.5).

Often performance evaluation is done assuming only exponentially distributed clocks or delays. In this case the stochastic delay is simply denoted by the parameter of the negative exponential distribution as depicted in the right-hand side in Figure 1.6. The exponentially distributed delays have several important properties: (1) such delays are memoryless, meaning that passage of time does not alter the distribution of the delay to its expiration, which is also a unique property of the negative exponential distribution in the continuous domain, (2) they are closed for the minimum, implying that multiple delays originating and ending in same states can be replaced only by one delay, which parameter is the sum of the parameters of the other delays, and (3) knowing nothing about the distributions of the delays except for their mean, they are statistically the most suitable fit.

If we assume that the clocks a , d , and p are exponentially distributed with parameters λ , μ , and ν , respectively, then the generalized semi-Markov process, becomes a continuous-time Markov chain. Due to the memoryless property, the starting state can now be merged with its target state. This is because the simultaneous expiration of the clocks does not alter the distribution of a , which remains exponentially distributed with a parameter λ . Also, the winning transitions of d and p are represented by only one transition which represents the shortest sample of d and p distributed with parameter $\mu + \nu$. The continuous-time Markovian representation is given at the right-hand side in Figure 1.6.

To support verification and performance analysis from the same specification, one can add, e.g., exponential delays to the untimed description of the components in Figure 1.1. This leads to a Markovian description of the testing system as depicted in Figure 1.7.

For the purpose of doing performance analysis the qualitative information from the specification should be eliminated. This reduction should also guarantee that the performance of the model depicted in the bottom of Figure 1.7 is equal to the performance of the pure Markov chain depicted on the right-hand side in Figure 1.6. In this thesis, we study the properties of aggregation methods based on stochastic interpretations of the action transitions, devoid of their meaning, as infinitely fast exponential delays with an unknown parameter. We show that the aggregation methods induce pre-order relations and that the aggregations themselves can be performed in a compositional manner. So, the aggregation of the components is allowed before composing the complete system, which reduces the space required to calculate the final process.

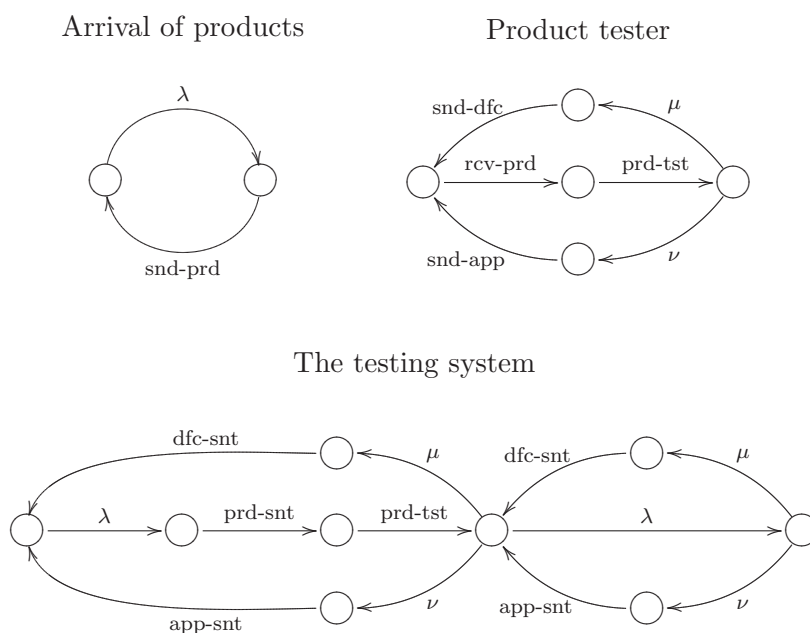
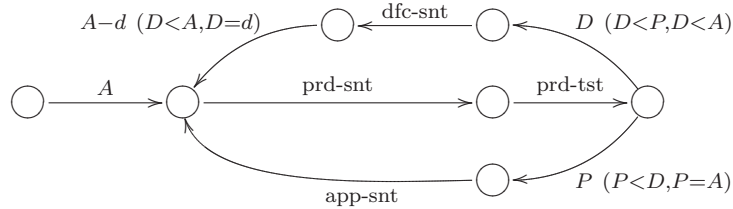


Figure 1.7: Markovian description

Additionally, we wish to explore the domain of generally-distributed stochastic delays and their relation to real (deterministic) delays as in the description in Figure 1.4. The expirations of the clocks in the generalized semi-Markov processes do preserve time determinism, but they do not straightforwardly support time additivity. For that purpose, we look at stochastic time from a different perspective, by using conditionally distributed delays as depicted in Figure 1.8.

Here, we model stochastic delays as (conditional) random variables that guide the distribution of the delays. Different from the decreasing stochastic clocks as given in Figure 1.6, here we probabilistically decide on the winner(s) of the race and condition the distribution of the remaining stochastic delays according to the exhibited winning sample. The race condition is partially represented in Figure 1.8 as we assume only two (out of seven) possible outcomes of the race between the random variables A , D , and P which guide the stochastic delays of the arrival of products, testing for a defective, and testing for an approved product, respectively. In case one stochastic delay depends on the sample of the shorter one (as in the case

The testing system (partial representation)

**Figure 1.8:** Description with generally-distributed delays

when $D < P$ and $D < A$), the remaining distribution of A must be adjusted by the sample d exhibited by the stochastic delay guided by D as indicated by the label of the topmost left transition. This extended representation of the race allows for a deeper understanding of the relation between the winning and the losing delays of the race and provides a better insight into the relationship between the race condition and passage of time. It is the foundation upon which the first and largest part of this thesis is built.

We continue with a more formal introduction to the topics dealt with within this thesis. We give an overview of the timed and stochastic extensions of process algebras and relate to the relevant concepts covered in this thesis.

1.2 Formal Methods and Performance Evaluation

Formal methods have arisen as prominent techniques for the validation of functionality and the evaluation of performance of complex systems. They are constantly promoted by the need to manage and support (with ample confidence) the correct functioning and quality of time critical systems and their supporting components (with ever-growing complexity of software and hardware). Such systems include, e.g., health-care equipment, airplanes, space shuttles, and nuclear power plants, as well as other, less vital, but societally important devices, like mobile phones, Internet protocols, cash machines, etc.

The purpose of formal verification is to show that the model of the system or its conception behaves or will behave correctly according to the specification. For example, the flight management software does not stall

the airplane or the new cash machine will not block the bank card when the correct pin code is supplied. In addition to the correct functional or qualitative behavior, the quantitative behavior plays a crucial role as well. The quantitative analysis or performance evaluation gives an approximation how well the system behaves or will behave. For example, in 95 percent of the cases, the Internet video protocol enables smooth viewing of high-definition movies or the expected utilization of the new jet engine design is 50 percent.

Modeling formalisms come in different flavors. Originally, they modeled only the qualitative behavior of the system, focusing on different aspects of the specification. Here we can mention some of them, e.g., automata, finite-state machines, Petri nets, or process theories. These high-level formalisms produce an explicit (or underlying) model of the system, that we will typically represent as a kind of labeled transition system. A relation, normally an equivalence, is given between transitions systems to identify the ones that are considered to have the same ‘behavior’. This behavioral equivalence is used to check whether the specification and the model of the implementation coincide. The distinguishing power of this relation can range from identifying processes with the same set of traces to mutual simulation of the branching potential. The use of a particular relation depends on the formalism that is used to describe the system, as well as the purpose of the model and the level of abstraction.

Much earlier and in a different community, performance evaluation techniques have been developed in order to assess the performance of a system. These techniques include the study of renewal processes, queueing theory and queueing networks, Markovian and non-Markovian analysis, simulation, etc. The underlying models of the non-simulation techniques are usually types or extensions of Markovian processes. The most prominent are discrete- and continuous-time Markov (reward) chains, Markov decision processes, semi-Markov, and generalized semi-Markov processes. These models can also be represented as transition systems. The behavioral relation between these models is generally given in terms of partitions, called lumpings, or aggregations, which preserve the performance measures of the model.

In this thesis, we mainly restrict our research domain to process theories in the form of process algebras, and, more specifically, ACP-style process algebras. Process algebras provide for an equational characterization (axiomatization) of the behavioral equivalence that is typically required to be a congruence for an interesting set of operators. Besides being compositional, the equational reasoning has an advantage against model checking and theorem proving as it avoids (as much as possible) construction of large state

spaces. The style of the process algebra indicates the way some general features are brought into the theory, like alternative and parallel composition, inclusion of time and probabilities, etc. Following the design rationale of ACP-style process algebras, we define strong bisimulation relations for each new setting and we identify a set of primitive operators that are used to bring more complex features in the theory [18].

Notably, the results from Chapter 7 are applicable to all formalisms that use continuous-time Markov reward chains as underlying performance models. Chapter 8 discusses the modeling language χ , which is a process algebra with data. Moreover, the performance model developed in the same chapter can be derived from any formalism that comprises probabilistic choices and discrete-time delays.

1.3 Process Algebras

Similarly to other modeling formalisms, process algebras were initially developed for qualitative modeling solely, but they gradually have been extended with time, probabilities, Markovian (exponential), and generally-distributed stochastic time. For an overview of the history and crucial milestones in the field of process algebra, we refer to [7, 19, 1]. Usually, qualitative behavior is specified by using action prefixed terms that give the dynamics of the system. The action prefix operators induce labeled transitions in the underlying transition system. The process terms are combined using two basic operators:

1. Alternative composition that provides the alternatives in a given situation, i.e., the outgoing labeled transitions of a state.
2. Parallel composition that enables the compositional modeling by building more complex systems from communicating components. The (synchronous) communication is modeled as synchronization of action prefixes or merging of action transitions.

A typical model of an ACP-style process algebra is the term model, that is obtained as the quotient algebra modulo the behavioral equivalence. Therefore, this equivalence must be also a congruence for the given operations. An equational theory (axiomatization) identifies the equivalent process terms according to the behavioral equivalence. A typical requirement for an equational theory is to be ground-complete, i.e., to identify all equivalent processes that do not contain term variables. Sometimes, ω -complete axiomatizations that include term variables are needed as well.

The definitions of the behavioral relations can be involved, so the axiomatization gives another point of view. Usually, for closely related behavioral relations the equational theories differ only on some axioms, which exactly pinpoint their difference.

An expansion law gives the relation between the two basic composition operators by transforming a parallel composition of two terms into an alternative composition of action prefixed terms. Therefore, the parallel composition is prone to state explosion, as the number of states increases exponentially when it is resolved to explicitly state all alternatives in the transition system. This expansion law plays a central role in process algebras as it provides for a so-called head normal form. Every process in such form is represented as an alternative composition of action prefixed terms in head normal form [18]. The head normal form itself has an important role as it supports many technical results like ground-completeness and uniqueness of solutions of recursive relations [9, 8].

1.4 Timed and Probabilistic Extensions

Timed features were introduced to model time-critical systems, which cannot be modeled realistically without capturing their temporal behavior. The temporal aspects were brought in by conservatively extending some existing standard process theory. The extensions were conservative because they did not introduce any new equalities or behavior when restricted to the untimed part of the theories. For an overview and a generic approach to extensions with time, we refer to [84]. The most prominent timed versions of ACP-style process algebras are given in [11].

Time can be introduced in several manners. The time domain can be discrete or continuous, depending on the support set. Then, the timing itself can be relative, which is typically introduced by timed delay prefixes that give the duration of the timed delay, or absolute, which is incorporated as time-stamped actions. In the setting of this thesis, we will employ discrete relative timing in the form of timed delay prefix operators that induce timed transitions. The behavioral equivalence usually requires that equivalent processes allow passage of time of equal duration.

From a process-theoretical point of view, the identifying features of the timed process theories are time determinism and time additivity. Time determinism states that passage of time does not decide a choice by itself. As a consequence, timed prefixes and timed transitions in the alternative and parallel composition are merged. Time additivity allows subsequent timed delays to be merged together and form an accumulative delay. This

supports the intuition that passage of time does not have an observable role, so timed delays can be “dissected” to suit our needs. Of interest is also the treatment of maximal progress, i.e., the priority of undelayable action transitions that do not allow passage of time over timed transitions.

In ACP-style process algebras, a nondeterministic weak choice in the alternative composition between undelayable actions and passage of time is assumed, similar to the choice between action transitions. The underlying intuition is that future alternatives should not be disabled by default, unless that is what is actually wanted. In the latter case, this is accomplished by a maximal progress operator that disables passage of time in the presence of outgoing prioritized labeled transitions. It is also practice to derive composite notions, instead of introducing them as separate constructs. For example, the delayable action prefix that either delays indefinitely long or performs an undelayable action transition is derived by combining undelayable action and timed delay prefixes. In this way, the manipulation of the higher constructs is supported and justified by the manipulation of the comprising primitive operations. This also validates the design of the primitives, as the intuition on the higher level and the derivation on the lower must match.

We briefly discuss probabilistic extensions. We refer to [59] for an in-depth discussion. Notably, probabilistic extensions are also conservative extensions of existing process algebras. Typically, the probabilistic choice has priority over the alternative composition and it is synchronized in the parallel composition. The behavior equivalence relates processes that have the same accumulative probability of reaching the same partitioning class. The underlying models are probabilistic transition systems in which the next state/transition is determined by a probabilistic distribution. There are also probabilistic extensions of timed process theories, which have discrete-time Markov reward chains as an underlying model [50]. In timed and probabilistic extensions, we study a more natural performance model comprising immediate probabilistic choices and deterministic delays. The performance measures of the model are derived by a translation to a corresponding discrete-time Markov reward chain.

1.5 Markovian Time Extensions

The process theories up to stochastic time usually (conservatively) extended previously well-established theories. However, due to the nature of the underlying (non-)Markovian performance models mostly, the stochastic process algebras were built from scratch. Markovian extensions employ exponential delays that are either coupled with the action prefixes (like in TIPP,

EMPA, PEPA) [52, 20, 55] or orthogonally introduced as separate delays (e.g., IMC) [51]. The appeal of the exponential delay lies in the fact that it is memoryless, i.e., its distribution does not change if the delay is observed after some passage of time during which it did not expire. Moreover, the exponential distribution is the only continuous distribution with this property. The memoryless property and the fact that the minimum of two exponential distribution is an exponential distribution enabled the development of the Markovian performance models.

The same properties also supported the development of Markovian process algebras that aimed for a single specification suitable for both functional verification and performance evaluation. The advantage of such an approach lies in the possibility to develop a common framework for automated validation and performance evaluation based on the concept of model checking. Model checking is the process of certifying whether a model satisfies some logical formula. It is extended to performance evaluation by reusing existing algorithms for computing performance measure of Markovian models and adapting the logical formulae to specify performance-like requirements [14]. As the performance model is derived in a compositional manner from the stochastic process theories, it suffers from the state space explosion problem.

An essential problem arose when actions coupled with exponential delays had to be synchronized. The problem is due to the fact that the maximum of two exponential distributions is not exponentially distributed. Here, we do not enter in a discussion of the proposed solutions and we refer the interested reader to [54]. Notably, the orthogonal extension of IMC that introduces exponential delays as separate constructs circumvents this problem. The parallel composition is resolved by interleaving exponential delays and synchronizing only on delayable action transitions. The underlying performance model is obtained by eliminating the abstracted action transitions using some weak behavioral equivalence, whereas exponential delays are lumped as in Markov reward chains.

We note that the models employed for performance evaluation have their performance measures founded on broadly-accepted notions in probability theory. Although usually represented as transition systems, the performance models are in fact stochastic processes with strong mathematical background. The behavioral equivalence between the transition systems that model the processes comprising Markovian time standardly reduces such graphs to Markovian models. Such reduction supports the intuition to a great extent, but there is no obvious way to show that the original graph (where the notion of performance is at best ambiguous) and the underlying Markovian model have the same performance characteristics.

One approach to showing the correctness of these reductions is to treat the transition systems as generalizations of Markovian processes and, then, show that the reduction methods preserve the performance measures. Remarkably, different formalisms use different methods for determining the performance model, but they should all eventually reduce the original process to the same aggregated version of the model. It is also interesting to see whether the (existing) aggregation methods that induce behavioral relations in the Markovian realm are actually usable. More precisely, the behavioral relation should be an equivalence and preferably a congruence, at least for the parallel composition. Thus a probe into the relational and compositional properties of the preorders induced by the aggregation methods is in place.

From the standpoint of timed process algebra, it can be argued that time determinism can be supported in Markovian time, as the choice is not made by the passage of time per se, but by the probabilistic choice that determines the delay that expires first. However, time additivity cannot be directly supported as the sum of two exponentially distributed random variables is not exponentially distributed. This means that the passage of time of two consecutive exponential delays cannot be represented by a single exponential delay. Moreover, as exponential distributions are continuous, standard deterministic timed delays cannot be ‘mimicked’ by exponential ones, so conservative extensions are not immediate, if possible. Also, although the classes of deterministic and exponential distributions are closed for the minimal sets of operations as discussed above, their combinations are not. For example, the residual distribution of a deterministic distribution after an exponential one expired is neither deterministic nor exponential.

1.6 Extensions with Generally-Distributed Stochastic Time

The need for general distributions arose as exponential delays are not efficient for the modeling of deterministic delays or high-variance heavy-tail distributions, e.g., the fixed timeouts of the Internet protocols or the distributions of the delays in media streaming services. Notably, most (well-behaved) distributions can be approximated by so-called phase-type distributions that can be viewed as absorbing Markov chains comprising exponential delays that replace the original probability distribution. However, discrete or high-variance distributions require substantial effort and space to be satisfactorily estimated [82]. Prominent stochastic process algebras with generally-distributed delays include TIPP, GSMMPA, SPADES, IGSMP, NMSPA, and MODEST [52, 27, 42, 26, 64, 22]. Despite the greater expressiveness, compositional modeling with general distributions proved to be

challenging, as the memoryless property could not be relied on [60, 28]. Other stochastic process algebras that we mention here are the stochastic π -calculus and stochastic LOTOS [87, 4]. More can be found in the review [28].

Usually, the underlying performance model is a generalized semi-Markov process that exploits clocks to memorize past behavior in order to retain the Markov property of history independence [48]. Similarly, the semantics of stochastic process algebras is given using clocks that represent the stochastic delays at a symbolic level. Such a symbolic representation allows for the manipulation of finite structures, e.g., stochastic automata [41] that support SPADES or extensions of generalized semi-Markov processes [26] for IGSM. The concrete execution model is subsequently obtained by sampling the clocks, frequently yielding infinite probabilistic timed transition systems. For the sampling of the clock two execution policies can be adopted:

1. A race condition [52, 42, 64, 22], which enables the action transitions guarded by the clocks that expire first (the execution policy of the Markov chains), and
2. pre-selection policy [27, 26], which preselects the clocks by making a probabilistic choice (the execution policy of generalized semi-Markov processes).

Notably, more execution policies have been developed for stochastic and generalized stochastic Petri nets, comprising exponential delays and immediate probabilistic choices. There, multiple transitions can be enabled and taken at the same time, leading to more complicated runs.

In absence of the memoryless property, the samples of the clocks must be updated after each stochastic delay transition. This is because the residual sample/distribution of the clock depends on the duration of time that the clock has been active. Again, the literature provides two techniques for doing this:

1. keeping track of the *residual lifetime* of clocks, i.e., the time that is left before the clock expires; or
2. keeping track of the *spent lifetime* of clocks, i.e., the time that the clock has been active.

The residual lifetime semantics [42], depicted in Figure 1.9a, supports performance analysis via discrete event simulation, that is extensively exploited when analytical methods do not apply. However, it has been criticized for its

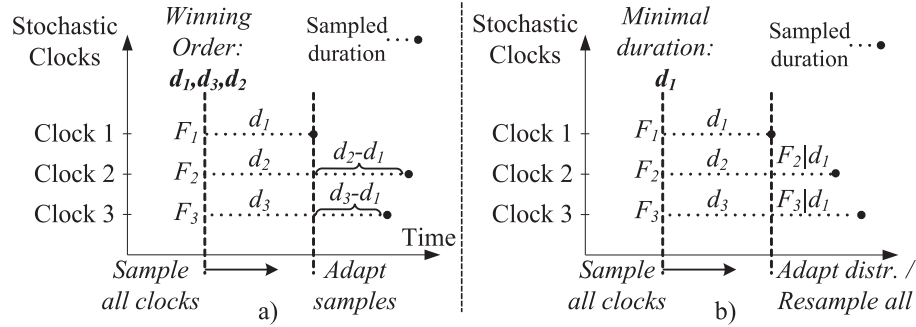


Figure 1.9: a) Residual lifetime semantics with clocks and b) spent lifetime semantics with clocks. The notation $F_2|d_1$ and $F_3|d_1$ denotes that the distributions F_2 and F_3 of the clocks C_2 and C_3 , respectively, have been shifted to the right by the duration d_1 .

being unfair as the outcome of the race condition is known upfront due to the early sampling of clocks. The spent lifetime semantics [52, 27, 26, 64], depicted in Figure 1.9b, has been advocated for its correspondence to standard real-time, as the clocks increase as time passes. Additionally, the approach is considered fair with respect to the race condition as the clocks are first pre-sampled to statistically determine the minimal sample. Afterwards, the original samples are discarded, while the probability distributions of the remaining clocks are ‘aged’ with the minimal sample. However, the fairness comes at a price: re-sampling of the clocks is required after each resolution of the race condition.

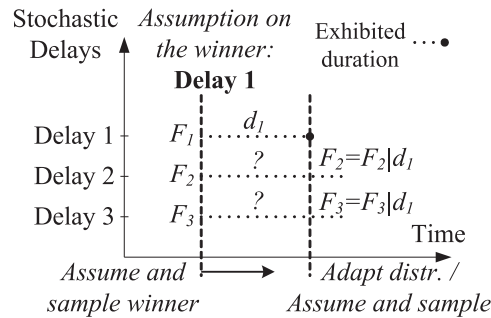


Figure 1.10: Spent lifetime semantics with stochastic delays. The notation $F_2|d_1$ and $F_3|d_1$ denotes that the distributions F_2 and F_3 of the stochastic delays 2 and 3, respectively, have been shifted to the right by d_1 .

An alternative, but equivalent approach to the race condition is to make a probabilistic assumption on the outcome of the race condition by conditioning the clocks that win the race, and, afterward, to sample from the (joint) probability distribution of the winning clocks [57]. See Figure 1.10. In this approach each clock is sampled only once. So, there is no need to keep track of the lifetimes of clocks. Instead, distributions have an ‘age’ which takes account for time exhibited by previous samples. We refer to the samples as *stochastic delays*, resembling the notion of timed delays. In the present setting, we employ the race condition with spent-lifetime semantics. We rely on its interpretation in terms of conditional random variables, which makes a probabilistic assumption on the winning stochastic delays, i.e., the ones that expire. This is followed by conditioning the distributions of the losing delays, i.e., the ones that do not expire, on the time spent for the winning samples [57].

Of interest is the interplay between real and stochastic time that coexist in the same process theory. We investigate the possibility and means to (conservatively) extend real-time process theories with stochastic time. We also look into the possibility of extending timed delays with probabilistic features that might enable the derivation of stochastic delays, similar to delayable actions. We opt for discrete time as continuous distributions cannot be restricted to mimic standard real-time delays. Moreover, the extension with discrete stochastic time is more complicated (if the probability distributions of the stochastic delays are measurable [35]) as there is non-zero probability that several delays expire simultaneously. We also look into the problem of embedding real time in a stochastic-time setting. Finally, we consider the replacement of timed delays with stochastic ones and we examine closer the effect of such an experiment.

The relation between real and stochastic time has already been studied in various settings. Due to the nature of the stochastic process theories the results show an embedding or translations to a purely timed formalism. A structural translation from stochastic automata to timed automata with deadlines that preserves the timed traces is given in [40]. This approach found its way into MODEST [22] as a means to introduce real and stochastic time as separate constructs in the same formalism. Also, a translation from IGSMP into pure real-time models called interactive timed automata is reported in [26]. In [4] a proposal of extending timed LOTOS is made by exploiting stochastic timers.

1.7 Outline and Contribution

In the course of this thesis we investigate several issues concerning real and stochastic time and their interaction in process algebras for performance evaluation:

- What is the relationship between discrete real and generally-distributed stochastic time in process theories?
- Is it possible, and if so, how, to (conservatively) extend timed process theories with stochastic time?
- Reversely, is it possible, and if so, how, to embed (discrete) real time in generally-distributed process theories?
- What is the effect of replacing timed delays by stochastic ones and what are the consequences of such a generalization?
- Is it possible to show that the abstraction using the weak behavioral equivalence in Markovian process theories (and other modeling formalisms) is performance preserving. Moreover, is such an approach compositional?
- Can we do performance analysis using discrete-time delays and probabilistic choices?

To tackle these issues, first we develop a ground-complete process theory that accommodates timed delays in racing contexts. These timed delays model the expiration of stochastic delays in race condition semantics per time unit. Basically, they dissect the execution of the race per unit time step, symbolically representing the choice whether the stochastic delay expires in one time unit or not. Different from other approaches, instead of introducing both timed and stochastic delays as separate constructs, we derive stochastic delays as time-delayed processes in a racing context. The relationship between the expiring stochastic delays and the ones that have to be aged is made explicit, which allows for an explicit handling of the race condition.

The theory provides a non-trivial expansion law for the parallel composition, as well as an explicit treatment of the maximal progress operator. It also enables the possibility of specifying a partial race of stochastic delays, e.g., that one stochastic delay always has a shorter sample than another one. This feature facilitates the modeling of timed systems whose correct behavior depends on the ordering of the durations of the timed delays, e.g.,

in a time dependent controller. It also supports the replacement of timed delays by stochastic ones. In that case, the total order of the samples is, in general, lost, unless it is possible to specify which delays are the ones that expired first and which are made dependent in the imposed race.

Then, we isolate an independent part of the theory that comprises undelayable and delayable action prefixes, and stochastic delays. We show that even though the delayable action and stochastic delay prefix are derived notions, their interaction can be handled without resorting to the defining timed specifications. Still, to justify the derived equational theory we analyze the representations in terms of equations involving timed delays in racing contexts.

Afterwards, we take exactly the opposite approach by treating real time from a stochastic viewpoint and we reveal the other side of the same coin. Here, we treat timed and stochastic delays as ‘atomic’, rather than series of timed delays in racing contexts. This puts the timed delays on the same level with the stochastic ones as passage of time is studied in terms of discrete events, where the actual duration/sample of the delay plays more of a background role. The race condition remains the central notion in both settings. We investigate what needs to be in place to generalize timed delays to stochastic ones. Therefore, we analyze stochastic bisimulation as well as the fit of identifying real-time features, like time determinism and time additivity, in a stochastic-time setting. This brings us to the notion of *context-sensitive interpolation*, which can be viewed as an interpretation of the race condition in the timed setting. We benefit from our findings in the development of a stochastic process algebra that retains many features of the timed process theories, but permits a restricted form of time additivity only.

We illustrate the developed theories by specifying $G/G/1/\infty$ queue and solving its recursive specification. We also specify a variant of the concurrent alternating bit protocol that has fixed time-outs (represented by timed delays) and lossy channels (with discrete generally-distributed delays), stressing the interplay of real and stochastic time. It is well known that only a small, restricted classes of models comprising generally-distributed delays are analytically solvable. Preliminary research on model checking of stochastic automata is reported in [29] and a proposal for model checking probabilistic timed systems is given in [90]. However, at the moment, the performance analysts turn to simulation when it comes to analyzing models with generally distributed delays.

For the purpose of analyzing the specification of the concurrent alternating bit protocol we depend on the toolset of the χ -language [6, 25]. At

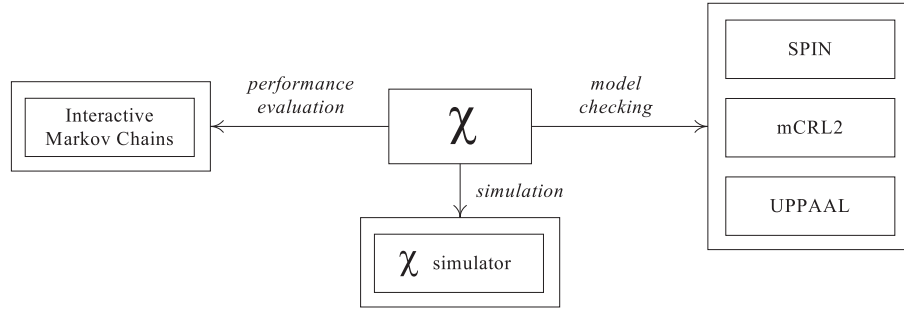


Figure 1.11: The framework of the language timed χ

the start, χ was used to model discrete-event systems only, not supported by an explicit semantics. Later, it was turned into a formal specification language set up as a timed process algebra with data [25]. More recently, the χ language was extended with differential algebraic equations, leading to hybrid χ [88]. The framework of timed χ is depicted in Figure 1.11. Specifications in χ can be compiled as an input language to several model checkers for validation purposes. Performance evaluation is done either by Markovian analysis (by translating the model to an interactive Markov chain [51]) or by discrete-event simulation. We augment the χ -toolset with a prototype extension to support performance evaluation of probabilistic timed specifications as well. The protocol case study illustrates the new approach when the channel distributions are deterministic.

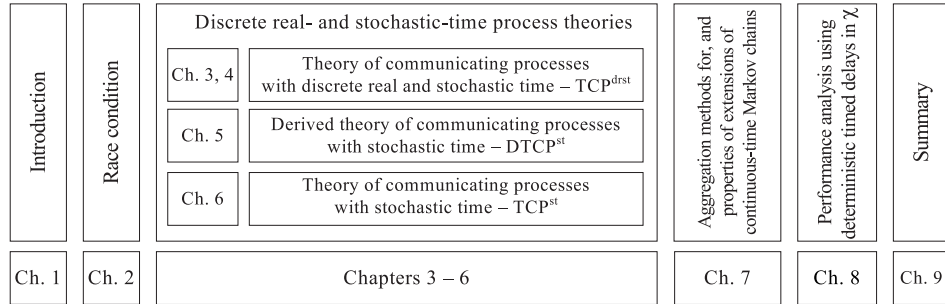


Figure 1.12: Outline of the thesis

The outline of the thesis can be visualized as in Figure 1.12. Chapter 2 deals with the central concept of the race condition along the lines sketched above and using a representation in terms of conditional random variables. It gives the base for the development of a process theory that comprises timed

delays in a racing context in Chapter 3. We proceed with the equational theory in Chapter 4. Next, in Chapter 5, we introduce delayable actions and stochastic delays in the theory as derived notions and we show they can be manipulated without resorting to the primitives that comprise them. Afterwards, in Chapter 6, we approach the issues from a different perspective. We develop a stochastic process theory from scratch and attempt to fit discrete time by both extending and embedding it into the theory. Chapter 7 studies the derivation of Markov reward chains from modeling formalisms and their relational and compositional properties. Chapter 8 illustrates the developed theory by analyzing the concurrent alternating bit protocol with deterministic, Markovian, and generally-distributed lossy channels. Here, we also develop a performance model for systems comprising immediate probabilistic choices and deterministic delays. We conclude the thesis with a summary in Chapter 9.

This thesis is based on the following publications and submitted manuscripts:

- J. Markovski and E.P. de Vink: “Embedding Real Time in Stochastic Process Algebra” [66]. In a longer version the paper also appeared as [67]. It gives a preliminary account of stochastic process algebras that embed real-time delays. It introduces the notion of environments as constructs that keep track of the age of the distributions and gives semantics of stochastic delay prefixed terms. The ideas underlying Chapters 6 originate from this work.
- J. Markovski and E.P. de Vink: “Real-Time Process Algebra with Stochastic Delays” [69]. The paper introduces two types of race conditions that enable a non-trivial expansion law for stochastic delay prefixed terms. It provides the groundwork for Chapters 2 and 3.
- J. Markovski and E.P. de Vink: “Real-Time in Stochastic Process Algebra: Keeping Track of Winners and Losers” [68]. This technical report introduces the splitting of a race on disjoint events by explicitly stating the relationship between expired and aged stochastic delays. The technical results are adapted in Chapters 3 and 6.
- J. Markovski and E.P. de Vink: “Discrete Real-Time and Stochastic-Time Process Algebra for Performance Analysis of Distributed Systems” [71]. In a longer version the paper also appeared as a part of [70]. The paper is the base for Chapters 3, 4, and 5 as it discusses the semantics and presents a ground-complete equational theory for

race-complete specifications. It also presents a method for performing transient analysis of the discrete-time probabilistic reward graphs discussed in Chapter 8.

- J. Markovski and E.P. de Vink: “Extending Timed Process Algebra with Discrete Stochastic Time” [72]. In a longer version the paper also appeared as a part of [70]. This paper analyzes the effect of replacing real-time delays by stochastic-time delays. To support the “stochastifying” of real-time delays it discusses the race condition semantics from the viewpoint of real-time process theories. It proposes a restricted notion of time-additivity, referred to as context-sensitive interpolation, that conforms to the race condition. It is incorporated in the thesis as a part of Chapter 6.
- J. Markovski and N. Trčka: “Lumping Markov Chains with Silent Steps” [75]. In a longer version the paper also appeared as part of [76]. This paper paves the way of looking at intermediate performance models containing nondeterministic silent steps as stochastic processes. It defines a lumping method for such models and presents the initial idea underlying Chapter 7.
- J. Markovski and N. Trčka: “Aggregation Methods for Markov Reward Chains with Fast and Silent Transitions” [78]. In a longer version the paper also appeared as part of [77]. This paper makes a comparative analysis of lumping- and reduction-based aggregation methods for extensions of Markov reward chains with immediate probabilistic and nondeterministic transitions. It gives the base of Chapter 7.
- J. Markovski, A. Sokolova, N. Trčka, and E.P. de Vink: “Compositionality for Markov Reward Processes with Fast Transitions” [74]. In a longer version the paper also appeared as [73]. A version generalized with nondeterministic silent transitions has been submitted to a special issue of the journal Performance Evaluation. This paper studies the relational and composition properties of aggregation methods based on lumping and reduction. It gives the base of Section 7.3.
- N. Trčka, S. Georgievska, J. Markovski, S. Andova, and E.P. de Vink: “Performance Analysis of Chi Models using Discrete-Time Probabilistic Reward Graphs” [95]. In a longer version the paper also appeared as [96]. It shows the extension of the framework of χ with discrete-time probabilistic reward graphs and we build on it to obtain the theoretical and empirical results in Chapter 8.

Chapter 2

Race Condition

In this chapter we provide the mathematical background and we postulate the central concepts of race condition, racing context, timed, and stochastic delays. We define two types of race conditions to accommodate for compositional modeling as well as manipulation of stochastic delays in expansion laws. One condition treats delays as having independent samples, whereas the other synchronizes on delays with the same name.

We use discrete random variables to represent durations of stochastic delays. The set of discrete distribution functions F such that $F(n) = 0$ for $n \leq 0$ is denoted by \mathcal{F} ; the set of the corresponding random variables by \mathcal{V} . We use X, Y , and Z to range over \mathcal{V} and F_X, F_Y , and F_Z for their respective distribution functions. Also, W, L, V , and D range over $2^{\mathcal{V}}$. By assumption, the support set $\text{supp}(X) = \{n > 0 \mid P(X = n) > 0\}$ of a random variable X is finite or countably infinite. The domain A of a function $f: A \rightarrow B$ is denoted by $\text{dom}(f)$. In case f is bijective, we write $f: A \leftrightarrow B$. The identity bijection on the set A is denoted by id_A . We write $p \subseteq A$ for a predicate $p: A \rightarrow \{\top, \perp\}$, where \top and \perp denote the truth values true and false, respectively. Composition of two relations $r_1 \subseteq A \times B$ and $r_2 \subseteq B \times C$ is given by $r_2 \circ r_1 \subseteq A \times C$ where $(x, z) \in r_2 \circ r_1$ if there exists a $y \in B$ such that $(x, y) \in r_1$ and $(y, z) \in r_2$. We restrict and rename functions on disjoint parts of the domain by $g\{f_1/D_1\} \dots \{f_n/D_n\}(x) = f_i(x)$ if $x \in D_i$, and $g(x)$ if $x \in D \setminus (\bigcup_{i=1}^n D_i)$, for functions $g, f_1, \dots, f_n: A \rightarrow B$ and disjoint subsets $D_1, \dots, D_n \subseteq A$. By $\mathcal{P}(A)$ we denote the set of standard discrete probabilistic spaces (A, P) over the set A with probability measure P .

2.1 Racing Stochastic Delays

A stochastic delay is a timed delay of a duration that is guided by a random variable. We use the random variable as the *name* of the stochastic delay. We observe simultaneous passage of time for a number of stochastic delays until one or some of them expire. This phenomenon is referred to as the *race condition* and the underlying process as the *race*. For multiple racing stochastic delays, different stochastic delays may be observed simultaneously as being the shortest. The ones that have the shortest duration are called the *winners*, the others are referred to as the *losers*. We illustrate the concepts by an example.

Example 2.1.1 Let X and Y be random variables with $P(X = 1) = P(X = 2) = P(X = 3) = \frac{1}{3}$ and $P(Y = 2) = \frac{1}{2}$, $P(Y = 3) = P(Y = 4) = \frac{1}{4}$. Now, let us assume that two delays X and Y are guided by the variables with the same name. The probability that X wins the race is the probability $P(X < Y) = \frac{1}{3} \cdot (\frac{1}{2} + \frac{1}{4} + \frac{1}{4}) + \frac{1}{3} \cdot (\frac{1}{4} + \frac{1}{4}) + \frac{1}{3} \cdot \frac{1}{4} = \frac{7}{12}$. Then, the winning delay is distributed as $W_X = \langle X \mid X < Y \rangle$ with $P(W_X = 1) = P\langle X = 1 \mid X < Y \rangle = \frac{P(X=1, X<Y)}{P(X<Y)} = \frac{\frac{1}{3}}{\frac{7}{12}} = \frac{4}{7}$, $P(W_X = 2) = \frac{2}{7}$, and $P(W_X = 3) = \frac{1}{7}$. Similarly, the probability that Y wins the race is the probability $P(Y < X) = \frac{2}{12}$. Then, the winning delay is distributed as $W_Y = \langle Y \mid Y < X \rangle$ with $P(W_Y = 2) = 1$. Both, X and Y win the race together with probability $P(X = Y) = \frac{3}{12}$ and a winning delay distributed as $W_{XY} = \langle X \mid X = Y \rangle$ (or, the equivalent, $\langle Y \mid X = Y \rangle$) with $P(W_{XY} = 2) = \frac{2}{3}$ and $P(W_{XY} = 3) = \frac{1}{3}$. \square

An outcome of a race is completely determined by the winners and the losers. So, we can explicitly represent the outcome of the race by a pair of sets of stochastic delays $[\frac{W}{L}]$, where W is the set of winners and L is the set of losers. We write $[W]$ instead of $[\frac{W}{\emptyset}]$ and omit the set brackets when clear from the context. Thus, $[X]$ represents a stochastic delay with name X , guided by the random variable X .

Outcomes of races may be involved in other races, so we refer to an outcome $[\frac{W}{L}]$ as a (conditional) *stochastic delay* induced by the disjoint sets of winners W and losers L . By $W < L$ we denote the event

$$W < L \text{ iff } X_1 = X_2 \text{ for } X_1, X_2 \in W \text{ and } X_3 < Y \text{ for } X_3 \in W, Y \in L$$

and by $W < n$ for $n \in \mathbb{N}$ we denote

$$W < n \text{ iff } X < n \text{ for } X \in W.$$

Similarly, we also use $W = n$.

Now, the probability of the outcome $[L^W]$ is $P(W < L)$ and the stochastic delay is guided by the conditional random variable $\langle X \mid W < L \rangle$ for any $X \in W$. Two stochastic delays $[L_1^{W_1}]$ and $[L_2^{W_2}]$ can race against each other and they can form a joint outcome if it is possible to consistently combine the winners and the losers such that the resulting outcome has disjoint winners and losers. Here, by consistently we mean that in the joint outcome no winner can come from the original sets of losers L_1 or L_2 .

We take a closer look at the relation between the winners and the losers of the racing delays $[L_1^{W_1}]$ and $[L_2^{W_2}]$. There are three possible combinations that give the relation between the winners and the losers: (1) $L_1 \cap W_2 \neq \emptyset$, which means that the race must be won by W_1 and lost by $L_1 \cup W_2 \cup L_2$, (2) $W_1 \cap W_2 \neq \emptyset$, which means that the race must be won by $W_1 \cup W_2$ together and lost by $L_1 \cup L_2$, and (3) $W_1 \cap L_2 \neq \emptyset$, which means that the race must be won W_2 and lost by $W_1 \cup L_1 \cup L_2$. Obviously, these ‘restrictions’ are disjoint and cannot be applied together. If more than one holds, then they lead to ill-defined outcomes. For example, if both (1) and (2) hold at the same time, then L_1 and W_2 must exhibit the same sample and also W_1 and W_2 must exhibit the same sample. Then W_1 and L_1 must exhibit the same sample, which is a contradiction.

To summarize, there are four possible joint outcomes of a race between $[L_1^{W_1}]$ and $[L_2^{W_2}]$: if (1) holds then the outcome is given by $[L_1 \cup W_2 \cup L_2^{W_1}]$, if (2) holds the outcome is given by $[L_1 \cup L_2^{W_1 \cup W_2}]$, if (3) holds then the outcome is given by $[L_1 \cup L_2^{W_2}]$ and if none of the restrictions (1)–(3) hold, then all three (disjoint) outcomes are possible: $[L_1 \cup W_2 \cup L_2^{W_1}]$, $[L_1 \cup L_2^{W_1 \cup W_2}]$, and $[L_1 \cup L_2^{W_2}]$. If at least two restrictions apply, then the outcomes cannot be combined as they represent disjoint events. In this case we say the race between the delays $[L_1^{W_1}]$ and $[L_2^{W_2}]$ with $W_1 \cup L_1 = W_2 \cup L_2$, is *resolved*. The extra condition ensures that the outcomes stem from the same race, i.e, they have the same racing delays. For example, $[Y^X]$ and $[X^Z]$ cannot form a joint outcome. The delays do not stem from the same race, which renders their combination inconsistent.

Resolved races play an important role as they enumerate every possible outcome of the race. We define a predicate $\text{rr}([L_1^{W_1}], [L_2^{W_2}])$ that checks whether two delays $[L_1^{W_1}]$ and $[L_2^{W_2}]$ are in a resolved race. It is satisfied if $W_1 \cup L_1 =$

$W_2 \cup L_2$ and at least two of the above three restrictions hold, i.e.,

$$\text{rr}([L_1^{W_1}], [L_2^{W_2}]) \text{ if } W_1 \cup L_1 = W_2 \cup L_2 \text{ and} \\ ((L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap W_2 \neq \emptyset) \text{ or} \\ (L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset) \text{ or} \\ (W_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset)).$$

We proceed by introducing processes that are prefixed by stochastic delays.

2.2 Stochastic Delay Prefix

By $[L^W].p$ we denote a process term p prefixed by a stochastic delay $[L^W]$. This prefixed term denotes a process that behaves as p after $[L^W]$ expires. To express a race, we use the alternative composition $_ + _$. So, $[X].p_1 + [Y].p_2$ represents two processes that are prefixed by the stochastic delays X and Y that are racing against each other. As discussed above, there are three possible outcomes of this race in terms of the participating stochastic delays: (1) $[X]$, (2) $[X_\emptyset^Y]$, and (3) $[Y]$, i.e., the first stochastic delay expires before the second, they both expire together, or the second stochastic delay expires before the first. The passage of time of the stochastic delay $[X]$ is guided by the conditional random variable $\langle X \mid X < Y \rangle$. In this case, the stochastic delay X expires, whereas Y becomes dependent on the amount of time that has passed for X . Intuitively, this is represented by the term $[X].(p_1 + [Y].p_2)$, where both occurrences of Y refer to the same stochastic delay, i.e., the second occurrence of Y is bound by the first one. Similarly, we have $[Y].([X].p_1 + p_2)$, when the winner is Y . In the case when both delays win, they expire together. By the notion of time determinism, which states that passage of time by itself cannot make a choice, the resulting term should be $[X_\emptyset^Y].(p_1 + p_2)$.

The race is resolved when every possible outcome of the race is enumerated, i.e., no more outcomes are possible. Thus, we can also write $[X].(p_1 + [Y].p_2) + [X_\emptyset^Y].(p_1 + p_2) + [Y].([X].p_1 + p_2)$ instead of $[X].p_1 + [Y].p_2$ as both expressions have the same final outcomes of a race. The advantage of the first term is that it explicitly states all possible outcomes of the race and that these events are disjoint. Thus, we can clearly separate the disjoint stochastic behavior of the term depending on the resolved outcomes of the race. If an additional racing delay Z is added to the race, this also leads to the same outcomes, i.e., $([X] + [Y]) + [Z]$ and $([X_\emptyset^Y] + [Y]) + [Z]$ will yield the same racing behaviour. As an example, the outcome of $[X_\emptyset^Y] + [Z]$ is given by $[X_\emptyset^Z] + [Y^X]$. When considering complete races, i.e., race

which have all possible outcomes, such an alternative composition is associative (cf. [68]). However, when considering incomplete races, e.g., the race induced by the term $[\overset{X}{Y}].p_1 + [\overset{X}{X}].p_2$, the alternative composition is no longer associative as discussed below in Section 4.3.

Next, we motivate the need and introduce an additional type of a race condition.

2.3 Dependent and Independent Race Condition

We give a motivation and illustrate the notions of a dependent and an independent race condition by a simple example. Consider the term $[X].p \parallel [X].p$, where \parallel denotes parallel composition. The semantics of the race condition in the parallel composition is the same as for the alternative composition. We can interpret the race between the two processes above in two ways: (1) from the standard viewpoint of Markovian/race condition semantics, the process is a composition of two independent components that are competing for the same resource and (2) from real-time perspective this composition synchronizes the two components that exhibit the same sample as they have the same name. The former interpretation is according to the *independent* (standard) race condition and it enables compositional modeling. It states that stochastic delays with the same name have the same distribution, but do not necessarily exhibit the same sample. The latter interpretation is according to the *dependent* race condition that forces racing delays with the same name to always exhibit the same duration. It supports the existence of expansion laws and it enables resolution of races. We give an example to illustrate the situation by interpreting a simple race in both ways.

Example 2.3.1 The term $[\overset{X}{Y}].p_1 + [\overset{X}{Z}].p_2$ should be equivalent to the term $[\overset{X}{Y,Z}].(p_1 + p_2)$ if X is treated as a dependent stochastic delay. Both stochastic delays have a winner guided by X , which exhibits the same sample in both terms and, therefore, the winners of both delays must exhibit passage of time together. On the other hand, if X is treated as an independent stochastic delay, then the same term is equivalent to $[\overset{X}{Y,Z,U}].(p_1 + [\overset{U}{Z}].p_2) + [\overset{X,U}{Y,Z}].(p_1 + p_2) + [\overset{U}{X,Y,Z}].([\overset{X}{Y}].p_1 + p_2)$ for a random variable U satisfying $F_U = F_X$. In the standard independent race condition interpretation, the two occurrences of X can exhibit different samples that are guided by the same distribution. Therefore, they actually represent separate stochastic delays and the second occurrence of X is renamed to a new stochastic delay U with the same distribution. □

We introduce a dependence scope operator $|p|_D$ for $D \subseteq \mathcal{V}$ to specify dependent and independent delays. The racing delays in the races induced by the term p that are in D are treated as dependent. The names of dependent delays are important as they identify stochastic delays that exhibit the same sample. On the contrary, the names of the independent delays play no role except for identifying stochastic delays with the same distribution. In the previous example, $[[\overset{X}{Y}].p_2]_X$ would denote that X is a dependent stochastic delay, but Y is an independent one. Intuitively, this term is equivalent to $[[\overset{Z}{Z}].p_2]_X$, for every Z such that $F_Z = F_Y$, but it is not equivalent to $[[\overset{U}{Y}].p_2]_U$ for any $U \neq X$, even if $F_U = F_X$. Multiple scopes intersect, i.e., $[[p]_{D_2}]_{D_1}$ is equivalent to $|p|_{D_1 \cap D_2}$. For example, $[[\overset{X}{Y}].p]_X|_Y$ denotes a process prefixed by the independent delay $[[\overset{X}{Y}].p]_{\emptyset}$ because $\{X\} \cap \{Y\} = \emptyset$.

The dependence scope plays an important role in giving operational semantics to the terms. Recall, the stochastic delay prefix $[\overset{W}{L}].p$ denotes an outcome of a race between the stochastic delays in $W \cup L$, where the winners are given by W and the losers are given by L . Moreover, it denotes that there was passage of time for the losing delays in L that may continue to persist in p . This means that the losers do not have their original distribution in the resulting process p and that their distributions must be ‘aged’ by the duration of the sample exhibited by the winners W . Therefore, the names of the losing delays must be protected in p , i.e., they become dependent. This is achieved by writing $|p|_L$ as the remaining term after the expiration of the delay given by $[\overset{W}{L}]$. Thus, $[\overset{W}{L}].p$ is actually equivalent to $[\overset{W}{L}].|p|_L$ as only the names in L must be preserved in p . This also means that the stochastic delays that are not in L become independent. To support the interpretation of process terms as discussed above, the stochastic delays that are not encompassed by any dependence scope are considered as dependent. Thus, $[\overset{W}{L}].p$ is equivalent to $[[\overset{W}{L}].p]_{W \cup L}$. We illustrate the above discussion by an example.

Example 2.3.2 The first occurrences of X and Y in the term $[\overset{X}{Y}].[X, Y].p$, denote dependent stochastic delays $[X]$ and $[Y]$. However, the second occurrence of X in the subterm $[X, Y].p$, which by the discussion above is equivalent to $[[X, Y].p]_Y$, denotes an independent stochastic delay, whereas the second occurrence of Y in the same subterm refers to the losing dependent delay $[Y]$ from the first race. \square

Next, we analyze the expiration of a stochastic delay per unit of time, which leads us to the notion of a timed delay in a racing context.

2.4 Timed Delays in a Racing Context

Before introducing timed delays in the process theory, we give a simple example of an expiration of a stochastic delay over a period of time.

Example 2.4.1 Suppose that X is a random variable such that $P(X = 1) = \frac{1}{2}$, $P(X = 2) = \frac{1}{3}$, and $P(X = 3) = \frac{1}{6}$. We observe what happens to the stochastic delay $[X]$ after 1 unit of time. Then, either the stochastic delay expires with probability $\frac{1}{2}$ or it is aged by one time unit, i.e., its distribution is shifted to the right by 1. In the latter case the aged stochastic delay $[X]$ allows a passage of time according to the random variable X' where $X' = \langle X - 1 \mid X > 1 \rangle$ with $P(X' = 1) = \frac{2}{3}$ and $P(X' = 2) = \frac{1}{3}$.

Now, we observe what happens to the delay $[X']$ after one unit of time. The delay $[X']$ expires with probability that $[X]$ did not expire after one time unit multiplied by the probability that $X' = 1$, i.e., $P(X > 1) \cdot P(X' = 1) = \frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}$. Note that the probability of expiration of $[X']$ in one time unit is the same as the probability of expiration of $[X]$ in two time units. However, $[X']$ can also delay more than one time unit and become aged by 1. Then, it allows passage of time according to X'' where $X'' = \langle X' - 1 \mid X' > 1 \rangle$, with $P(X'' = 1) = 1$.

Obviously, $[X'']$ must expire in one time unit and it does so with probability that both $[X]$ and $[X']$ did not expire in one time unit, i.e., $P(X > 1) \cdot P(X' > 1) \cdot P(X'' = 1) = \frac{1}{2} \cdot \frac{1}{3} \cdot 1 = \frac{1}{6}$. Again, the expiration of $[X'']$ in one time unit is equivalent to expiration of $[X']$ in two time units or to the expiration of $[X]$ in three time units. \square

Although being a simple exercise in probability, Example 2.4.1 illustrates how to handle an expiration of a stochastic delay per unit of time. It shows that the distribution of the expiring delay does not have to be re-adapted each time, but it is sufficient to remember its age. First, we formalize the notion of the aging of a distribution, which gives the right shift of a distribution over passage of time.

Definition 2.4.2 A distribution function F can be aged by $m \in \mathbb{N}$ if $F(m) < 1$. The resulting distribution $F|m$ is given by

$$(F|m)(n) = \frac{F(n+m) - F(m)}{1 - F(m)}. \quad \square$$

If the condition of Definition 2.4.2 is fulfilled, then $F|m$ is again a probability distribution function. Because we work with probability distributions satisfying $F(0) = 0$, we have that $F|0 = F$. Moreover, iterative application of the

aging function is the same as aging the function once by the accumulative time duration as illustrated by Example 2.4.1 [66]. This is stated by the following lemma.

Lemma 2.4.3 *If $(\dots(F|d_1)\dots)|d_n$ is defined for $d_1, \dots, d_n \in \mathbb{N}$ and $n \in \mathbb{N}$, then*

$$(\dots(F|d_1)\dots)|d_n = F \mid \left(\sum_{i=1}^n d_i \right). \quad \square$$

PROOF By induction on the number of applications of $|$. The case when $n = 1$ is trivial. Assume that the proposition holds for $n = k$, $k \in \mathbb{N}$. We denote by S the sum $S = \sum_{i=1}^n d_i$. We prove that the proposition holds for $k + 1$ applications of $|$. One obtains the following derivation:

$$\begin{aligned} (\dots(F|d_1)\dots|d_k)|d &= (F|S)|d \\ &= \frac{(F|S)(t+d) - (F|S)(d)}{1 - (F|S)(d)} \\ &= \frac{\frac{F(t+S+d) - F(S)}{1 - F(S)} - \frac{F(S+d) - F(S)}{1 - F(S)}}{1 - \frac{F(S+d) - F(S)}{1 - F(S)}} \\ &= \frac{F(t + (S+d)) - F(S+d)}{1 - F(S+d)} \\ &= F|(S+d), \end{aligned}$$

which completes the proof. ■

As a direct consequence, to compute a total age of a distribution of a stochastic delay it suffices only to compute the sum of the duration of the samples of every race that it lost.

Now, let us denote by σ_\emptyset^x the event that the delay $[X]$ expires after one time unit has passed, i.e., in race condition terminology the stochastic delay $[X]$ wins a race with a sample of one unit timed delay and there are no losers. Let us assume that the age of X is m and let us denote by $X|m = \langle X - m \mid X > m \rangle$ the conditional random variable with distribution $F_X|m$. Then, the probability of the event σ_\emptyset^x is $P((X|m) = 1)$, i.e., the probability that $[X]$ expired after $m + 1$ unit of time. By σ_x^\emptyset , we denote the event that the delay $[X]$ does not expire in one time unit, i.e., the stochastic delay $[X]$ loses the race to a unit timed delay and there are no additional winners. Again, by assuming that X has age m , the probability of this

event is $P((X|m) > 1)$, and after the expiration of the timed delay, the age of X becomes $m + 1$. Thus, at each point in time we have two possibilities: either the delay expires, or the delay does not expire and it is aged by one time unit. Then, the process $[X].p$ can be specified as the solution of the recursive equation

$$A = \sigma_{\emptyset}^x.p + \sigma_x^{\emptyset}.A,$$

for the recursion variable A .

In a generalized context, by the same reasoning, we specify a stochastic delay $[L^W].p$ as the solution of the recursive equation for B :

$$B = \sigma_L^W.p + \sigma_{W \cup L}^{\emptyset}.B,$$

where either the set of winners expire after a unit time step and the losers are aged by one time unit or all racing delays are aged one time unit. We will refer to σ_L^W as a unit timed delay prefix in a racing context of the race induced by the winner W and the losers L , or simply timed delay prefix for short. The probability of this event is denoted by

$$\text{RC}_1(W, L) = P(W = 1, L > 1),$$

where the racing delays in $W \cup L$ can have their own ages as in the discussion for a race with a single delay $[X]$ above.

We emphasize that timed delays are not stochastic delays that impose a race condition and form joint outcomes to resolving it, but they allow passage of one time unit in presupposed racing contexts that can be consistently merged as shown below. In our setting we build a process theory for timed delays in a racing context and retrieve stochastic delays via guarded recursive specifications as indicated above. The standard unit timed delay prefix is embedded in the theory as $\sigma_{\emptyset}^{\emptyset}$, i.e., a timed delay in an empty racing context. By convention we put the probability $\text{RC}_1(\emptyset, \emptyset) = 1$. We omit the empty sets from the notation when clear from the context and we also write σ^n for $n \geq 1$ subsequent timed delays prefixes σ_{\dots} .

Timed delays can also be in a context of resolved races. If $\text{rr}([L_1^{W_1}], [L_2^{W_2}])$ holds, then $\sigma_{L_1}^{W_1}$ and $\sigma_{L_2}^{W_2}$ are in the context of the resolved race between $[L_1^{W_1}]$ and $[L_2^{W_2}]$. However, this does not cover the case when there are no winners in the racing context, i.e., no stochastic delays expire after one unit time step. For that purpose we overload the resolved race predicate $\text{rr}(_)$ to $\text{rr}(\sigma_{L_1}^{W_1}, \sigma_{L_2}^{W_2})$

as follows:

$$\text{rr}(\sigma_{L_1}^{W_1}, \sigma_{L_2}^{W_2}) \text{ if } W_1 \cup L_1 = W_2 \cup L_2 \text{ and}$$

$$\begin{aligned} & ((L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap W_2 \neq \emptyset) \text{ or} \\ & (L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset) \text{ or} \\ & (W_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset) \text{ or} \\ & (W_1 = \emptyset \text{ and } W_2 \cap L_1 \neq \emptyset) \text{ or} \\ & (W_2 = \emptyset \text{ and } W_1 \cap L_2 \neq \emptyset)). \end{aligned}$$

Recall that the predicate $\text{rr}(\cdot)$ defines the context in which the race between the stochastic delays $[L_1]$ and $[L_2]$ is resolved. The extra conditions deal with the overloaded situation for the timed delays $\sigma_{W \cup L}^\emptyset$ and σ_L^W where in the context of one timed delay no racing delay has yet expired, whereas in the context of the other the winners have expired, creating a disjoint event.

As stochastic delays can form inconsistent races, timed delays can also have inconsistent racing contexts. However, unlike the stochastic delays, the context of the timed delay is static, i.e., the racing condition is not resolved, but only endorsed. We illustrate the situation by an example.

Example 2.4.4 The process $\sigma^x.p_1 + \sigma_x^y.p_2$ can only deadlock. The process $\sigma^x.p_1$ performs a unit time step after which $[X]$ expires. The process $\sigma_x^y.p_2$ performs a unit time step after which $[Y]$ expires in a context of a race in which $[Y]$ won over $[X]$. Thus, the process allows $[X]$ to expire in one timed unit, but it also allows for $[Y]$ to expire in one time unit. However, $[Y]$ should delay less than $[X]$ as implied by the racing context of σ_x^y , which leads to an inconsistency as there is no information about $[Y]$ in context of the first timed delay. \square

Example 2.4.4 also illustrates the main difference between stochastic delays and timed delays in a racing context as $[X].p_1 + [x^y].p_2$ is equivalent to $[x^y].([X].p_1 + p_2)$, after the resolution of the race between $[X]$ and $[x^y]$. This type of dynamics is enabled for the timed delays by the unfolding of the guarded recursive specifications that model the stochastic delays (see Section 5.2 below).

2.5 Design Choices

We model processes using probabilistic timed automata that have probabilistic timed transition systems as the underlying model. We note that the probabilistic timed automata used in the thesis are not related to the probabilistic extensions of timed automata used in PRISM [62]. Processes have

outgoing timed delay transitions and undelayable action transitions that do not allow any passage of time. The choice between several action transitions is nondeterministic and, in general, depends on the environment as in standard process algebras. The choice between timed delays is probabilistic as it is induced by the racing context of the delays. We favor time-determinism, i.e., the principle that passage of time alone cannot make a choice [99, 84, 11]. The probabilistic choices only resolve the race condition, but do not resolve the choice in the alternative composition. Also, we adopt the weak choice between undelayable actions and passage of time, i.e., we impose a nondeterministic choice on the undelayable action transitions and the passage of time in the vein of ACP-styled timed process algebras [84, 11]. To support maximal progress, i.e., to prefer undelayable action to passage of time, we include a maximal progress operator in the theory together with encapsulation of actions, thereby disabling unwanted action transitions. We also opt for guarded recursion introduced by means of guarded recursive specifications. We derive delayable actions as solutions of guarded recursive equations that can perform an undelayable action at any point in time. Stochastic delays are also introduced in the theory using guarded recursive specifications as briefly discussed above. We believe this approach to be systematic as it builds on well-established notions. Moreover, it helps to identify the set of primitive operators that can be combined to bring the other more complex features into the theory.

2.6 Summary

We define two types of race conditions:

1. Independent, which enables compositional modeling by treating every delay as having an independent sample.
2. Dependent, which relates the treatment of stochastic time to standard real time and enables the expansion laws and resolution of races.

Then, we dissect races to unit timed delays in racing contexts, that actually provide information about the expiration of the winning and the losing delays of the race. Such timed delays can be used to derive discrete stochastic delays by means of recursive equations. Finally, we bring the two types of races into the theory by identifying dependent and independent racing delays that induce the corresponding race condition.

In the next section, we introduce the signature of a theory comprising timed delays in racing contexts. We give semantics to closed process terms

using a type of probabilistic timed automata we refer to as racing timed transition schemes.

Chapter 3

Process Theory TCP^{drst}

In this section we begin the introduction to $\text{TCP}_{\text{rec}}^{\text{drst}}(\mathcal{A}, \mathcal{V}, \mathcal{R}, \gamma)$ – the theory of communicating processes with discrete real and stochastic time, where \mathcal{A} denotes the set of actions, \mathcal{V} denotes the set of random variables, \mathcal{R} denotes the set of recursion variables, and γ is the ACP-style [13] commutative and associative action synchronization function. First, we analyze the nonrecursive part of the theory denoted by $\text{TCP}^{\text{drst}}(\mathcal{A}, \mathcal{V}, \gamma)$. We introduce guarded recursion later in Section 4.10 by means of guarded recursive specifications. We give operational semantics to process terms using racing timed transition schemes. We define the strong bisimulation relation and show that it is congruence for the given operators. Afterwards, we use it to define a term model for the theory.

3.1 Racing Timed Transition Schemes

In essence, racing timed transition schemes are probabilistic timed automata in which the probabilistic choice is implicitly and symbolically stated by the racing context of the timed delays. The states determine the timed transitions, whereas we use an additional construct, called an *environment*, to keep track of the ages of the racing delays. It is denoted by a function α that holds the age of the distribution function of each racing delay. We put $\alpha: \mathcal{V} \rightarrow \mathbb{N}$ and we write \mathcal{E} for the set of all such environments. We recall that age 0 actually means that the stochastic delay has no age, i.e., it did not lose any race until that point. The independent racing delays are identified in each state by the function $I(-)$.

Definition 3.1.1 A racing timed transition scheme is a tuple $(S \times \mathcal{E}, A, V, \longrightarrow, \dashrightarrow, \downarrow, I)$, where the extended state $u = \langle s, \alpha \rangle \in S \times \mathcal{E}$

represents a state s in an environment α , A is a set of actions, V is a set of random variables giving the stochastic delay names, and

- $\longrightarrow \subseteq (S \times \mathcal{E}) \times A \times (S \times \mathcal{E})$ is the undelayable action transition relation.
- $\longmapsto \subseteq (S \times \mathcal{E}) \times 2^V \times 2^V \times (S \times \mathcal{E})$ is a timed delay transition relation. For every timed delay transition $u \xrightarrow[W]{W} u'$ (in infix notation) it holds that the winners and the losers are disjoint, i.e., $W \cap L = \emptyset$. Moreover, for every two different timed delay transitions originating from the same state $u \xrightarrow[L_1]{W_1} u_1 \neq u \xrightarrow[L_2]{W_2} u_2$ the predicate $\text{rr}(\sigma_{L_1}^{W_1}, \sigma_{L_2}^{W_2})$ is satisfied.
- $\downarrow \subseteq S \times \mathcal{E}$ is the undelayable termination predicate.
- $\text{I}: S \rightarrow 2^V$ is the independent racing delays function. It satisfies $\text{I}(s) \subseteq \bigcup \{W \cup L \mid \langle s, \alpha \rangle \xrightarrow[L]{W} \langle s', \alpha' \rangle\}$, for every $\alpha \in \mathcal{E}$. \square

Definition 3.1.1 requires that the predicate $\text{rr}(\sigma_{L_1}^{W_1}, \sigma_{L_2}^{W_2})$ holds for every two different timed delay transitions $u \xrightarrow[L_1]{W_1} u_1 \neq u \xrightarrow[L_2]{W_2} u_2$ originating from the same state u . This implies that $W_1 \cup L_1 = W_2 \cup L_2$. Thus, for every state s there exists a set of racing delays $\text{R}(s)$ satisfying $\text{R}(s) = W \cup L$ for every $\langle s, \alpha \rangle \xrightarrow[L]{W} \langle s', \alpha' \rangle$. Then, for the independent racing delays it holds that $\text{I}(s) \subseteq \text{R}(s)$ and the set of dependent racing delays is given by $\text{D}(s) = \text{R}(s) \setminus \text{I}(s)$. For notational convenience, we sometimes write $\text{I}(u)$ instead of $\text{I}(s)$ for $u = \langle s, \alpha \rangle$ and, similar for $\text{R}(u)$ and $\text{D}(u)$. We illustrate the situation by an example.

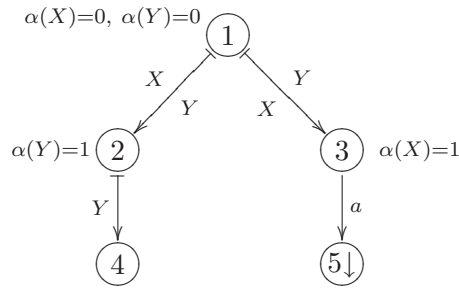


Figure 3.1: Racing timed transition scheme

Example 3.1.2 We depict a racing timed transition scheme as in Figure 3.1. The states are numbered for ease of reference. In state 1 there are two outgoing timed transitions. Note that the race is incomplete as the outcomes where both X and Y are winners or losers are missing. The age of both delays in the beginning is 0. When the transition from state 1 to state 2 is taken the age of the loser Y is increased by 1 because it waits one time unit, whereas X has expired. The racing delays of state 1 are $R(1) = \{X, Y\}$. If we assume that X is an independent delay, i.e., $I(1) = \{X\}$, then Y is a dependent delay and $D(1) = \{X, Y\} \setminus \{X\} = \{Y\}$. The termination predicate holds only in state 5 as indicated by \downarrow . In state 2 the only racing delay is Y , i.e., $R(2) = \{Y\}$ and it is also a dependent delay as it has age 1, so $I(2) = \emptyset$. Also the race in state 2 is not complete as the outcome when Y is a loser is missing. \square

3.2 Probabilistic Timed Transition Systems

A *probabilistic timed transition system* represents an instantiation of a transition scheme with respect to a given assignment $d: V \rightarrow \mathcal{F}$ of the probability distributions. The race condition is used to derive the underlying probability spaces that define the probabilistic behavior of each timed delay transition. In order to compute the correct distributions of the racing delays we will use the environment and the aging function. More precisely, the distribution of a racing delay $[X]$ in an environment α is given by $F_X = d(X)|\alpha(X)$.

Definition 3.2.1 A probabilistic timed transition system $(S, A, \rightarrow, \mapsto, \downarrow)$ is a tuple, where S is the set of states, A is a set of labels, and

- $\rightarrow \subseteq S \times A \times S$ is the action transition relation;
- $\mapsto: S \rightarrow \mathcal{P}(\mathbb{N} \times S)$ is the probabilistic timed transition function; and
- $\downarrow \subseteq S$ is the undelayable termination predicate. \square

Each racing timed transition scheme coupled with an assignment of probability distributions to the stochastic delays induces a probabilistic timed transition system. The action transitions and the termination predicate are adopted from the racing timed transition scheme. The probability measure of the (unit) timed delay is induced by its racing context. The formal definition is as follows.

Definition 3.2.2 Let $R = (S \times \mathcal{E}, A, V, \longrightarrow, \mapsto, \downarrow, \mathbb{I})$ be a racing timed transition scheme and $d: V \rightarrow \mathcal{F}$ a distribution assignment function. Then, the pair (R, d) induces the probabilistic timed transition system $P = (S \times \mathcal{E}, A, \rightarrow, \mapsto, \downarrow)$, where the action transition and termination options \rightarrow and \downarrow of P are given by \longrightarrow and \downarrow of R , respectively, and $\mapsto(u) = ((1, S \times \mathcal{E}), \mathbf{P})$ is the probability space induced by the race condition. The probability measure \mathbf{P} is given by

$$\mathbf{P}(1, u') = \begin{cases} \frac{\text{RC}_1(W', L')}{\sum \{\text{RC}_1(W, L) \mid u \xrightarrow[W]{L} \bar{u}\}} & \text{if } R(u) = W' \cup L' \neq \emptyset \\ 1 & \text{otherwise} \end{cases},$$

where $u \xrightarrow[W]{L} u'$ and $F_X = d(X) | \alpha(X)$ for $X \in R(u)$. \square

We remind the reader that $W' \cup L' = W \cup L$ for every timed delay transition $u \xrightarrow[W]{L} \bar{u}$ of u . The probability measure is normalized because the race need not be complete, i.e., $\sum_{u \xrightarrow[W]{L} \bar{u}} \text{RC}_1(W, L) \leq 1$. Only if the race is complete, i.e., all possible outcomes are stated by the timed delay transitions, the sum above equals one for every possible race. We illustrate the situation by an example.

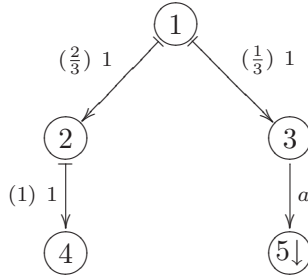


Figure 3.2: Probabilistic timed transition system

Example 3.2.3 Let X and Y be random variables with $\mathbf{P}(X = 1) = \mathbf{P}(X = 2) = \frac{1}{2}$ and $\mathbf{P}(Y = 1) = \frac{1}{3}$, $\mathbf{P}(Y = 2) = \frac{2}{3}$. The probabilistic timed transition system that is induced by the racing timed transition scheme from Example 3.1.2 and the above assignment of distributions to the delays X and Y is depicted in Figure 3.2. The probability mass is indicated in brackets, next to the duration of the timed transition on the \mapsto arrow. As we

deal with unit time steps, the duration of every timed transition is 1. The probability in state 1 that X expires in one time step and Y does not is $\text{RC}_1(X, Y) = \frac{1}{2} \frac{2}{3} = \frac{1}{3}$. The probability in the same state that Y expires in one time step and X does not is $\text{RC}_1(Y, X) = \frac{1}{3} \frac{1}{2} = \frac{1}{6}$. As the race is not complete, the probabilities are normalized to $\frac{2}{3}$ and $\frac{1}{3}$, respectively, as depicted in Figure 3.2. In state 2 the probability is normalized to 1. The action transitions and the termination options are inherited from the racing timed transition scheme. \square

3.3 Bisimulation Relation

We define a strong bisimulation relation on racing timed transition schemes. It requires timed delays to be in the same racing context modulo names of independent delays. This ensures that the related racing timed transition schemes have the same probabilistic behavior, i.e., they induce the same probabilistic timed transition systems when coupled with corresponding distribution assigning functions. As usual, bisimilar terms are required to have the same termination options and action transitions [8, 12].

Definition 3.3.1 Let $R \subseteq (S \times \mathcal{E})^2 \times (\mathcal{V} \leftrightarrow \mathcal{V})$ be a relation. Then R is a racing timed bisimulation if for all $(u_1, u_2, r) \in R$ it holds that also $(u_2, u_1, r^{-1}) \in R$ and $r: R(u_1) \leftrightarrow R(u_2)$ is a bijection with $r(I(u_1)) = I(u_2)$, and $F_X = F_{r(X)}$ and $\alpha_1(X) = \alpha_2(r(X))$ for $X \in \text{dom}(r)$, and:

1. if $u_1 \downarrow$ then $u_2 \downarrow$;
2. if $u_1 \xrightarrow{a} u'_1$ for some $u'_1 \in S \times \mathcal{E}$, then $u_2 \xrightarrow{a} u'_2$ for some $u'_2 \in S \times \mathcal{E}$ such that $(u'_1, u'_2, r') \in R$ for some $r' \in \mathcal{V} \leftrightarrow \mathcal{V}$; and
3. if $u_1 \xrightarrow[L_1]{W_1} u'_1$ for some $u'_1 \in S \times \mathcal{E}$, then $u_2 \xrightarrow[L_2]{W_2} u'_2$ for some $u'_2 \in S \times \mathcal{E}$ where $r(W_1) = W_2$, $r(L_1) = L_2$, and $(u'_1, u'_2, r') \in R$ for some $r' \in \mathcal{V} \leftrightarrow \mathcal{V}$ satisfying $r'(X) = r(X)$ for $X \in L_1 \cap D(u'_1)$.

We say that two states u_1 and u_2 are racing timed bisimilar, notation $u_1 \simeq_t u_2$, if there exists a bisimulation relation R such that $(u_1, u_2, r) \in R$ for some $r \in \mathcal{V} \leftrightarrow \mathcal{V}$. \square

The relationship between racing contexts of timed delays of bisimilar states is established using the bijection r . It is a bijection as the same number of racing delays must be present in both states. It also must respect the independent delays stated by $r(I(u_1)) = I(u_2)$. The independent delays can

have different names, but they must have the same distribution and age, meaning that they will exhibit the same probabilistic behavior. Conditions 1 and 2 state that bisimilar states have the same termination options and action transitions. The timed delay transitions have racing contexts induced by winners and losers related by r . Condition 3 requires that the losers, identified in the resulting state by $L_1 \cap D(u'_1)$, are backward compatible, i.e., they retain their names as they are bound in the first race that they lost. We illustrate the situation by an example.

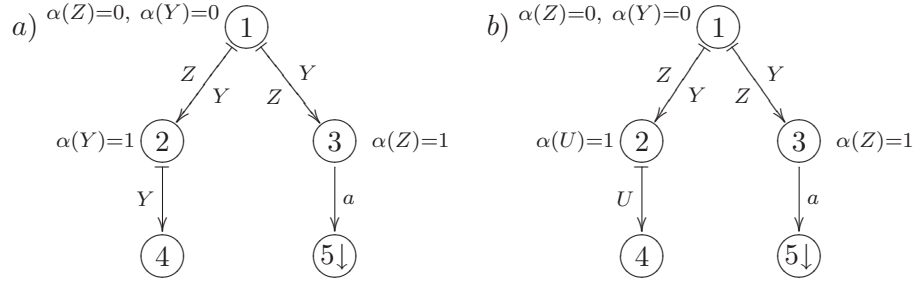


Figure 3.3: Racing timed transition schemes

Example 3.3.2 The racing timed transition scheme depicted in Figure 3.3a is racing timed bisimilar to the one from Figure 3.1 provided that $F_Z = F_X$. As X is an independent racing delay, it can be renamed to the delay Z with the same distribution. However, the racing timed transition scheme depicted in Figure 3.3b is not racing timed bisimilar to the one from Figure 3.1 (nor to the one from Figure 3.3a) even if $F_U = F_Y$. This is because Y is a loser in a previous race and its name must be preserved. As we have strong bisimilarity the action transitions and termination options must be mutually simulated in bisimilar states. \square

As a prerequisite to being a congruence in TCP^{drst} , bisimilarity should be an equivalence relation as stated in the following theorem.

Theorem 3.3.3 *Bisimilarity is an equivalence relation.* \square

PROOF It should be clear that \simeq_t is a reflexive relation, i.e., $u \simeq_t u$, by putting $R = \{(u, u, \text{id}_{R(u)}) \mid u \in S \times \mathcal{E}\}$.

For symmetry, assume that $u \simeq_t v$. Then there exists a bisimulation R such that $(u, v, r) \in R$, for some bijection r satisfying the conditions of

Definition 3.3.1. Put $R' = \{(v, u, r^{-1}) \mid (u, v, r) \in R\}$. Clearly r^{-1} satisfies the conditions of Definition 3.3.1 and R' is a stochastic bisimulation.

For transitivity, assume that $u_1 \rightleftharpoons_t u_2 \rightleftharpoons_t u_3$, i.e., there exist two bisimulation relations R_1 and R_2 such that $(u_1, u_2, r_1) \in R_1$ and $(u_2, u_3, r_2) \in R_2$. Define R_3 as the composition $R_3 = R_2 \circ R_1$, where $r_3 = r_2 \circ r_1$ is again a bijection satisfying the conditions of Definition 3.3.1. It is not difficult to see that R_3 is a racing timed bisimulation, which completes the proof. ■

Next, we introduce the process theory and we give semantics to the process terms using racing timed transitions schemes.

3.4 Signature

We informally introduce the operators before giving the signature of TCP^{drst} . The deadlocked process that does not have any outgoing transitions is denoted by $\underline{0}$; successful termination by $\underline{1}$. Undelayable action prefixing is a unary operator scheme $\underline{a}.$, for every $a \in \mathcal{A}$. Similarly, timed delay prefixes are of the form $\sigma_L^W.$ for $W, L \subseteq \mathcal{V}$ disjoint. The dependence scope operator scheme is given by $|-|_D$, for a dependence binding set $D \subseteq \mathcal{V}$. The encapsulation operator scheme $\partial_H(-)$ for $H \subseteq \mathcal{A}$ blocks the actions in H . The maximal time progress operator scheme $\theta_I(-)$ for $I \subseteq \mathcal{A}$ gives priority to the undelayable actions in I over passage of time. The alternative composition is given by $+$, at the same time representing a nondeterministic choice between action transitions and termination, a weak nondeterministic choice between action and timed delay transitions, and probabilistic choice between the resolved racing contexts of the timed delay transitions. The parallel composition is given by $\|$. It allows passage of time only if both components do so.

Definition 3.4.1 The signature of TCP^{drst} is given by

$$P ::= \underline{0} \mid \underline{1} \mid \underline{a}.P \mid \sigma_L^W.P \mid |P|_D \mid \partial_H(P) \mid \theta_I(P) \mid P + P \mid P \| P,$$

where $a \in \mathcal{A}$, $W, L, D \subseteq \mathcal{V}$ with $W \cap L = \emptyset$, and $H, I \subseteq \mathcal{A}$. The set of closed terms that do not contain term variables is denoted by $\mathcal{C}(\text{TCP}^{\text{drst}})$ and it is ranged over by p and q . □

Next, we take a closer look at the races induced by the timed delay prefixes.

3.5 Auxiliary Operations

The general idea of having both dependent and independent delays available is the following: For specification one can use multiple instances of a component comprising independent delays. As the delays are independent, there is no need to worry about the actual samples. However, outgoing timed delay transitions from the states of the racing timed transition schemes have racing delays with unique names (as there the races are resolved). So, process terms may exhibit naming conflicts. For example, the term $p = |\sigma^x.q|_\emptyset \parallel |\sigma^x.q|_\emptyset$ expresses a race between two components guided by independent delays with the same name. However, the timed delay transitions of $\langle p, \alpha \rangle$ comprise two racing delays with unique different names, but equal distributions.

For p to have proper semantics, the conflicting independent delay names have to be detected and renamed, e.g., to $|\sigma^y.q|_\emptyset \parallel |\sigma^x.q|_\emptyset$ where $F_X = F_Y$. To detect the conflicting racing delay names, we use auxiliary operations $D(p)$ and $I(p)$ to extract the dependent racing delays and the independent racing delay names of the term p , respectively. We say independent delay names instead of independent delays since there might not be one-to-one correspondence between the two in the process terms, e.g., in p from above. Having the dependent racing delays and the independent racing delay names, the set of racing delay names is given by $R(p) = D(p) \cup I(p)$.

One more type of naming conflicts arises when a loser and some new independent delay, which became enabled due to an expiration of a winner, have the same name. For example, such situation is given by the term $\sigma^x.\sigma^y.\underline{0} + \sigma^y.\underline{0}$. If the winner of the race between $[X]$ and $[Y]$ is $[X]$, then the resulting term is $|\sigma^y.\underline{0}|_\emptyset + \sigma^y.\underline{0}$. It has two racing delays with the name Y that do not represent the same racing delay, because the one on the right has age of at least 1, whereas the one on the left is independent (as $[X]$ has no losers it does not induce any dependence) and it has no age at all. To detect this type of naming conflicts, the set of newly enabled independent delay names $N(p)$ of a term p is extracted.

We will use α -conversion to enable dynamic renaming that resolves local naming conflicts in the vein of [41]. Intuitively, α -conversion enables renaming of independent delay names without distorting the structure of the term and conforming to the bisimulation relation. Its definition requires renaming of racing delay names, including the ones that are in the dependence set D of the dependence scope operator $|-|_D$. We refer to the binding delay names of the dependence scope operators encompassing racing delays as the dependence binding delay names and we denote them by $B(p)$.

The definitions of the auxiliary operations are given in Table 3.1. The

dependent racing delays $D(p)$ of the process term p are calculated as the racing delays in the context of the timed delays connected by the outermost composition that are not in any scope; and as the ones that are in the intersection of the dependence sets of all encompassing dependence scope operators. The independent racing delay names cannot be calculated directly, as we need to keep track and exclude the delays of the intersection of the dependence scopes. For that purpose we extend $I(p)$ with an auxiliary (exclusion) set E and obtain $I(p, E)$. Now, the set of independent racing delay names can be computed as the set of dependent racing delays of p excluding the ones in E . Initially, we put $E = \mathcal{V}$ as by default all racing delay names are treated as dependent, i.e., $I(p) = I(p, \mathcal{V})$. The newly enabled independent delay names $N(p)$ are the independent delay names that are introduced in the race due to an expiration of a winner. Note that the losers of the prefixing timed delay are the only dependent delays in the resulting term. The dependence binding delay names $B(p)$ are extracted as the names in the dependence binding sets of the scope operators encompassing racing delays of the topmost race.

$$\begin{aligned}
D(\underline{1}) &= D(\underline{0}) = D(\underline{a}.p) = \emptyset, & D(|p|_D) &= D(p) \cap D, & D(\sigma_L^w.p) &= W \cup L, \\
D(\partial_H(p)) &= D(\theta_I(p)) = D(p), & D(p_1 + p_2) &= D(p_1 \parallel p_2) = D(p_1) \cup D(p_2) \\
I(\underline{1}, E) &= I(\underline{0}, E) = I(\underline{a}.p, E) = \emptyset, & I(\partial_H(p), E) &= I(\theta_I(p), E) = I(p, E) \\
I(p_1 + p_2, E) &= I(p_1 \parallel p_2, E) = I(p_1, E) \cup I(p_2, E) \\
I(|p|_D, E) &= I(p, D \cap E), & I(\sigma_L^w.p, E) &= (W \cup L) \setminus E \\
N(\underline{1}) &= N(\underline{0}) = N(\underline{a}.p) = \emptyset, & N(|p|_D) &= N(\partial_H(p)) = N(\theta_I(p)) = N(p) \\
N(\sigma_L^w.p) &= I(|p|_L), & N(p_1 + p_2) &= N(p_1 \parallel p_2) = N(p_1) \cup N(p_2) \\
B(\underline{1}) &= B(\underline{0}) = B(\underline{a}.p) = B(\sigma_L^w.p) = \emptyset, & B(|p|_D) &= B(p) \cup D \\
B(\partial_H(p)) &= B(\theta_I(p)) = B(p), & B(p_1 + p_2) &= B(p_1 \parallel p_2) = B(p_1) \cup B(p_2)
\end{aligned}$$

Table 3.1: Auxiliary operations

We illustrate the situation by a simple example.

Example 3.5.1 Let $p = \|\sigma_{y,z}^x \sigma^{x,y} \underline{0}\|_{X,Z}|_{X,Y}$. Then (1) $D(p) = \{X\}$ and (2) $I(p) = I(p, \mathcal{V}) = \{Y, Z\}$ because $\mathcal{V} \cap \{X, Z\} \cap \{X, Y\} = \{X\}$, (3) $N(p) = I(|\sigma_{y,z}^x \underline{0}|_{Y,Z}) = \{X\}$, and (4) $B(p) = \{X, Z\} \cup \{X, Y\} = \{X, Y, Z\}$. \square

$$\begin{aligned}
C(\underline{1}) &= C(\underline{0}) = C(\underline{a}.p) = \emptyset, \\
C([\overset{W}{L}].p) &= L \cap I(p) \\
C(|p|_D) &= C(\partial_H(p)) = C(\theta_I(p)) = C(p) \\
C(p_1 + p_2) &= C(p_1 \parallel p_2) = \\
&((I(p_1) \cup N(p_1)) \cap R(p_2)) \cup (R(p_1) \cap (I(p_2) \cup N(p_2))) \cup C(p_1) \cup C(p_2).
\end{aligned}$$

Table 3.2: Set of conflicting names

Remark 3.5.2 We note that in case there is a maximal progress operator in the term, then it may happen that not all timed delay transitions are actually taken because of prioritization of undelayable actions. Hence, the auxiliary operators may actually result in more stochastic delay names than actually observed in the racing contexts of the timed delay transitions. To model this behavior precisely, the operators have to become more complicated in order to examine the behavior of the maximal progress. However, this does not contribute in any sense to the semantics and the only side effect is that the α -conversion and the requirements for naming conflicts defined below yield more delays in some cases. For that reason and for the sake of clarity and compactness we leave these redundant stochastic delay names in place. \square

We proceed by identifying the naming conflicts that may lead to inconsistent probabilistic behavior as discussed above.

3.6 Naming Conflicts

When an independent and a dependent delay or multiple independent delays have the same name, naming conflicts arise that influence the probabilistic behavior of the race. Moreover, naming conflicts arise in the environment when a loser with an age and a newly enabled independent delay have the same name. In principle, all naming conflicts in closed terms can be statically resolved by giving unique names to independent delays [69]. In the current setting, however, we adopt a dynamic approach by using α -conversion in the vein of [41] to support renaming for guarded recursion as well, which cannot easily be handled statically. The set of conflicting names $C(p)$ of a term $p \in \mathcal{C}(\text{TCP}^{\text{drst}})$ is given in Table 3.2.

Conflicts arise when the set of losers and the set of newly enabled independent delays have a common name as given by $C([\overset{W}{L}].p)$. In that case the stochastic delay guiding the losers has an age, but the same stochastic

delay guiding the newly enabled independent delay does not have an age, leading to conflict. Also, compositions can introduce conflicting names as independent or newly enabled independent delay names of one component can overlap with the racing delay names of the other. Here, the search for conflicting names must continue in the components as well, as they also might comprise alternative or parallel compositions.

In case naming conflicts arise, we resolve them using α -conversion as discussed in Section 3.8 below. For the time being, we give the operational semantics for process terms without naming conflicts. In case naming conflicts arise, the process term ‘ignores’ the conflicting behavior by disregarding timed delay transitions with conflicting racing contexts.

3.7 Structural Operational Semantics

The semantics of a term $p \in \mathcal{C}(\text{TCP}^{\text{drst}})$ in an environment $\alpha \in \mathcal{E}$ is given by the racing timed transition scheme $(\mathcal{C}(\text{TCP}^{\text{drst}}) \times \mathcal{E}, \mathcal{A}, \mathcal{V}, \longrightarrow, \longmapsto, \downarrow, \text{I})$, where \longrightarrow , \longmapsto , and \downarrow are defined by the operational rules in Table 3.3 and Table 3.4. For notational convenience, we write α_0 for the environment such that $\alpha_0(X) = 0$, for $X \in \mathcal{V}$. Also, we write $\alpha + 1$ for the function satisfying $(\alpha + 1)(X) = \alpha(X) + 1$. We use four additional predicates in the operational rules: (1) $\langle p, \alpha \rangle \longmapsto$ denoting that the state has an outgoing timed delay transition, (2) $\langle p, \alpha \rangle \not\longmapsto$ denoting that the state has no outgoing timed delay transitions, (3) $\langle p, \alpha \rangle \xrightarrow[W]{L}$ denoting that the state does not have an outgoing timed delay transition with winners W and losers L , and (4) $\langle p, \alpha \rangle \not\xrightarrow{a}$ denoting that the state does not have outgoing action transitions labeled by the action a .

Table 3.3 gives the operational rules for the termination constant, the prefix operators, the dependence scope operator, and the alternative composition. Rule 3.1 states that the termination constant terminates independent of the environment. Rule 3.2 states that action prefixes enable action transitions and reset the ages of the racing delays to the zero environment. Rule 3.3 states that timed delay prefixes enable timed transitions with racing contexts induced by the winners and the losers provided the term does not exhibit naming conflicts. The resulting environment contains the ages of the losers increased by one time unit. Rules 3.4–3.6 show that the dependence scope does not affect the termination nor the outgoing transitions of the term. If the term has an outgoing timed delay transition, then it is conflict-free as the scope operator cannot introduce naming conflicts. Rules 3.7 and 3.8 state that the alternative composition has a termination

$$\begin{array}{l}
\mathbf{3.1} \frac{}{\langle \underline{1}, \alpha \rangle \downarrow} \quad \mathbf{3.2} \frac{}{\langle \underline{a}.p, \alpha \rangle \xrightarrow{a} \langle |p|_{\emptyset}, \alpha_0 \rangle} \quad \mathbf{3.3} \frac{C(\sigma_L^w.p) = \emptyset}{\langle \sigma_L^w.p, \alpha \rangle \vdash_L^w \langle |p|_L, \alpha_0\{(\alpha+1)/L\} \rangle} \\
\mathbf{3.4} \frac{\langle p, \alpha \rangle \downarrow}{\langle |p|_D, \alpha \rangle \downarrow} \quad \mathbf{3.5} \frac{\langle p, \alpha \rangle \xrightarrow{a} \langle p', \alpha' \rangle}{\langle |p|_D, \alpha \rangle \xrightarrow{a} \langle p', \alpha' \rangle} \quad \mathbf{3.6} \frac{\langle p, \alpha \rangle \vdash_L^w \langle p', \alpha' \rangle}{\langle |p|_D, \alpha \rangle \vdash_L^w \langle p', \alpha' \rangle} \\
\mathbf{3.7} \frac{\langle p_1, \alpha \rangle \downarrow}{\langle p_1 + p_2, \alpha \rangle \downarrow} \quad \mathbf{3.8} \frac{\langle p_2, \alpha \rangle \downarrow}{\langle p_1 + p_2, \alpha \rangle \downarrow} \\
\mathbf{3.9} \frac{\langle p_1, \alpha \rangle \xrightarrow{a_1} \langle p'_1, \alpha_1 \rangle}{\langle p_1 + p_2, \alpha \rangle \xrightarrow{a_1} \langle p'_1, \alpha_1 \rangle} \quad \mathbf{3.10} \frac{\langle p_2, \alpha \rangle \xrightarrow{a_2} \langle p'_2, \alpha_2 \rangle}{\langle p_1 + p_2, \alpha \rangle \xrightarrow{a_2} \langle p'_2, \alpha_2 \rangle} \\
\mathbf{3.11} \frac{\langle p_1, \alpha \rangle \vdash_{L_1}^{w_1} \langle p'_1, \alpha_1 \rangle, \langle p_2, \alpha \rangle \dashv \dashv}{\langle p_1 + p_2, \alpha \rangle \vdash_{L_1}^{w_1} \langle p'_1, \alpha_1 \rangle} \quad \mathbf{3.12} \frac{\langle p_1, \alpha \rangle \dashv \dashv, \langle p_2, \alpha \rangle \vdash_{L_2}^{w_2} \langle p'_2, \alpha_2 \rangle}{\langle p_1 + p_2, \alpha \rangle \vdash_{L_2}^{w_2} \langle p'_2, \alpha_2 \rangle} \\
\mathbf{3.13} \frac{\langle p_1, \alpha \rangle \vdash_{L_1}^{w_1} \langle p'_1, \alpha_1 \rangle, \langle p_2, \alpha \rangle \vdash_{L_2}^{w_2} \langle p'_2, \alpha_2 \rangle, \\ (W_1 \cup W_2) \cap (L_1 \cup L_2) = \emptyset, C(p_1 + p_2) = \emptyset}{\langle p_1 + p_2, \alpha \rangle \vdash_{L_1 \cup L_2}^{w_1 \cup w_2} \langle p'_1 + p'_2, \alpha_1\{\alpha_2/L_2\} \rangle} \\
\mathbf{3.14} \frac{\langle p_1, \alpha \rangle \vdash_{L_1}^{w_1} \langle p'_1, \alpha_1 \rangle, \text{rr}(\sigma_{L_1}^{w_1}, \sigma_{L_2}^{w_2}) \text{ for } \langle p_2, \alpha \rangle \vdash_{L_2}^{w_2} \langle p'_2, \alpha_2 \rangle, C(p_1 + p_2) = \emptyset}{\langle p_1 + p_2, \alpha \rangle \vdash_{L_1}^{w_1} \langle p'_1, \alpha_1 \rangle} \\
\mathbf{3.15} \frac{\langle p_2, \alpha \rangle \vdash_{L_2}^{w_2} \langle p'_2, \alpha_2 \rangle, \text{rr}(\sigma_{L_1}^{w_1}, \sigma_{L_2}^{w_2}) \text{ for } \langle p_1, \alpha \rangle \vdash_{L_1}^{w_1} \langle p'_1, \alpha_1 \rangle, C(p_1 + p_2) = \emptyset}{\langle p_1 + p_2, \alpha \rangle \vdash_{L_2}^{w_2} \langle p'_2, \alpha_2 \rangle}
\end{array}$$

Table 3.3: Operational rules for the termination constant, the prefix operators, the dependence scope operator, and the alternative composition operator

option if one of the summands does. Rules 3.9 and 3.10 enable the nondeterministic choice between two action transitions. Rules 3.11 and 3.12 enable the weak choice between action transitions and timed delays. As one summand cannot perform a timed delay, the alternative composition does not introduce a naming conflict. Rule 3.13 gives the synchronization of timed delays when the racing contexts can be merged provided that there are no naming conflicts. We note that the resulting environment can also be represented by $\alpha_2\{\alpha_1/L_1\}$ as the winners from both summands expire together.

Rules 3.14 and 3.15 enable the resolution of races on disjoint events, again provided that there are no naming conflicts. A timed delay transition is in a context of a resolved race if it is in a resolved race with every timed delay transition of the other term. For example, the requirement that the timed delay $\sigma_{L_2}^{W_2}$ of the right summand is in a resolved race is ensured by the condition $\text{rr}(\sigma_{L_1}^{W_1}, \sigma_{L_2}^{W_2})$ for $\langle p_1, \alpha \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \alpha_1 \rangle$.

$$\begin{array}{c}
\mathbf{3.16} \quad \frac{\langle p_1, \alpha \rangle \downarrow, \langle p_2, \alpha \rangle \downarrow}{\langle p_1 \parallel p_2, \alpha \rangle \downarrow} \\
\mathbf{3.17} \quad \frac{\langle p_1, \alpha \rangle \xrightarrow{a_1} \langle p'_1, \alpha_1 \rangle, \langle p_2, \alpha \rangle \dashv \dashrightarrow}{\langle p_1 \parallel p_2, \alpha \rangle \xrightarrow{a_1} \langle p'_1 \parallel p_2, \alpha_1 \rangle} \quad \mathbf{3.18} \quad \frac{\langle p_1, \alpha \rangle \dashv \dashrightarrow, \langle p_2, \alpha \rangle \xrightarrow{a_2} \langle p'_2, \alpha_2 \rangle}{\langle p_1 \parallel p_2, \alpha \rangle \xrightarrow{a_2} \langle p_1 \parallel p'_2, \alpha_2 \rangle} \\
\mathbf{3.19} \quad \frac{\langle p_1, \alpha \rangle \xrightarrow{a_1} \langle p'_1, \alpha_1 \rangle, \langle p_2, \alpha \rangle \dashv \dashrightarrow}{\langle p_1 \parallel p_2, \alpha \rangle \xrightarrow{a_1} \langle p'_1 \parallel p_2, \alpha \rangle} \quad \mathbf{3.20} \quad \frac{\langle p_1, \alpha \rangle \dashv \dashrightarrow, \langle p_2, \alpha \rangle \xrightarrow{a_2} \langle p'_2, \alpha_2 \rangle}{\langle p_1 \parallel p_2, \alpha \rangle \xrightarrow{a_2} \langle p_1 \parallel p'_2, \alpha \rangle} \\
\mathbf{3.21} \quad \frac{\langle p_1, \alpha \rangle \xrightarrow{a_1} \langle p'_1, \alpha_1 \rangle, \langle p_2, \alpha \rangle \xrightarrow{a_2} \langle p'_2, \alpha_2 \rangle, \gamma(a_1, a_2) = a_3}{\langle p_1 \parallel p_2, \alpha \rangle \xrightarrow{a_3} \langle p'_1 \parallel p'_2, \alpha_0 \rangle} \\
\mathbf{3.22} \quad \frac{\langle p_1, \alpha \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \alpha_1 \rangle, \langle p_2, \alpha \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \alpha_2 \rangle, (W_1 \cup W_2) \cap (L_1 \cup L_2) = \emptyset, C(p_1 \parallel p_2) = \emptyset}{\langle p_1 \parallel p_2, \alpha \rangle \xrightarrow{W_1 \cup W_2}_{L_1 \cup L_2} \langle p'_1 \parallel p'_2, \alpha_1 \{ \alpha_2 / L_2 \} \rangle} \\
\mathbf{3.23} \quad \frac{\langle p, \alpha \rangle \downarrow}{\langle \partial_H(p), \alpha \rangle \downarrow} \quad \mathbf{3.24} \quad \frac{\langle p, \alpha \rangle \xrightarrow{a} \langle p', \alpha' \rangle, a \notin H}{\langle \partial_H(p), \alpha \rangle \xrightarrow{a} \langle \partial_H(p'), \alpha' \rangle} \\
\mathbf{3.25} \quad \frac{\langle p, \alpha \rangle \xrightarrow{W}_L \langle p', \alpha' \rangle}{\langle \partial_H(p), \alpha \rangle \xrightarrow{W}_L \langle \partial_H(p'), \alpha' \rangle} \\
\mathbf{3.26} \quad \frac{\langle p, \alpha \rangle \downarrow}{\langle \theta_I(p), \alpha \rangle \downarrow} \quad \mathbf{3.27} \quad \frac{\langle p, \alpha \rangle \xrightarrow{a} \langle p', \alpha' \rangle}{\langle \theta_I(p), \alpha \rangle \xrightarrow{a} \langle \theta_I(p'), \alpha' \rangle} \\
\mathbf{3.28} \quad \frac{\langle p, \alpha \rangle \xrightarrow{W}_L \langle p', \alpha' \rangle, \langle p, \alpha \rangle \dashv \dashrightarrow \text{ for } a \in I}{\langle \theta_I(p), \alpha \rangle \xrightarrow{W}_L \langle \theta_I(p'), \alpha' \rangle}
\end{array}$$

Table 3.4: Operational rules for the parallel composition, the encapsulation, and maximal progress operator

Table 3.4 gives the operational rules for the parallel composition, the

encapsulation, and the maximal progress operator. Rule 3.16 states that the parallel composition can terminate only when both components can. Rules 3.17–3.20 enable interleaving of action transitions in the parallel composition. Rules 3.17 and 3.18 state that the environment is reset when the other component cannot perform a timed delay transition. This is to preserve the desired property that only the ages of the losers persist in the environment. However, the environment must be preserved in case the other component can perform a timed delay as given by rules 3.19 and 3.20. Rule 3.21 allows for synchronization of action transitions if defined by the synchronization function. Similarly to the alternative composition, synchronization of timed delays is allowed when the racing contexts can be merged as given by rule 3.22 provided that there are no naming conflicts. Rule 3.23 states that the termination option is not affected by the encapsulation operator. Rule 3.24 states that action transitions are allowed only if they are not labeled by actions that should be suppressed. Rule 3.25 states that the encapsulation does not affect the timed delays. Rules 3.26 and 3.27 state that the maximal progress operator does not affect the termination options nor the action transitions. Timed delay transitions, however, are exhibited only if the term cannot perform a transition labeled by a prioritized action as given by rule 3.28.

Next, we give a racing timed bisimulation relation on closed TCP^{drst} terms. Intuitively, the names of the dependent racing delays must be preserved, whereas the independent ones must have the same distributions.

Definition 3.7.1 Two terms $p_1, p_2 \in \mathcal{C}(\text{TCP}^{\text{drst}})$ are racing timed bisimilar, notation $p_1 \rightleftharpoons_t p_2$ if there exists a racing timed bisimulation relation R such that $(\langle p_1, \alpha_0 \rangle, \langle p_2, \alpha_0 \rangle, r) \in R$ for some $r \in \mathcal{V} \leftrightarrow \mathcal{V}$ satisfying $r(X) = X$ for $X \in \text{D}(p_1)$. \square

The condition that $r(X) = X$ for $X \in \text{D}(p_1)$ states that bisimilar terms must have the same dependent delays. This preserves the congruence property as dependent delays are explicitly aged by the timed delay prefix σ_L^w , whereas independent delays cannot have an explicit age dependence. The definition may seem restrictive as it deals with process terms only in the zero environment α_0 . However, by an inspection of the operational rules it is easily observed that the environment does not influence the outgoing transitions nor the predicates. It is only used to properly define the underlying probabilistic timed transition system. To show this, we have the following lemma, which also justifies the use of the zero environment.

Lemma 3.7.2 *Let R be a racing timed bisimulation relation and $(\langle p_1, \alpha_1 \rangle, \langle p_2, \alpha_2 \rangle, r) \in R$. Then there exist a racing timed bisimulation relation R' such that $(\langle p_1, \alpha'_1 \rangle, \langle p_2, \alpha'_2 \rangle, r) \in R'$ for every $\alpha'_1, \alpha'_2 \in \mathcal{E}$ satisfying $\alpha'_1(X) = \alpha'_2(r(X))$ for $X \in \text{dom}(r)$. \square*

PROOF It is clear that the initial environments α'_1 and α'_2 satisfy the conditions of Definition 3.3.1 for the bisimulation relation, i.e., corresponding stochastic delays have the same ages. By direct inspection of the operational rules, one concludes that the termination options, the action, and the timed delay transitions do not depend on the aging of the delays, i.e., $\langle p, \alpha \rangle \downarrow$, $\langle p, \alpha \rangle \xrightarrow{a} \langle p', \alpha' \rangle$, and $\langle p, \alpha \rangle \xrightarrow{W/L} \langle p'', \alpha'' \rangle$ for some $a \in \mathcal{A}$, $W, L \subseteq \mathcal{V}$, $p', p'' \in \mathcal{C}(\text{TCP}^{\text{drst}})$, and $\alpha', \alpha'' \in \mathcal{E}$, if and only if $\langle p, \alpha' \rangle \downarrow$, $\langle p, \alpha' \rangle \xrightarrow{a} \langle p', \bar{\alpha}' \rangle$, and $\langle p, \alpha' \rangle \xrightarrow{W/L} \langle p'', \bar{\alpha}'' \rangle$ for some $\alpha', \bar{\alpha}', \bar{\alpha}'' \in \mathcal{E}$. Thus, the states $\langle p_1, \alpha_1 \rangle$ and $\langle p_1, \alpha'_1 \rangle$, and $\langle p_2, \alpha_2 \rangle$ and $\langle p_2, \alpha'_2 \rangle$, respectively, have the same termination options and perform the same action and timed delay transitions. We conclude that the bijections that relate the stochastic delay names in the racing context of the timed delays in R and R' are the same. Now by following the operational rules for $\langle p_1, \alpha_1 \rangle$, $\langle p_1, \alpha'_1 \rangle$, $\langle p_2, \alpha_2 \rangle$, and $\langle p_2, \alpha'_2 \rangle$ it should not be difficult to see that the relation R' that has triples built of the same process terms and bijections relating the random variables of the racing delays as R , but different initial environments, is a bisimulation. \blacksquare

Before we define the term model of TCP^{drst} we provide means to give operational semantics to process terms that exhibit naming conflicts. We follow the approach of [41] and we use α -conversion to rename independent delay names and resolve naming conflicts.

3.8 α -conversion

Intuitively, two terms can be α -converted if they have the same dependent delays and the names of the independent ones are consistently renamed. We illustrate the situation by an example.

Example 3.8.1 The term $\sigma_z^x(\sigma_x^y.\underline{0} + \sigma_y^x.z.\underline{0})$ is α -convertible to $\sigma_v^s(\sigma_t^u.\underline{0} + \sigma_u^t.v.\underline{0})$ provided that $F_X = F_S = F_T$, $F_Y = F_U$, and $F_Z = F_V$. The stochastic delay X of the outermost prefix σ_z^x can be renamed to S , whereas X in the subterm $\sigma_x^y.\underline{0} + \sigma_y^x.z.\underline{0}$ can be renamed to T . These two occurrences of X are independent of each other, having in common only that they are guided by the same distribution function F_X . Both X and Y in the subterm $\sigma_x^y.\underline{0} + \sigma_y^x.z.\underline{0}$ must be consistently renamed to T and U in the subterm

$\sigma_V^U.\underline{0} + \sigma_V^{T,V}.\underline{0}$, respectively. This is to preserve the correct probabilistic behavior as they are dependent delays in the corresponding subterms. The loser Z of the topmost race is a dependent delay is aged because of the timed transition of the prefixing delay σ_Z^X . So, its name is bound and it must be consistently renamed in the whole term to V . \square

To formalize the renaming as illustrated by Example 3.8.1, we introduce a predicate $\text{ccr}_{d,i}(p_1, E_1, p_2, E_2)$ that checks whether the stochastic delays of the closed terms p_1 and p_2 have been consistently renamed. Renaming of dependent racing delays is represented by a bijection d between the union of the dependent racing and dependence binding delay names of the terms. It is a bijection because dependent racing and dependence binding delay names of one term can have only one counterpart in the other. The renaming of the independent racing delay names is given by a total surjective relation i . It is a relation because there might be multiple stochastic delays with the same name related to their counterpart with different names. For example, the renaming of X in $|\sigma^X.\underline{0}|_\emptyset + |\sigma^X.\underline{0}|_\emptyset$ to both Y and Z in $|\sigma^Y.\underline{0}|_\emptyset + |\sigma^Z.\underline{0}|_\emptyset$ provided that $F_X = F_Y = F_Z$. It must be a total and surjective relation as all independent delay names from one term must be related to some independent delay names of the other. Still, the renaming must be consistent with respect to the subterm in which independent delay names are renamed, e.g., the renaming of X to T in the subterm $\sigma_X^Y.\underline{0} + \sigma_Y^{X,Z}.\underline{0}$ in Example 3.8.1. As in the definition of $I(p)$, to extract the independent delay names, we need auxiliary (exclusion) sets of delays E_1 and E_2 that keep track of the intersections of the dependence binding sets. Finally, two states can be α -converted if the process terms can be α -converted and, moreover, the environments differ only on the independent delay names provided that corresponding delays have the same age.

Definition 3.8.2 Two closed terms $p_1, p_2 \in \mathcal{C}(\text{TCP}^{\text{drst}})$ are α -convertible, notation $p_1 \sim_\alpha p_2$, if the predicate $\text{ccr}_{d,i}(p_1, \mathcal{V}, p_2, \mathcal{V})$ given in Table 3.5 holds, for the identity bijection $d: D(p_1) \cup B(p_1) \leftrightarrow D(p_2) \cup B(p_2)$ and a total surjective relation $i \subseteq I(p_1) \times I(p_2)$.

Two states $\langle p_1, \alpha_1 \rangle, \langle p_2, \alpha_2 \rangle \in \mathcal{C}(\text{TCP}^{\text{drst}}) \times \mathcal{E}$ are α -convertible, notation $\langle p_1, \alpha_1 \rangle \sim_\alpha \langle p_2, \alpha_2 \rangle$, if $p_1 \sim_\alpha p_2$ and the environment differs only on the racing independent delays provided that renamed delays have the same age, i.e., $\alpha_1(X) = \alpha_2(Y)$ for every $(X, Y) \in i$, and $\alpha_0\{\alpha_1/(\mathcal{V} \setminus I(p_1))\} = \alpha_0\{\alpha_2/(\mathcal{V} \setminus I(p_2))\}$. \square

$$\begin{aligned}
& \text{ccr}_{d,i}(\underline{1}, E_1, \underline{1}, E_2) = \text{ccr}_{d,i}(\underline{0}, E_1, \underline{0}, E_2) = \top \\
& \text{ccr}_{d,i}(\underline{a}.p_1, E_1, \underline{a}.p_2, E_2) \text{ if } \text{ccr}_{d',i'}(p_1, \mathcal{V}, p_2, \mathcal{V}) \\
& \quad \text{for a bijection } d': D(p_1) \cup B(p_1) \leftrightarrow D(p_2) \cup B(p_2) \text{ and} \\
& \quad \text{a total surjective relation } i' \subseteq I(p_1) \times I(p_2) \\
& \text{ccr}_{d,i}(\sigma_{L_1}^{W_1}.p_1, E_1, \sigma_{L_2}^{W_2}.p_2, E_2) \\
& \quad \text{if there exists a bijection } j: (W_1 \cup L_1) \setminus E_1 \leftrightarrow (W_2 \cup L_2) \setminus E_2 \\
& \quad \text{satisfying } j(X) = Y \text{ if } (X, Y) \in i, \\
& \quad j(W_1 \setminus E_1) = W_2 \setminus E_2, \quad j(L_1 \setminus E_1) = L_2 \setminus E_2, \\
& \quad F_X = F_{j(X)} \text{ for } X \in (W_1 \cup L_1) \setminus E_1, \text{ and} \\
& \quad d(W_1 \cap E_1) = W_2 \cap E_2, \quad d(L_1 \cap E_1) = L_2 \cap E_2, \\
& \quad F_X = F_{d(X)} \text{ for } X \in (W_1 \cup L_1) \cap E_1, \text{ and} \\
& \quad \text{ccr}_{d',i'}(p_1, \mathcal{V}, p_2, \mathcal{V}) \text{ holds} \\
& \quad \text{for a bijection } d': D(p_1) \cup B(p_1) \leftrightarrow D(p_2) \cup B(p_2) \\
& \quad \text{with } d'(X) = d(X) \text{ for } X \in L_1 \cap E_1 \cap D(p_1) \text{ and} \\
& \quad d'(X) = i(X) \text{ for } X \in (L_1 \setminus E_1) \cap D(p_1), \text{ and} \\
& \quad \text{a total surjective relation } i' \subseteq I(p_1) \times I(p_2) \\
& \text{ccr}_{d,i}(|p_1|_{D_1}, E_1, |p_2|_{D_2}, E_2) \text{ if } d(D_1) = D_2 \text{ and } \text{ccr}_{d,i}(p_1, D_1 \cap E_1, p_2, D_2 \cap E_2) \\
& \text{ccr}_{d,i}(\partial_H(p_1), E_1, \partial_H(p_2), E_2) \text{ if } \text{ccr}_{d,i}(p_1, E_1, p_2, E_2) \\
& \text{ccr}_{d,i}(\theta_I(p_1), E_1, \theta_I(p_2), E_2) \text{ if } \text{ccr}_{d,i}(p_1, E_1, p_2, E_2) \\
& \text{ccr}_{d,i}(p_1 + p'_1, E_1, p_2 + p'_2, E_2) \text{ if } \text{ccr}_{d,i}(p_1, E_1, p_2, E_2) \text{ and } \text{ccr}_{d,i}(p'_1, E_1, p'_2, E_2) \\
& \text{ccr}_{d,i}(p_1 \parallel p'_1, E_1, p_2 \parallel p'_2, E_2) \text{ if } \text{ccr}_{d,i}(p_1, E_1, p_2, E_2) \text{ and } \text{ccr}_{d,i}(p'_1, E_1, p'_2, E_2)
\end{aligned}$$
Table 3.5: Definition of $\text{ccr}_{d,i}(-)$

We comment on the definition of the predicate $\text{ccr}_{d,i}(-)$. As an example we consider the renaming of the term p_1 to p_2 given as follows:

$$\begin{aligned}
p_1 &= |\sigma_v^x.\sigma^y.\underline{0}|_\emptyset + |\sigma_x^z.\underline{a}.\sigma^x.\underline{0}|_Z \\
p_2 &= |\sigma_v^u.\sigma^v.\underline{0}|_\emptyset + |\sigma_w^z.\underline{a}.\sigma^x.\underline{0}|_Z.
\end{aligned}$$

The bijection d relating dependent racing delays relates only Z and Z . The total surjective relation i contains the pairs (X, U) , (Y, V) , and (X, W) .

The renaming of the constant processes is always consistent. The action prefix is α -convertible as long as the remaining process is α -convertible, ex-

pressed by the existence of the bijection d' and the total surjective relation i' . Recall that all racing delays prefixed by an action prefix are independent. For that reason the occurrence of X following the action prefix in p_1 can remain with the same name in p_2 even though the occurrence of X in σ_x^z has been renamed to W in p_2 .

The most involved consistency requirement is for the timed delay prefix. First, the independent delays must be isolated by subtracting the exclusion sets from the racing delays. Then, there has to be a one-to-one correspondence between the independent racing delays of the racing contexts. This is expressed by the bijection j that respects the relation i between the independent racing delay names. The independent delays in the racing context of $\sigma_{L_1}^{W_1}$ and $\sigma_{L_2}^{W_2}$ are identified as the ones that are not in the exclusion sets E_1 and E_2 , respectively. Then, there must a correspondence between the independent winners and losers, respectively, such that they have the same distribution functions. The remaining processes must also be α -convertible, which is given by the existence of the bijection d' and the total surjective relation i' . The bijection d' that relates the dependent delays of the remaining processes must respect the names of both independent and dependent losers as stated by the last two conditions. In the example, the bijection j relating the independent delays of the timed prefix of the first summand relates $X \mapsto U$ and $Y \mapsto V$. Note that there can not be multiple occurrences of the same independent delay in one racing context, so j can always be defined if the renaming is consistent. The bijection d' must respect the renaming of $Y \mapsto V$, so in the remaining term Y continues to be renamed as V . Note that this is not the case in the second summand, as the action prefix resets the race and all delays become independent.

For the dependence scopes, the dependence binding sets must be related and the remaining processes are checked with the adapted exclusion sets. The alternative and the parallel composition are α -convertible if the components are. The encapsulation and the maximal progress operator are α -convertible if the encompassed processes are.

We add one more operational rule to the ones in Table 3.3 and Table 3.4 that exploits α -conversion to resolve naming conflicts as follows:

$$\mathbf{3.29} \quad \frac{\langle p, \alpha \rangle \sim_\alpha \langle p'', \alpha'' \rangle, \langle p'', \alpha'' \rangle \xrightarrow[L]{W} \langle p', \alpha' \rangle, C(p'') = \emptyset}{\langle p, \alpha \rangle \xrightarrow[L]{W} \langle p', \alpha' \rangle}.$$

This rule renames the independent delays that cause conflicts, thus keeping the timed delay transitions locally free of conflict. The approach is similar to the one of [41].

Remark 3.8.3 Here, however, we are slightly more liberal as rule 3.29 can potentially produce infinitely many transitions, although its purpose is to support the resolution of possible naming conflicts. More precisely, the rule allows for a renaming of an independent delay to every other non-conflicting stochastic delay, whereas the intention is to use it only once. One way to formally resolve this would be to alter the logic that drives the deduction of the operational rules by introducing the ∇ operator of [79] that enables local scopes. This operator locally binds an arbitrary name, enabling a choice of names for the conflicting stochastic delay that resolves the naming conflicts. In [79] an embedding of late π -calculus is given in the extended logic that formalizes α -conversion in that setting. Another approach would be to adopt the approach of history-based automata in order to explicitly represent the dependencies between variable names by means of relations that keep the past behavior of the system [81]. Also in this setting, a translation of π -calculus to the proposed theory is given that shows how to explicitly model α -conversion. In the current setting, however, we decide not to explicitly model the one-time usage of the α -conversion rule as this goes beyond the scope of our work and does not contribute to the presentation of ideas in the current setting. \square

The following theorem shows that α -conversion is a congruence on closed TCP^{drst} terms. This theorem in combination with Theorem 3.8.5, which shows that α -congruence implies bisimulation, enables the treatment of the process terms modulo α -conversion, i.e., modulo naming of independent delays.

Theorem 3.8.4 *α -conversion is a congruence on $\mathcal{C}(\text{TCP}^{\text{drst}})$.* \square

PROOF It should be clear that α -conversion is an equivalence relation as it is based on bijections to provide renaming of the stochastic delays. To show reflexivity, take d to be the identity bijection and i the identity relation. To show symmetry, suppose that $p_1 \sim_\alpha p_2$ for some bijection d and some total surjective relation i . Now, $p_2 \sim_\alpha p_1$ by using the reverse bijection d^{-1} and the total surjective relation i^{-1} . Transitivity follows from the fact that composition of two bijections is again a bijection and a composition of two total surjective relations is again a total surjective relation.

It is straightforward that α -conversion is a congruence for the trivial contexts of $\underline{0}$ and $\underline{1}$.

Now, suppose that $p_1 \sim_\alpha p_2$ and that $\text{ccr}_{d,i'}(p_1, \mathcal{V}, p_2, \mathcal{V})$ holds for the identity bijection $d' = \text{id}_{D(p_1) \cup B(p_1)}$ and some total surjective relation $i' \subseteq I(p_1) \times I(p_2)$.

For the action prefixed terms we readily have that $\underline{a}.p_1 \sim_\alpha \underline{a}.p_2$ because the conditions are trivially satisfied for the empty bijection and the empty total surjective relation on $\emptyset \times \emptyset$ as there are no racing delays. The predicate $\text{ccr}_{\emptyset, \emptyset}(a.p_1, \mathcal{V}, a.p_2, \mathcal{V})$ holds as $\text{ccr}_{d', i'}(p_1, \mathcal{V}, p_2, \mathcal{V})$ holds.

For the timed delay prefixed terms $\sigma_L^w.p_1$ and $\sigma_L^w.p_2$, we have that the dependent delays are the same in both terms and that there are no independent terms. Thus, $\sigma_L^w.p_1 \sim_\alpha \sigma_L^w.p_2$ as $\text{ccr}_{d, \emptyset}(\sigma_L^w.p_1, \mathcal{V}, \sigma_L^w.p_2, \mathcal{V})$ holds for the identity bijection $d = \text{id}_{W \cup L}$ that is respected by the identity bijection $d' = \text{id}_{L \cup D(p_1)}$.

For the encapsulation operator and the maximal progress operator it is straightforward that $\partial_H(p_1) \sim_\alpha \partial_H(p_2)$ and $\theta_I(p_1) \sim_\alpha \theta_I(p_2)$ as the dependent, dependence binding, and independent delays are the same as for p_1 and p_2 . Therefore, $\text{ccr}_{d', i'}(\partial_H(p_1), \mathcal{V}, \partial_H(p_2), \mathcal{V})$ and $\text{ccr}_{d', i'}(\theta_I(p_1), \mathcal{V}, \theta_I(p_2), \mathcal{V})$ hold.

The dependence delays scope operator $|-|_D$ can introduce additional independent and dependence binding delays. We obtain the bijection d as the identity bijection $d = \text{id}_{D(\text{dom}(d') \cup D)}$. The total surjective relation i is obtained by extending i' with the additional independent delays as $i = i' \cup \text{id}_{D(p_1) \setminus D}$. Trivially $d(D) = D$. We proceed by analyzing $\text{ccr}_{d, i}(p_1, D, p_2, D)$. Assume that $p_1 = \sigma_{L_1}^{w_1}.p'_1$ and $p_2 = \sigma_{L_2}^{w_2}.p'_2$. Then $d'(W_1) = W_2$, $d'(L_1) = L_2$, and $i' = \emptyset$ for the identity bijection d' . It is not difficult to see that in this case the bijection j is the identity bijection $j = \text{id}_{(W_1 \cup L_1) \setminus D}$ and that $\text{ccr}_{d, i}(p_1, D, p_2, D)$ holds. Next, assume that $\bar{p}_1 = |\bar{p}'_1|_{D_1}$ and $\bar{p}_2 = |\bar{p}'_2|_{D_2}$ after several applications of the rule. Then, $d'(D_1) = D_2$ and $\text{ccr}_{d', i'}(\bar{p}_1, E_1, \bar{p}_2, E_2)$ holds for $d'(E_1) = E_2$. Again, the same cases repeat except for the timed delay prefix. In this case we extend the existing bijection j' with $\text{id}_{D(p_1) \setminus E_1}$ to obtain j , which is covered by the definition of i . Thus, we conclude that $\text{ccr}_{d, i}(|p_1|_D, \mathcal{V}, |p_2|_D, \mathcal{V})$ holds.

Now, suppose that $p'_1 \sim_\alpha p'_2$ and that $\text{ccr}_{d'', i''}(p'_1, \mathcal{V}, p'_2, \mathcal{V})$ holds for the identity bijection $d'' = \text{id}_{D(p'_1) \cup B(p'_1)}$ and some total surjective relation $i'' \subseteq I(p'_1) \times I(p'_2)$.

To show that $p_1 + p'_1 \sim_\alpha p_2 + p'_2$ and $p_1 \parallel p'_1 \sim_\alpha p_2 \parallel p'_2$, we put d to be the identity bijection $d = \text{id}_{D(p_1 + p'_1)}$. It should be clear that it conforms to d' and d'' . We build i as the union of i' and i'' , i.e., $i = i' \cup i''$. Now, we have that $\text{ccr}_{d, i}(p_1 + p'_1, \mathcal{V}, p_2 + p'_2, \mathcal{V})$ and $\text{ccr}_{d, i}(p_1 \parallel p'_1, \mathcal{V}, p_2 \parallel p'_2, \mathcal{V})$ hold as both $\text{ccr}_{d, i}(p_1, \mathcal{V}, p_2, \mathcal{V})$ and $\text{ccr}_{d, i}(p'_1, \mathcal{V}, p'_2, \mathcal{V})$ hold, which completes the proof. ■

Because α -conversion is a congruence, we will also refer to it as α -congruence. The following theorem states that α -congruence implies racing timed bisimilarity.

Theorem 3.8.5 *If $\langle p_1, \alpha_1 \rangle \sim_\alpha \langle p_2, \alpha_2 \rangle$ then $\langle p_1, \alpha_1 \rangle \rightleftharpoons_t \langle p_2, \alpha_2 \rangle$.* □

PROOF If $C(p_1) = \emptyset$ and $C(p_2) = \emptyset$ hold, then the relation i giving the renaming of independent racing delays becomes a one-to-one total surjection and, therefore, a bijection. Moreover, $\text{dom}(i) \cap \text{dom}(d) = \emptyset$. Now, it should not be difficult to observe that the union $d \cup i$ is the renaming bijection r of the bisimulation relation between the states, whereas the condition on the environments is satisfied by the definition of α -conversion. ■

We will show that bisimulation is a congruence, which paves the way for defining a term model for the process theory.

3.9 Term Model

The congruence property of the racing timed bisimilarity is stated in the following theorem, which is a requirement for the definition of the term model.

Theorem 3.9.1 *The racing timed bisimilarity relation \simeq_t is a congruence on $\mathcal{C}(\text{TCP}^{\text{drst}})$.* □

PROOF Suppose that $p_1 \simeq_t p_2$ and $p'_1 \simeq_t p'_2$. Then there exist racing timed bisimulation relations R and R' , respectively, such that $(\langle p_1, \alpha_0 \rangle, \langle p_2, \alpha_0 \rangle, r) \in R$ and $(\langle p'_1, \alpha_0 \rangle, \langle p'_2, \alpha_0 \rangle, r') \in R'$. The trivial contexts $\underline{0}$ and $\underline{1}$ are clearly bisimilar.

[$\underline{a}.$] Define $R'' = \{(\langle \underline{a}.p_1, \alpha_0 \rangle, \langle \underline{a}.p_2, \alpha_0 \rangle, \emptyset)\} \cup R$. That R'' is a racing timed bisimulation relation follows from the fact that only the following action transitions are possible: $\langle \underline{a}.p_1, \alpha_0 \rangle \xrightarrow{a} \langle |p_1|_\emptyset, \alpha_0 \rangle$ and $\langle \underline{a}.p_2, \alpha_0 \rangle \xrightarrow{a} \langle |p_2|_\emptyset, \alpha_0 \rangle$, and that the rest is captured by the bisimulation R .

[$\sigma_L^w.$] By using Lemma 3.7.2, let R''' be the bisimulation relation relating $\langle p_1, \alpha_0 \{ \alpha_0 + 1/L \} \rangle$ and $\langle p_2, \alpha_0 \{ \alpha_0 + 1/L \} \rangle$. Define $R'' = \{(\langle \sigma_L^w.p_1, \alpha_0 \rangle, \langle \sigma_L^w.p_2, \alpha_0 \rangle, \text{id}_{W \cup L})\} \cup R'''$. It should be clear that R'' is a racing timed bisimulation relation.

[$| _D$] Define $R'' = \{(\langle |p_1|_D, \alpha_0 \rangle, \langle |p_2|_D, \alpha_0 \rangle, r)\} \cup R$. By direct inspection of the operational rules we have that the processes $\langle |p|_D, \alpha \rangle$ and $\langle p, \alpha \rangle$ have the same termination options and action transitions, and result in the same states. Putting the term p in a dependence scope may just turn a dependent delay into an independent one. However, the racing delay names remain the same. Thus, $|p_1|_D$ and $|p_2|_D$ have the same timed delay transitions as p_1 and p_2 , respectively, which makes R'' a racing timed bisimulation relation.

[$\partial_H(-)$] Define $R'' = \{(\langle \partial_H(p_1), \alpha_1 \rangle, \langle \partial_H(p_2), \alpha_2 \rangle, r) \mid (\langle p_1, \alpha_1 \rangle, \langle p_2, \alpha_2 \rangle, r) \in R\}$. By inspection of the operational rules for $\partial_H(-)$ it should be clear

that R'' is a racing timed bisimulation relation by using the same bijection for the stochastic delays as R .

$[\theta_I(-)]$ Define $R'' = \{(\langle \theta_I(p_1), \alpha_1 \rangle, \langle \theta_I(p_2), \alpha_2 \rangle, r') \mid (\langle p_1, \alpha_1 \rangle, \langle p_2, \alpha_2 \rangle, r) \in R\}$. Now, the proof is the same as for $\partial_H(-)$.

$[- + -]$ Before defining the bisimulation relation we analyze the alternative composition of p_1 and p'_1 . If $\langle p_1 + p'_1, \alpha_0 \rangle \downarrow$, then either $\langle p_1, \alpha_0 \rangle \downarrow$ or $\langle p'_1, \alpha_0 \rangle \downarrow$, and consequently, either $\langle p_2, \alpha_0 \rangle \downarrow$ or $\langle p'_2, \alpha_0 \rangle \downarrow$. It easily follows that $\langle p_2 + p'_2, \alpha_0 \rangle \downarrow$. Similarly for the other direction.

If $\langle p_1, \alpha_0 \rangle \xrightarrow{a} \langle \bar{p}_1, \alpha_0 \rangle$, then $R'' = R$ on this part of the transition scheme. In the symmetric case when $\langle p'_1, \alpha_0 \rangle \xrightarrow{a} \langle \bar{p}'_1, \alpha_0 \rangle$, we have $R'' = R'$.

Possible naming conflicts arise in $p_1 + p'_1$ if $C(p_1 + p'_1) \neq \emptyset$. Let p''_1 and p'''_1 be the α -converted versions of p_1 and p'_1 , respectively, such that $C(p''_1 + p'''_1) = \emptyset$ holds. By Theorem 3.8.5 there exist racing timed bisimulation relations R_1 and R'_1 such that $(\langle p_1, \alpha_0 \rangle, \langle p''_1, \alpha_0 \rangle, r_1) \in R_1$ and $(\langle p'_1, \alpha_0 \rangle, \langle p'''_1, \alpha_0 \rangle, r'_1) \in R'_1$. Similarly for p_2 and p'_2 we have that $(\langle p_2, \alpha_0 \rangle, \langle p''_2, \alpha_0 \rangle, r_2) \in R_2$ and $(\langle p'_2, \alpha_0 \rangle, \langle p'''_2, \alpha_0 \rangle, r'_2) \in R'_2$, for some racing timed bisimulation relations R_2 and R'_2 and $p''_2 \sim_\alpha p_2$ and $p'''_2 \sim_\alpha p'_2$. Now, we define r'' as $r'' = r_2 \text{ or } r_1 \cup r'_2 \text{ or } r'_1$. It is well defined as $C(p''_1 + p'''_1) = \emptyset$ and $C(p''_2 + p'''_2) = \emptyset$ hold.

Now, we construct the racing timed bisimulation relation R'' . If p_1 or p'_1 performs an action transition then the alternative composition degrades to a part of the transitions schemes of p_1 or p'_1 , respectively. Similarly, if p_1 or p'_1 perform a timed delay in a resolved race. Thus, we initially put $R'' = R \cup R' \cup \{(\langle p_1 + p'_1, \alpha_0 \rangle, \langle p_1 + p'_1, \alpha_0 \rangle, r'')\}$. However, when the summands synchronize on performing a timed delay transition, i.e., when the summands perform timed delays that induce an unresolved race, then both results of the timed delay transitions persist in the final term. In this case, one proceeds in the same manner as before by induction and, again, the union of the composition of the bijections induced by the α -conversion is computed to obtain the bijection for the racing timed bisimulation relation between the alternative compositions.

$[- \parallel -]$ As in the case of the alternative composition, with the exception that an action transition does not make a choice. \blacksquare

Now, we have all the prerequisites to define the term model of TCP^{drst} as the quotient algebra modulo racing timed bisimulation [13].

Definition 3.9.2 The term model of TCP^{drst} is the quotient algebra $\mathbb{P}(\text{TCP}^{\text{drst}})/\simeq_t$, where $\mathbb{P}(\text{TCP}^{\text{drst}}) = (\mathcal{C}(\text{TCP}^{\text{drst}}), \underline{0}, \underline{1}, \underline{a}, \cdot \text{ for } a \in \mathcal{A}, \sigma_L^w \text{ for } W, L \subseteq \mathcal{V}, \text{ satisfying } W \cap L = \emptyset, |-\!|_D \text{ for } D \subseteq \mathcal{V}, \partial_H(-) \text{ for } H \subseteq \mathcal{A}, \theta_I(-) \text{ for } I \subseteq \mathcal{A}, - + -, - \parallel -)$. \square

Remark 3.9.3 We note that because of the congruence property of α -conversion and because it implies racing timed bisimilarity, we could also take the set of processes to be $(\mathbb{P}(\text{TCP}^{\text{drst}})/\sim_\alpha)/\Leftrightarrow_t$ as originally done in [66, 68]. However, in the current setting we opt for explicit equations to show α -conversion, as we believe that this provides an additional insight in the nature of the process theory. \square

3.10 Summary

We give the signature and semantics of the theory of communicating processes with discrete real and stochastic time – TCP^{drst} . The theory comprises timed delays in racing contexts that can express an expiration of an outcome of a race per time unit. The semantics is given in terms of racing timed transitions schemes that present a kind of probabilistic timed automata where the probabilistic choices are symbolically given by the race condition. Consequently, an assignment of probability distributions to the stochastic delays induces a probabilistic timed transition system.

As multiple independent racing delays can have the same name, the racing semantics can be ambiguous when two independent delays with the same name occur simultaneously in a race. We resolve this problem by employing α -conversion. In the following chapter we provide a sound and ground-complete axiomatization of the developed theory.

Chapter 4

Equational Theory

As we mentioned before, the associativity of the alternative composition does not hold for process terms that induce incomplete races. Moreover, the expansion of the parallel composition and the resolution of the maximal progress operator require resolved races. This forces us to give the theory TCP^{drst} in terms of equations on normal forms. First, we give axioms for manipulation of the dependence scope operator. We employ them to derive an intermediate normal form that enumerates all possible outcomes of a race, making the alternative composition associative. It is unique for the timed delays modulo commutativity, associativity, and naming of independent delays. We use this normal form to give expansion laws for the rest of the operators, such that the expansions are again in the same normal form. Afterwards, we define a head norm form, in which every operator except the alternative composition and the prefix operators is eliminated. Relying on it, we show that the equational theory presented is ground-complete. At the end of this section, we introduce guarded recursion by means of guarded recursive specifications, which have unique solutions in the term model of $\text{TCP}_{\text{rec}}^{\text{drst}}$.

4.1 Renaming of Independent Delays

As already elaborated upon, the main idea behind having two types of race condition is that systems are modeled by independent delays whereas, the race condition is resolved by assigning unique names to racing delays and afterwards treating them as dependent. Thus, we need a mechanism for renaming independent delays and turning them into dependent ones. We give a simple example to illustrate the situation.

Example 4.1.1 Given the simple component $|\sigma_Y^x.\sigma^y.\underline{a}.\underline{0}|_\emptyset$, we can use it as a building block of the system $|\sigma_Y^x.\sigma^y.\underline{a}.\underline{0}|_\emptyset \parallel |\sigma_Y^x.\sigma^y.\underline{a}.\underline{0}|_\emptyset$. However, for analysis we revert to the system $|\sigma_Y^x.\sigma^y.\underline{a}.\underline{0}|_\emptyset \parallel |\sigma_V^u.\sigma^v.\underline{a}.\underline{0}|_\emptyset$, where $F_X = F_U$ and $F_Y = F_V$. The advantage of encompassing the whole term within a single dependence scope is that all independent delays are given unique names. Moreover, the dependent delays are ‘declared’ in the dependence binding set, the parameter of the dependence scope. \square

Proper resolution of the race condition requires uniqueness of names of the racing delays as suggested by Definition 3.1.1 (for more details also refer to the maximal distinct representation of terms in [68]). The mechanism that enforces all independent delays to be assigned a different name is to encompass them using a single dependence scope.

It is clear that naming conflicts may arise in such a situation, as in Example 4.1.1 above. Therefore, it has to be checked whether there are independent racing delays with conflicting names and the ones introducing the clash must be renamed. Care has to be taken to rename losing delays consistently as their names are made dependent and bound by the winners in the first race that they lost. To this end, we define a renaming operation $p^{[Y/X]}$ (cf. Table 4.1) that consistently renames the stochastic delay X into Y in the term $p \in \mathcal{C}(\text{TCP}^{\text{drst}})$.

$$\begin{array}{lll}
\underline{0}^{[Y/X]} = \underline{0}, & \underline{1}^{[Y/X]} = \underline{1}, & (\underline{a}.p)^{[Y/X]} = \underline{a}.p \\
\partial_H(p)^{[Y/X]} = \partial_H(p^{[Y/X]}), & \theta_I(p)^{[Y/X]} = \theta_I(p^{[Y/X]}) & \\
(p_1 + p_2)^{[Y/X]} = p_1^{[Y/X]} + p_2^{[Y/X]}, & (p_1 \parallel p_2)^{[Y/X]} = p_1^{[Y/X]} \parallel p_2^{[Y/X]} & \\
(\sigma_L^w.p)^{[Y/X]} = \sigma_L^w.p & & \text{if } X \notin W \cup L \\
(\sigma_L^w.p)^{[Y/X]} = \sigma_L^{(W \setminus \{X\}) \cup \{Y\}}.p & & \text{if } X \in W \\
(\sigma_L^w.p)^{[Y/X]} = \sigma_{(L \setminus \{X\}) \cup \{Y\}}^w.p^{[Y/X]} & & \text{if } X \in L \\
(|p|_D)^{[Y/X]} = |p|_D & & \text{if } X \notin D \\
(|p|_D)^{[Y/X]} = |p^{[Y/X]}|_{(D \setminus \{X\}) \cup \{Y\}} & & \text{if } X \in D
\end{array}$$

Table 4.1: Renaming operation

By now, we have gathered all the prerequisites to present the axioms and the expansion laws for the operators.

4.2 Dependence Scope

We begin by giving the axioms for manipulating with the dependence scope operators in Table 4.2. Axioms A4.1–A4.3 deal with terms that have no

$ \underline{0} _{\emptyset} = \underline{0}$	A4.1
$ \underline{1} _{\emptyset} = \underline{1}$	A4.2
$ \underline{a}.p _{\emptyset} = \underline{a}.p$	A4.3
$\underline{a}.p = \underline{a}. p _{\emptyset}$	A4.4
$\sigma_L^w.p = \sigma_L^w.p _{W \cup L}$	A4.5
$\sigma_L^w.p = \sigma_L^w. p _L$	A4.6
$ p _{D_1} _{D_2} = p _{D_1 \cap D_2}$	A4.7

Table 4.2: Axioms for the dependence scope operator

timed delays, so they impose an empty dependence scope. Axiom A4.4 states that there is no dependence of timed delays that are enabled by an action transition. Axiom A4.5 states that all delays are treated as dependent by default. Axiom A4.6 states that the losers of a timed delay retain their names and that they are treated as dependent in the remaining process. Axiom A4.7 states that multiple scope operators intersect. It enables the replacement of iterative application of scope operators by a single simultaneous one.

First we show that the axioms in Table 4.2 are sound. Afterwards, we derive intermediate normal forms that enable the merger of the racing contexts of timed delay prefixed terms in the alternative composition.

Theorem 4.2.1 *The axioms in Table 4.2 are sound.* □

PROOF We give a racing timed bisimulation relation that relates the left-hand and the right-hand side of every axiom. By $\Delta(p)$ we denote the bisimulation relation satisfying $(\langle p, \alpha_0 \rangle, \langle p, \alpha_0 \rangle, r) \in \Delta(p)$, for some $r \in \mathcal{V} \leftrightarrow \mathcal{V}$.

[A4.1] Define $R = \{(\langle \underline{0}, \alpha_0 \rangle, \langle |\underline{0}|_{\emptyset}, \alpha_0 \rangle, \emptyset)\}$. It is clear that R is a racing timed bisimulation relation as both sides can do nothing.

[A4.2] Define $R = \{(\langle \underline{1}, \alpha_0 \rangle, \langle |\underline{1}|_{\emptyset}, \alpha_0 \rangle, \emptyset)\}$. It is clear that R is a racing timed bisimulation relation as both sides can only terminate.

[A4.3] Define $R = \{(\langle \underline{a}.p, \alpha_0 \rangle, \langle \underline{a}.p|_{\emptyset}, \alpha_0 \rangle, \emptyset)\} \cup \Delta(p)$. It is clear that R is a racing timed bisimulation relation as both sides do only action transitions with a label a to $\langle p|_{\emptyset}, \alpha_0 \rangle$.

[A4.4] Define $R = \{(\langle \underline{a}.p, \alpha_0 \rangle, \langle \underline{a}.p|_{\emptyset}, \alpha_0 \rangle, \emptyset), (\langle p|_{\emptyset}, \alpha_0 \rangle, \langle p|_{\emptyset}, \alpha_0 \rangle, r)\} \cup \Delta(p)$. It is clear that $\langle \underline{a}.p, \alpha_0 \rangle$ and $\langle \underline{a}.p|_{\emptyset}, \alpha_0 \rangle$ can do only action transitions labeled by \underline{a} to $\langle p|_{\emptyset}, \alpha_0 \rangle$ and $\langle p|_{\emptyset}, \alpha_0 \rangle$, respectively. By soundness of A4.7 (see below), we have that R is a racing timed bisimulation relation.

[A4.5] Analogous to A4.3 having in mind that the dependent and independent delays of both states are the same.

[A4.6] Analogous to A4.4.

[A4.7] Define $R = \{(\langle p|_{D_1}|_{D_2}, \alpha_0 \rangle, \langle p|_{D_1 \cap D_2}, \alpha_0 \rangle, r)\} \cup \Delta(p)$. By a direct inspection of the operational rules one concludes that both sides have the same termination options, action, and timed delay transitions. Moreover, the dependent and independent racing delay names are the same. ■

The axioms in Table 4.2 enable manipulation of iterated applications of the dependence scope operator and scopes encompassing action or timed delay prefixed processes. Next, we deal with the alternative composition.

4.3 Alternative Composition

In general, associativity does not hold for the alternative composition. Intuitively, the condition for resolved racing contexts is problematic as it may depend on the order we merge racing contexts of timed delays in incomplete races. The following example illustrates the situation.

Example 4.3.1 Consider the terms $(\sigma_Y^x.\underline{0} + \sigma^z.\underline{0}) + \sigma_X^{Y,Z}.\underline{0}$ and $\sigma_Y^x.\underline{0} + (\sigma^z.\underline{0} + \sigma_X^{Y,Z}.\underline{0})$. The transition scheme of the first term has two outgoing transitions, viz.

$$(\sigma_Y^x.\underline{0} + \sigma^z.\underline{0}) + \sigma_X^{Y,Z}.\underline{0} \xrightarrow{X,Z} \underline{0} + \underline{0} \text{ and } (\sigma_Y^x.\underline{0} + \sigma^z.\underline{0}) + \sigma_X^{Y,Z}.\underline{0} \xrightarrow{Y,Z} \underline{0} + \underline{0}$$

because $(\{X\} \cup \{Z\}) \cap (\{Y\} \cup \emptyset) = \emptyset$ and $\text{rr}(\sigma_Y^{X,Z}, \sigma_X^{Y,Z})$ holds. However, the second process only deadlocks as the timed delay transitions \xrightarrow{X} of $\sigma_Y^x.\underline{0}$ and $\xrightarrow{Y,Z}$ of $\sigma^z.\underline{0} + \sigma_X^{Y,Z}.\underline{0}$ are in inconsistent racing contexts as $\{X\} \cup \{Y\} \neq \{Y, Z\} \cup \{X\}$. □

Nevertheless, associativity holds for terms that comprise alternative composition of action prefixed terms and timed delay prefixed terms that are

already in a context of resolved races. In this case, there is no merging of the timed delays as they are in resolved racing contexts and, therefore, the timed delay transitions are distinctly modeled by the prefixes. Such a term p can be represented in a ‘normal’ form that is unique only for the timed delays modulo commutativity, associativity, and naming of independent delays (see Remark 4.3.2 below), as

$$p = \left| \sum_{i=1}^m \underline{a}_i \cdot p_i + \sum_{j=1}^n \sigma_{L_j}^{W_j} \cdot q_j (+\underline{1})(+\underline{0}) \right|_D,$$

where $W_j \cup L_j = R(p)$ for all $1 \leq j \leq n$ is the set of racing delay names, $D \subseteq R(p)$ determines the dependent delay names, the summand $\underline{1}$ may or may not exist, the summand $\underline{0}$ exists if none of the other summands does, and $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ holds for $1 \leq j < j' \leq n$. The notation $\sum_{i=1}^m p_i$ is shorthand for $p_1 + \dots + p_m$ if $m > 0$, and otherwise the summand does not exist.

It should be clear that $W_j \cup L_j = W_{j'} \cup L_{j'}$ and therefore $W_j \cup L_j = R(p)$ for every $1 \leq j, j' \leq n$. Then, $D(p) = D$ and $I(p) = R(p) \setminus D$.

Remark 4.3.2 Unlike standard head normal forms, e.g. [11, 13], we do not have $\underline{a}_i \cdot p_i \neq \underline{a}_{i'} \cdot p_{i'}$ for $1 \leq i < i' \leq m$, at this point. This is a prerequisite for the uniqueness of the normal form and we discuss it later on. Similarly, associativity still holds if we relax the condition of the timed delay prefixes to require that $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ holds or $(W_j = W_{j'} \text{ and } L_j = L_{j'})$ for $1 \leq j, j' \leq n$, relying on the fact that $\sigma_L^W \cdot p_1 + \sigma_L^W \cdot p_2 \stackrel{\text{tr}}{=} \sigma_L^W \cdot (p_1 + p_2)$. We also note that when restricting to race-complete process specifications that induce only complete races, the associativity of the alternative composition holds (see Section 5.11 and [68]). In this special case, the situation of Example 4.3.1 cannot arise as, in that setting, the timed delays with the remaining resolved racing contexts would also be available. \square

We give the following law for an alternative composition of two terms in a normal form. We refer to it as axiom A4.8.

Theorem 4.3.3 *Let p and p' have the normal forms*

$$p = \left| \sum_{i=1}^m \underline{a}_i \cdot p_i + \sum_{j=1}^n \sigma_{L_j}^{W_j} \cdot q_j (+\underline{1})(+\underline{0}) \right|_D, \quad p' = \left| \sum_{k=1}^{m'} \underline{a}'_k \cdot p'_k + \sum_{\ell=1}^{n'} \sigma_{L'_\ell}^{W'_\ell} \cdot q'_\ell (+\underline{1})(+\underline{0}) \right|_{D'}$$

with $D \subseteq W_j \cup L_j$, $D' \subseteq W'_\ell \cup L'_\ell$, $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ holds for $1 \leq j < j' \leq n$, and $\text{rr}(\sigma_{L'_\ell}^{W'_\ell}, \sigma_{L'_{\ell'}}^{W'_{\ell'}})$ holds for $1 \leq \ell < \ell' \leq n'$. If $I(p) \cap R(p') = R(p) \cap I(p') = \emptyset$,

then the normal form of their alternative composition is given by

$$\begin{aligned}
p + p' = & \left| \sum_{i=1}^m a_i \cdot p_i + \sum_{k=1}^{m'} a'_k \cdot p'_k + \right. \\
& \sum_{j,\ell: (W_j \cup W'_\ell) \cap (L_j \cup L'_\ell) = \emptyset} \sigma_{L_j \cup L'_\ell}^{W_j \cup W'_\ell} \cdot (|q_j|_{L_j} + |q'_\ell|_{L'_\ell}) + \\
& \sum_{j: \text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L'_\ell}^{W'_\ell}) \text{ for all } 1 \leq \ell \leq n'} \sigma_{L_j}^{W_j} \cdot q_j + \\
& \left. \sum_{\ell: \text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L'_\ell}^{W'_\ell}) \text{ for all } 1 \leq j \leq n} \sigma_{L'_\ell}^{W'_\ell} \cdot q_\ell \right) (+ \underline{1}) (+ \underline{0}) \Big|_{D \cup D'} \quad \mathbf{A4.8}
\end{aligned}$$

where the summand $\underline{1}$ exists if p or p' contain it, and $\underline{0}$ exists if none of the other summands does. \square

PROOF The required form of the dependence scope operator is easily obtained by using the axioms A4.1–A4.7 as a rewriting system from left to right. The condition $I(p) \cap R(p') = R(p) \cap I(p') = \emptyset$ ensures that there are no naming conflicts and, thus, enables the consistent merger of the dependence scope operators.

Trivially, $p + p' = \underline{0}$ if $p = p' = \underline{0}$. Also, $p + p'$ deadlocks if p and p' induce inconsistent races, e.g., $\sigma^x \cdot \underline{0} + \sigma^y \cdot \underline{0}$. The state $\langle p + p', \alpha_0 \rangle$ has a termination option if at least one of states $\langle p, \alpha_0 \rangle$ or $\langle p', \alpha_0 \rangle$ have a termination option. The termination option depends on the optional summand $\underline{1}$.

By inspection of the operational rules we have the outgoing transitions of $\langle p, \alpha_0 \rangle$ are $\langle p, \alpha_0 \rangle \xrightarrow{a_i} \langle |\bar{p}_i|_\emptyset, \alpha_0 \rangle$ for $1 \leq i \leq m$ and $\langle p, \alpha_0 \rangle \xrightarrow[|q_j|_{L_j}]{W_j} \langle |\bar{q}_j|_{L_j}, \alpha_j \rangle$ for $1 \leq j \leq n$. Similarly, for $\langle p', \alpha_0 \rangle$ we have $\langle p', \alpha_0 \rangle \xrightarrow{a'_k} \langle |\bar{p}'_k|_\emptyset, \alpha_0 \rangle$ for $1 \leq k \leq m'$ and $\langle p', \alpha_0 \rangle \xrightarrow[|q'_\ell|_{L'_\ell}]{W'_\ell} \langle |\bar{q}'_\ell|_{L'_\ell}, \alpha_\ell \rangle$ for $1 \leq \ell \leq n$. Thus, the outgoing action transitions of the alternative composition are given by the term $\sum_{i=1}^m a_i \cdot p_i + \sum_{k=1}^{m'} a'_k \cdot p'_k$. As the timed delays in p and p' are in the context of resolved races, then they can induce a joint race only with timed delays of the other term. This is expressed by the term $\sum_{j,\ell: (W_j \cup W'_\ell) \cap (L_j \cup L'_\ell) = \emptyset} \sigma_{L_j \cup L'_\ell}^{W_j \cup W'_\ell} \cdot (|q_j|_{L_j} + |q'_\ell|_{L'_\ell})$. Finally, a timed delay is considered to have a resolved racing context in the alternative composition if it is in a resolved race with all timed delays of the other summand. This is expressed by the term $\sum_{j: \text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L'_\ell}^{W'_\ell}) \text{ for all } 1 \leq \ell \leq n'} \sigma_{L_j}^{W_j} \cdot q_j$ when the timed delay in the resolved race originates from p and by the term $\sum_{\ell: \text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L'_\ell}^{W'_\ell}) \text{ for all } 1 \leq j \leq n} \sigma_{L'_\ell}^{W'_\ell} \cdot q_\ell$ when it originates from p' .

To show that $p + p'$ is in normal form, we have to show that the race is resolved for the timed delays. Suppose that the timed delay is in the racing context induced by the winners W and the losers L . First, suppose that $W \neq \emptyset$. Then $(W \cap W_j \neq \emptyset$ and $(W \cup W_j) \cap (L \cup L_j) \neq \emptyset)$ or $(W \cap W_j = \emptyset$, $W \cap L_j \neq \emptyset$, and $L \cap W_j \neq \emptyset)$ or $(W_j = \emptyset$ and $W \cap L_j \neq \emptyset)$ for all $1 \leq j \leq n$ and, similarly, $(W \cap W'_\ell \neq \emptyset$ and $(W \cup W'_\ell) \cap (L \cup L'_\ell) \neq \emptyset)$ or $(W \cap W'_\ell = \emptyset$, $W \cap L'_\ell \neq \emptyset$, and $L \cap W'_\ell \neq \emptyset)$ or $(W'_\ell = \emptyset$ and $W \cap L'_\ell \neq \emptyset)$ for all $1 \leq \ell \leq n'$. Now it should not be difficult to see, by inspecting all possible cases, that the condition $\text{rr}(\sigma_L^W, \sigma_{L_j \cup L'_\ell}^{W_j \cup W'_\ell})$ is fulfilled for any W_j, L_j, W'_ℓ , and L'_ℓ satisfying $(W_j \cup W'_\ell) \cap (L_j \cup L'_\ell) = \emptyset$. For example, suppose that $(W_j = \emptyset$ and $W \cap L_j \neq \emptyset)$ and $(W \cap W'_\ell = \emptyset, W \cap L'_\ell \neq \emptyset, \text{ and } L \cap W'_\ell \neq \emptyset)$. Then $(W \cap (W_j \cup W'_\ell) = W \cap W'_\ell = \emptyset, W \cap (L_j \cup L'_\ell) \supseteq W \cap L'_\ell \neq \emptyset, \text{ and } L \cap (W_j \cup W'_\ell) = L \cap W'_\ell \neq \emptyset)$. In the case when $W = \emptyset$ we have only one possible case, viz. $(W = \emptyset$ and $L \cap W_j \neq \emptyset)$ and $L \cap W'_\ell \neq \emptyset)$. Then clearly $L \cap (W_j \cup W'_\ell) \neq \emptyset$.

Finally, it is not difficult to see that by construction the timed delays are uniquely determined modulo commutativity, associativity, and naming of independent delays, which completes the proof. \blacksquare

Using Theorem 4.3.3 we can represent every term comprising alternative composition of deadlock, termination, action, and timed prefixed terms in a normal form provided there are no naming conflicts of the independent delays. In case there are conflicts, we have to resolve them by renaming the independent delays.

4.4 Renaming of Independent Delays

The following theorem shows how to rename the independent delays in a consistent manner as given by Definition 3.8.2 for the α -conversion. We give the renaming directly on normal forms as for incomplete races it is not always possible to propagate the dependence scope operator in the alternative composition. We give an example to illustrate the situation.

Example 4.4.1 The term $|\sigma_Y^X \cdot 0 + \sigma_X^Y \cdot 0|_X$ cannot be presented as an alternative composition of $|\sigma_Y^X \cdot 0|_D$ and $|\sigma_X^Y \cdot 0|_{D'}$ for any $D, D' \subseteq \mathcal{V}$. It should be clear that $X \in D$ and $X \in D'$ must hold. If $Y \in D$ and $Y \in D'$ then the stochastic delay $[Y]$ will be treated as a dependent delay also in the alternative composition. However, if $Y \notin D$ or $Y \notin D'$, then the resulting term will have two independent delays Y' and Y'' with $F_Y = F_{Y'} = F_{Y''}$. For example, $|\sigma_Y^X \cdot 0|_{X,Y} + |\sigma_X^Y \cdot 0|_X = |\sigma_Y^X \cdot 0 + \sigma_X^{Y'} \cdot 0|_{X,Y}$ with $F_{Y'} = F_Y$. \square

We refer to the equality given by the following theorem as axiom A4.9, which enables renaming of independent delays as given by the α -congruence of Definition 3.8.2.

Theorem 4.4.2 *Let p have the normal form*

$$p = \left| \sum_{i=1}^m a_i \cdot p_i + \sum_{j=1}^n \sigma_{L_j}^{w_j} \cdot q_j \ (+ \underline{1})(+ \underline{0}) \right|_D,$$

where $D \subseteq \mathbb{R}(p) = W_j \cup L_j$ and $\text{rr}(\sigma_{L_j}^{w_j}, \sigma_{L_{j'}}^{w_{j'}})$ holds for $1 \leq j < j' \leq n$. Then the independent racing delay $X \notin D$ can be renamed to Y as follows:

$$p = \left| \sum_{i=1}^m a_i \cdot p_i + \sum_{j: X \notin \mathbb{R}(p)} \sigma_{L_j}^{w_j} \cdot q_j + \sum_{j: X \in W_j} \sigma_{L_j}^{(w_j \setminus \{X\}) \cup \{Y\}} \cdot q_j + \sum_{j: X \in L_j} \sigma_{(L_j \setminus \{X\}) \cup \{Y\}}^{w_j} \cdot q_j[Y/X] \ (+ \underline{1})(+ \underline{0}) \right|_D \quad \mathbf{A4.9},$$

where the optional summands are as for p . □

PROOF We build the bisimulation relation R that relates $p = \left| \sum_{i=1}^m a_i \cdot p_i + \sum_{j=1}^n \sigma_{L_j}^{w_j} \cdot q_j \ (+ \underline{1})(+ \underline{0}) \right|_D$ and $p' = \left| \sum_{i=1}^m a_i \cdot p_i + \sum_{j: X \notin \mathbb{R}(p)} \sigma_{L_j}^{w_j} \cdot q_j + \sum_{j: X \in W_j} \sigma_{L_j}^{(w_j \setminus \{X\}) \cup \{Y\}} \cdot q_j + \sum_{j: X \in L_j} \sigma_{(L_j \setminus \{X\}) \cup \{Y\}}^{w_j} \cdot q_j[Y/X] \ (+ \underline{1})(+ \underline{0}) \right|_D$ inductively by using the racing timed transition scheme of p . All timed delays are in the scope of the same dependence scope operator, so there are no naming conflicts. Also, the timed delays are in the context of resolved races, so the timed delay transitions coincide with the timed delay prefixes. As $X \notin D$, we have that X is an independent delay.

Initially, we put $R = \{(\langle p, \alpha_0 \rangle, \langle p', \alpha_0 \rangle, r\{X \mapsto Y/\{X\}\})\} \cup \Delta(p)$ for r satisfying $(\langle p, \alpha_0 \rangle, \langle p', \alpha_0 \rangle, r) \in \Delta(p)$. For outgoing action transitions and timed delay transitions such that $X \notin W_j \cup L_j$ the resulting states coincide. If $X \in W_j$, then the renaming of X to Y in the timed delay transition of p and p' is captured by $r\{X \mapsto Y/\{X\}\}$ and both transition schemes result in the same state. Now, suppose that there is a state $\langle \bar{p}, \alpha \rangle$ in the transition scheme of p that can be reached by doing timed delay transitions in which X is a loser. Then by the definition of the renaming operation and by inspection of the operational rules it is not difficult to see that there is a state $\langle \bar{p}', \alpha' \rangle$ in the transition scheme of p' that can be reached by taking timed transitions in the same racing context, with the exception that X is replaced by Y . We note that if the process in $\langle \bar{p}, \alpha \rangle$ performs an action or a timed delay transition where X is not a loser, then those parts of the racing

timed transition scheme are the same for both processes, which is covered by $\Delta(p)$. Suppose $(\langle \bar{p}, \alpha \rangle, \langle \bar{p}, \alpha \rangle, r'') \in \Delta(|\sigma_{L \cup \{X\}}^w.p|_D)$. Then we include the triple $(\langle \bar{p}, \alpha \rangle, \langle \bar{p}', \alpha' \rangle, r''\{X \mapsto Y/\{X\}\}) \in R$. As X and Y are dependent delays, we have that r'' is well-defined. By construction R is a bisimulation relation, which completes the proof. ■

We proceed by giving axioms that deal with the encapsulation operator.

4.5 Encapsulation

The encapsulation operator suppresses unwanted action transitions. Unlike the alternative composition, the encapsulation operator does not require resolved races, and it freely propagates through the timed delay prefixes. It is handled using the axioms in Table 4.3.

$\partial_H(\underline{0}) = \underline{0}$	A4.10
$\partial_H(\underline{1}) = \underline{1}$	A4.11
$\partial_H(\underline{a}.p) = \underline{a}.\partial_H(p)$ if $a \notin H$	A4.12
$\partial_H(\underline{a}.p) = \underline{0}$ if $a \in H$	A4.13
$\partial_H(\sigma_L^w.p) = \sigma_L^w.\partial_H(p)$	A4.14
$\partial_H(p _D) = \partial_H(p) _D$	A4.15
$\partial_H(p_1 + p_2) = \partial_H(p_1) + \partial_H(p_2)$	A4.16

Table 4.3: Axioms for the encapsulation operator

Axioms A4.10 and A4.11 deal with the deadlock and successful termination that cannot perform action transitions. If the action prefix should not be suppressed, the encapsulation operator is propagated to the remaining process p as stated in axiom A4.12. In the opposite case, the whole process is turned to deadlock as given by A4.13. Axioms A4.14 and A4.16 state that encapsulation propagates through the timed delay prefixes and the alternative composition as it does not require resolved racing contexts.

Theorem 4.5.1 *The axioms in Table 4.3 are sound.* □

PROOF We give a bisimulation relation that relates the left-hand and the right-hand side of every axiom. By $\Delta(p)$ we denote the bisimulation relation satisfying $(\langle p, \alpha_0 \rangle, \langle p, \alpha_0 \rangle, r) \in \Delta(p)$, for some $r \in \mathcal{V} \leftrightarrow \mathcal{V}$.

[A4.10] Define $R = \{(\langle \partial_H(\underline{0}), \alpha_0 \rangle, \langle \underline{0}, \alpha_0 \rangle, \emptyset)\}$.

[A4.11] Define $R = \{(\langle \partial_H(\underline{1}), \alpha_0 \rangle, \langle \underline{1}, \alpha_0 \rangle, \emptyset)\}$.

[A4.12] Define $R = \{(\langle \partial_H(\underline{a}.p), \alpha_0 \rangle, \langle \underline{a}.\partial_H(p), \alpha_0 \rangle, \emptyset)\} \cup \Delta(\partial_H(p))$. As $a \notin H$ the left-hand state has only one possible transition $\langle \partial_H(\underline{a}.p), \alpha_0 \rangle \xrightarrow{a} \langle \partial_H(|p|_\emptyset), \alpha_0 \rangle$, which is same as the one on the right-hand side, i.e., $\langle \underline{a}.\partial_H(p), \alpha_0 \rangle \xrightarrow{a} \langle \partial_H(|p|_\emptyset), \alpha_0 \rangle$.

[A4.13] Define $R = \{(\langle \partial_H(\underline{a}.p), \alpha_0 \rangle, \langle \underline{0}, \alpha_0 \rangle, \emptyset)\}$. As $a \in H$ the left-hand state has no outgoing transitions.

[A4.14] Define $R = \{(\langle \partial_H(\sigma_L^w.p), \alpha_0 \rangle, \langle \sigma_L^w.\partial_H(p), \alpha_0 \rangle, \text{id}_{W \cup L})\} \cup \Delta(\partial_H(p))$ and proceed analogous to the proof of A4.12.

[A4.15] Define $R = \{(\langle \partial_H(|p|_D), \alpha_0 \rangle, \langle |\partial_H(p)|_D, \alpha_0 \rangle, r)\} \cup \Delta(\partial_H(|p|_D))$. It should be clear that both sides have the same termination options and the same outgoing transitions resulting in the same states.

[A4.16] We define R inductively on the racing timed transition scheme of $\partial_H(p_1 + p_2)$. Initially we put $R = \{(\langle \partial_H(p_1 + p_2), \partial_H(p_1) + \partial_H(p_2), r \rangle, \langle \partial_H(p_1 + p_2), \alpha_0 \rangle, r)\} \cup \Delta(\partial_H(p_1 + p_2))$ for r satisfying $(\langle \partial_H(p_1 + p_2), \alpha_0 \rangle, \langle \partial_H(p_1 + p_2), \alpha_0 \rangle, r) \in \Delta(\partial_H(p_1 + p_2))$. Suppose that a state $\langle \partial_H(p'_1 + p'_2), \alpha' \rangle$ is reached by taking none or more timed delay transitions. By direct inspection of the operational rules, if $\langle p'_1, \alpha' \rangle$ takes an action transition then the resulting state exists in the transition scheme of $\partial_H(p_1 + p_2)$ and this case is covered by $\Delta(\partial_H(p_1 + p_2))$. Similarly, for resolved timed delay transitions. Unresolved timed delays synchronize for both summands, and the resulting state has the form $\langle \partial_H(p''_1 + p''_2), \alpha'' \rangle$. We include the triple $(\langle \partial_H(p''_1 + p''_2), \alpha'' \rangle, \langle \partial_H(p''_1) + \partial_H(p''_2), \alpha'' \rangle, r'')$ in R for r'' satisfying $(\langle \partial_H(p''_1 + p''_2), \alpha'' \rangle, \langle \partial_H(p''_1 + p''_2), \alpha'' \rangle, r'') \in \Delta(\partial_H(p_1 + p_2))$ and proceed with $\langle \partial_H(p''_1 + p''_2), \alpha'' \rangle$. By construction R is a racing timed bisimulation relation. ■

Using the axioms from above it should not be difficult to see the application of the encapsulation operator on normal forms is given by the following corollary.

Corollary 4.5.2 *Let p have the normal form*

$$p = | \sum_{i=1}^m \underline{a}_i.p_i + \sum_{j=1}^n \sigma_{L_j}^{W_j}.q_j (+ \underline{1})(+ \underline{0}) |_D,$$

where $D \subseteq W_j \cup L_j$ and $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ holds for $1 \leq j < j' \leq n$. Then,

$$\partial_H(p) = | \sum_{a_i \notin H} \underline{a}_i.\partial_H(p_i) + \sum_{j=1}^n \sigma_{L_j}^{W_j}.\partial_H(q_j) (+ \underline{1})(+ \underline{0}) |_D,$$

where the optional summands are as for p . \square

PROOF By straightforward application of the axioms in Table 4.3. The normal form is preserved as there are no changes in the timed delay prefixes. \blacksquare

Next, we give an expansion law for the parallel composition.

4.6 Parallel Composition

The expansion law of the parallel composition requires resolved racing contexts. The following example illustrates the matter.

Example 4.6.1 Let $p = (\sigma_x.\underline{0} + \sigma^y.\underline{0}) \parallel \sigma^x.\underline{0}$. By first resolving the race in the left operand and afterwards eliminating the parallel composition according to the operational rules one readily obtains that $p = \sigma_x^y.\underline{0} \parallel \sigma^x.\underline{0} = \underline{0}$. However, if we attempt to naively expand the parallel composition as it is done in the timed process theories, we would wrongly obtain that $p = \sigma_x.\underline{0} \parallel \sigma^x.\underline{0} + \sigma^y.\underline{0} \parallel \sigma^x.\underline{0} = \underline{0} + \sigma^{x,y}.\underline{0} = \sigma^{x,y}.\underline{0}$. \square

The following theorem gives the expansion. The expansion law is referred to as A4.17.

Theorem 4.6.2 *Let p and p' have the normal forms*

$$p = \left| \sum_{i=1}^m \underline{a}_i \cdot p_i + \sum_{j=1}^n \sigma_{L_j}^{w_j} \cdot q_j (+\underline{1})(+\underline{0}) \right|_D, \quad p' = \left| \sum_{k=1}^{m'} \underline{a}'_k \cdot p'_k + \sum_{\ell=1}^{n'} \sigma_{L'_\ell}^{w'_\ell} \cdot q'_\ell (+\underline{1})(+\underline{0}) \right|_{D'}$$

with $D \subseteq W_j \cup L_j$, $D' \subseteq W'_\ell \cup L'_\ell$, $\text{rr}(\sigma_{L_j}^{w_j}, \sigma_{L_{j'}}^{w_{j'}})$ for $1 \leq j < j' \leq n$, and $\text{rr}(\sigma_{L'_\ell}^{w'_\ell}, \sigma_{L'_{\ell'}}^{w'_{\ell'}})$ for $1 \leq \ell < \ell' \leq n'$. If $\text{I}(p) \cap \text{R}(p') = \text{R}(p) \cap \text{I}(p') = \emptyset$, then the normal form of their parallel composition is given by $p \parallel p' =$

$$\begin{aligned} & \left| \sum_{i=1}^m \underline{a}_i \cdot (|p_i|_\emptyset \parallel p') + \sum_{k=1}^{m'} \underline{a}'_k \cdot (p \parallel |p'_k|_\emptyset) + \right. \\ & \quad \sum_{i,k: \gamma(a_i, a'_k) = b_{ik}} \underline{b}_{ik} \cdot (|p_i|_\emptyset \parallel |p'_k|_\emptyset) + \\ & \quad \left. \sum_{j,\ell: (W_j \cup W'_\ell) \cap (L_j \cup L'_\ell) = \emptyset} \sigma_{L_j \cup L'_\ell}^{w_j \cup w'_\ell} \cdot (|q_j|_{L_j} \parallel |q'_\ell|_{L'_\ell}) (+\underline{1})(+\underline{0}) \right|_{D \cup D'} \quad \mathbf{A4.17}, \end{aligned}$$

where the summand $\underline{1}$ exists only if it exists in both p and p' and the summand $\underline{0}$ exists if none of the other summands does. \square

PROOF The first three summands of the parallel composition are directly derivable from the structural operational semantics of the action prefix operator. Also it should be clear that the parallel composition has a termination option only if both components have a termination option. As both terms are in normal form the stochastic delays from one component can only race with the stochastic delays of the other component. The condition $I(p) \cap R(p') = R(p) \cap I(p') = \emptyset$ ensures that there are no naming conflicts. If it is not fulfilled, then we use the α -conversion law A4.9 to rename the conflicting independent racing delay names. The last summand captures the synchronized timed delays when both delays can delay together without any racing conflicts. Again, uniqueness of the timed delays modulo commutativity, associativity, and naming of independent delays follows by construction, which completes the proof. ■

Unlike alternative composition, parallel composition is associative for closed TCP^{drst} terms. This is an important property that supports compositional modeling. Intuitively, parallel composition is associative as it does not allow for resolved races that obstructed the associativity of the alternative composition. This is captured in the following theorem.

Theorem 4.6.3 *Parallel composition is associative, i.e., $(p \parallel p') \parallel p'' = p \parallel (p' \parallel p'')$ for all $p, p', p'' \in \mathcal{C}(\text{TCP}^{\text{drst}})$.* □

PROOF Let p, p' , and p'' have the normal forms

$$\begin{aligned} p &= \left| \sum_{i=1}^m \underline{a}_i \cdot p_i + \sum_{j=1}^n \sigma_{L_j}^{W_j} \cdot q_j (+ \underline{1})(+ \underline{0}) \right|_D, \\ p' &= \left| \sum_{k=1}^{m'} \underline{a}'_k \cdot p'_k + \sum_{\ell=1}^{n'} \sigma_{L'_\ell}^{W'_\ell} \cdot q'_\ell (+ \underline{1})(+ \underline{0}) \right|_{D'}, \\ p'' &= \left| \sum_{r=1}^{m''} \underline{a}''_r \cdot p''_r + \sum_{s=1}^{n''} \sigma_{L''_s}^{W''_s} \cdot q''_s (+ \underline{1})(+ \underline{0}) \right|_{D''} \end{aligned}$$

with $D \subseteq W_j \cup L_j$, $D' \subseteq W'_\ell \cup L'_\ell$, $D'' \subseteq W''_s \cup L''_s$, $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ for $1 \leq j < j' \leq n$, $\text{rr}(\sigma_{L'_\ell}^{W'_\ell}, \sigma_{L'_{\ell'}}^{W'_{\ell'}})$ for $1 \leq \ell < \ell' \leq n'$, and $\text{rr}(\sigma_{L''_s}^{W''_s}, \sigma_{L''_{s'}}^{W''_{s'}})$ for $1 \leq s < s' \leq n''$. Without any loss of generality, we assume that $I(p) \cap R(p') = R(p) \cap I(p') = \emptyset$, $I(p') \cap R(p'') = R(p') \cap I(p'') = \emptyset$, and $I(p) \cap R(p'') = R(p) \cap I(p'') = \emptyset$. In the opposite case, one can always use the α -conversion law A4.9 of Theorem 4.4.2 to rename conflicting independent delay names. We prove the claim by

total induction on the length of the terms. The initial cases are trivially satisfied as the parallel composition has a termination option only when all components have a termination option.

One calculates for $(p \parallel p') \parallel p''$ using Theorem 4.6.2 in the first and second step of the derivation, the induction hypothesis and associativity of the synchronization function and the union set operation in the third step, and Theorem 4.6.2 in the reverse direction in the last step:

$$\begin{aligned}
& (p \parallel p') \parallel p'' \\
= & \left| \sum_{i=1}^m \underline{a}_i \cdot (|p_i|_\emptyset \parallel p') + \sum_{k=1}^{m'} \underline{a}'_k \cdot (p \parallel |p'_k|_\emptyset) + \sum_{i,k: \gamma(a_i, a'_k) = b_{ik}} \underline{b}_{ik} \cdot (|p_i|_\emptyset \parallel |p'_k|_\emptyset) + \right. \\
& \left. \sum_{j,\ell: (W_j \cup W'_\ell) \cap (L_j \cup L'_\ell) = \emptyset} \sigma_{L_j \cup L'_\ell}^{W_j \cup W'_\ell} \cdot (|q_j|_{L_j} \parallel |q'_\ell|_{L'_\ell}) (+ \perp)(+ \perp) \right|_{D \cup D'} \parallel p'' \\
= & \left| \sum_{i=1}^m \underline{a}_i \cdot ((|p_i|_\emptyset \parallel p') \parallel p'') + \sum_{k=1}^{m'} \underline{a}'_k \cdot (p \parallel |p'_k|_\emptyset \parallel p'') + \sum_{r=1}^{m''} \underline{a}''_r \cdot ((p \parallel p') \parallel |p''_r|_\emptyset) + \right. \\
& \sum_{i,k: \gamma(a_i, a'_k) = b_{ik}} \underline{b}_{ik} \cdot ((|p_i|_\emptyset \parallel |p'_k|_\emptyset) \parallel p'') + \sum_{i,r: \gamma(a_i, a''_r) = b'_{ir}} \underline{b}'_{ir} \cdot ((|p_i|_\emptyset \parallel p') \parallel |p''_r|_\emptyset) + \\
& \sum_{k,r: \gamma(a'_k, a''_r) = b''_{kr}} \underline{b}''_{kr} \cdot (p \parallel |p'_k|_\emptyset \parallel |p''_r|_\emptyset) + \\
& \sum_{i,k,r: \gamma(\gamma(a_i, a'_k), a''_r) = c_{ikr}} \underline{c}_{ikr} \cdot ((|p_i|_\emptyset \parallel |p'_k|_\emptyset) \parallel |p''_r|_\emptyset) + \\
& \left. \sum_{j,\ell,s: ((W_j \cup W'_\ell) \cup W''_s) \cap ((L_j \cup L'_\ell) \cup L''_s) = \emptyset} \sigma_{(L_j \cup L'_\ell) \cup L''_s}^{(W_j \cup W'_\ell) \cup W''_s} \cdot (|q_j|_{L_j} \parallel |q'_\ell|_{L'_\ell} \parallel |q''_s|_{L''_s}) \right. \\
& \left. (+ \perp)(+ \perp) \right|_{(D \cup D') \cup D''} \\
= & \left| \sum_{i=1}^m \underline{a}_i \cdot (|p_i|_\emptyset \parallel (p' \parallel p'')) + \sum_{k=1}^{m'} \underline{a}'_k \cdot (p \parallel (|p'_k|_\emptyset \parallel p'')) + \sum_{r=1}^{m''} \underline{a}''_r \cdot (p \parallel (p' \parallel |p''_r|_\emptyset)) + \right. \\
& \sum_{i,k: \gamma(a_i, a'_k) = b_{ik}} \underline{b}_{ik} \cdot (|p_i|_\emptyset \parallel (|p'_k|_\emptyset \parallel p'')) + \sum_{i,r: \gamma(a_i, a''_r) = b'_{ir}} \underline{b}'_{ir} \cdot (|p_i|_\emptyset \parallel (p' \parallel |p''_r|_\emptyset)) + \\
& \sum_{k,r: \gamma(a'_k, a''_r) = b''_{kr}} \underline{b}''_{kr} \cdot (p \parallel (|p'_k|_\emptyset \parallel |p''_r|_\emptyset)) + \\
& \sum_{i,k,r: \gamma(a_i, \gamma(a'_k, a''_r)) = c_{ikr}} \underline{c}_{ikr} \cdot (|p_i|_\emptyset \parallel (|p'_k|_\emptyset \parallel |p''_r|_\emptyset)) + \\
& \left. \sum_{j,\ell,s: (W_j \cup (W'_\ell \cup W''_s)) \cap (L_j \cup (L'_\ell \cup L''_s)) = \emptyset} \sigma_{L_j \cup (L'_\ell \cup L''_s)}^{W_j \cup (W'_\ell \cup W''_s)} \cdot (|q_j|_{L_j} \parallel (|q'_\ell|_{L'_\ell} \parallel |q''_s|_{L''_s})) \right. \\
& \left. (+ \perp)(+ \perp) \right|_{D \cup (D' \cup D'')} \\
= & p \parallel (p' \parallel p''),
\end{aligned}$$

which completes the proof. ■

We continue with the resolution of the maximal progress operator.

4.7 Maximal Progress

The typical resolution of the maximal progress operator in timed process theory requires an additional operator that ascertains that a process has no timed delays transitions [11]. Alternatively, one can use normal forms that make the undelayable action transitions and the timed delay transitions explicit. For example, $\theta_a(\underline{a}.p_1 + \underline{b}.p_2 + \sigma.p_3) = \theta_a(\underline{a}.p_1) + \theta_a(\underline{b}.p_2)$ because the action a is prioritized over passage of time. Note that the maximal progress does not prioritize actions, so the second summand prefixed by the undelayable action b remains.

Unlike alternative and parallel composition, the resolution of maximal progress does not require resolved races. Nevertheless, for the sake of compactness we give a law on the existing normal forms without introducing an additional more relaxed type of normal form.

Theorem 4.7.1 *Let p have the normal form*

$$p = \left| \sum_{i=1}^m \underline{a}_i.p_i + \sum_{j=1}^n \sigma_{L_j}^{W_j}.q_j \right|_{(+\underline{1})(+\underline{0})|_D}$$

where $D \subseteq W_j \cup L_j$ and $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ holds for $1 \leq j < j' \leq n$. Then,

$$\theta_I(p) = \left| \sum_{i=1}^m \underline{a}_i.\theta_I(p_i) \right|_{(+\underline{1})(+\underline{0})|_D} \quad \text{if } a_i \in I \text{ for some } i \in \{1, \dots, m\} \quad \mathbf{A4.18}$$

$$\theta_I(p) = \left| \sum_{i=1}^m \underline{a}_i.\theta_I(p_i) + \sum_{j=1}^n \sigma_{L_j}^{W_j}.\theta_I(q_j) \right|_{(+\underline{1})(+\underline{0})|_D} \quad \text{if } \bigcup_{i=1}^m \{a_i\} \cap I = \emptyset \quad \mathbf{A4.19},$$

where the optional summands are as for p . □

PROOF It should be clear that when at least one enabled action transitions has priority then the stochastic delay transitions are no longer available. In the opposite case, the maximal progress operator propagates through the timed delay prefix as given by the operational rules. The normal form is preserved as there are no changes in the timed delay prefixes. ■

Now that we provided expansion laws for all operators, we proceed by giving head normal forms for closed TCP^{drst} terms that support the further development of the theory.

4.8 Head Normal Form

Using the axioms/expansion laws for every operator, it should not be difficult to see that every closed TCP^{drst} term can be represented in the normal form used in the previous derivations. To eliminate multiple instances of bisimilar action prefixed terms in alternative composition we introduce an additional axiom:

$$\underline{a}.p + \underline{a}.p = \underline{a}.p \quad \mathbf{A4.20.}$$

It gives idempotence of action prefixed terms in the alternative composition. It should be clear that axiom A4.20 is sound as $\langle \underline{a}.p + \underline{a}.p, \alpha \rangle \xrightarrow{a} \langle p|_{\emptyset}, \alpha_0 \rangle$ and $\langle \underline{a}.p, \alpha \rangle \xrightarrow{a} \langle p|_{\emptyset}, \alpha_0 \rangle$ and no other transitions are possible. It enables unique normal forms as discussed above in Remark 4.3.2. We proceed by giving a head normal form that is unique modulo commutativity, associativity, and naming of independent delays.

Corollary 4.8.1 *Every closed term $p \in \mathcal{C}(\text{TCP}^{\text{drst}})$ can be represented in a unique head normal form modulo commutativity, associativity, and naming of independent delays, viz.*

$$p = \left| \sum_{i=1}^m \underline{a}_i.p_i + \sum_{j=1}^n \sigma_{L_j}^{W_j}.q_j \left(+ \underline{\quad} \right) \left(+ \underline{\quad} \right) \right|_D$$

with $\underline{a}_i.p_i \neq \underline{a}_{i'}.p_{i'}$ for $1 \leq i < i' \leq m$, $D \subseteq \text{R}(p) = W_j \cup L_j$, $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ holds for $1 \leq j < j' \leq n$, the summand $\underline{\quad}$ is optional, and the summand $\underline{\quad}$ exists if none of the other summands does. \square

PROOF By the axioms A4.1 – A4.7 in Table 4.2 for manipulation with the dependence scope operator, the expansion law A4.8 of the alternative composition of Theorem 4.3.3, the α -conversion law A4.9 for renaming of independent delays of Theorem 4.4.2, axioms A4.10 – A4.16 in Table 4.3 that deal with the encapsulation operator, the expansion law A4.17 of the parallel composition of Theorem 4.6.2, the expansion laws A4.18 and A4.19 of the maximal progress of Theorem 4.7.1 every closed TCP^{drst} term can be reduced to the temporary normal form that is unique only for timed delays modulo commutativity, associativity, and naming of independent delays. By using axiom A4.20 for idempotence of the action prefixed terms in the alternative composition as a rewriting rule from left to right, we also obtain uniqueness for the action prefixed terms modulo commutativity and associativity. \blacksquare

The availability of a head normal form is technically important. It is instrumental for proving ground-completeness and showing uniqueness of solutions of guarded recursive specifications in the term model [9].

4.9 Ground Completeness

As every term can be reduced to the head normal form given by Corollary 4.8.1, which makes all transitions explicit, it should come as no surprise that the equations given in this section form a ground-complete theory.

Theorem 4.9.1 *Axioms A4.1 – A4.7 in Table 4.2, the α -conversion law A4.9, axioms A4.10 – A4.16 in Table 4.3, the expansion laws A4.8, A4.17–A4.19, and axiom A4.20 are ground-complete for the term model $\mathbb{P}(\text{TCP}^{\text{drst}})/\simeq_t$. \square*

PROOF The theorem is proven by natural induction on the total number of symbols in $q, q' \in \mathcal{C}(\text{TCP}^{\text{drst}})$. The base case is when q and q' are either $\underline{0}$ or $\underline{1}$. Trivially $\underline{0} = \underline{0}$ and $\underline{1} = \underline{1}$. Suppose that the total number of symbols is s and $q \simeq_t q'$. By Corollary 4.8.1 we have that the head normal forms p and p' of q and q' are given by $p \simeq_t q$ and $p' \simeq_t q'$:

$$p = \left| \sum_{i=1}^m \underline{a}_i \cdot p_i + \sum_{j=1}^n \sigma_{L_j}^{W_j} \cdot q_j (+\underline{1})(+\underline{0}) \right|_D, \quad p' = \left| \sum_{k=1}^{m'} \underline{a}'_k \cdot p'_k + \sum_{\ell=1}^{n'} \sigma_{L'_\ell}^{W'_\ell} \cdot q'_\ell (+\underline{1})(+\underline{0}) \right|_{D'}$$

with $\underline{a}_i \cdot p_i \neq \underline{a}_{i'} \cdot p_{i'}$ for $1 \leq i < i' \leq m$, $\underline{a}'_k \cdot p'_k \neq \underline{a}'_{k'} \cdot p'_{k'}$ for $1 \leq k < k' \leq m'$, $D \subseteq \mathbb{R}(p) = W_j \cup L_j$, $D' \subseteq \mathbb{R}(p') = W'_\ell \cup L'_\ell$, $\text{rr}(\sigma_{L_j}^{W_j}, \sigma_{L_{j'}}^{W_{j'}})$ for $1 \leq j < j' \leq n$, and $\text{rr}(\sigma_{L'_\ell}^{W'_\ell}, \sigma_{L'_{\ell'}}^{W'_{\ell'}})$ for $1 \leq \ell < \ell' \leq n'$.

From $q \simeq_t q'$ and Theorem 3.3.3, stating that the bisimulation relation is an equivalence, it immediately follows that $p \simeq_t p'$. Then there exists a bisimulation relation R , such that $(\langle p, \alpha \rangle, \langle p', \alpha' \rangle, \mathbf{r}) \in R$ for some bijection \mathbf{r} . Note that we use an arbitrary environment instead of the zero environment because of the inductive step, which is allowed by Lemma 3.7.2.

If $p \downarrow$ then it must be that $p' \downarrow$ and vice versa, so p contains an $\underline{1}$ summand if and only if p' contains a $\underline{1}$ summand.

Suppose that $\langle p, \alpha \rangle \xrightarrow{a} \langle |\bar{p}|_\emptyset, \alpha_0 \rangle$. Then it must be also that $\langle p', \alpha' \rangle \xrightarrow{a} \langle |\bar{p}'|_\emptyset, \alpha_0 \rangle$, where $\langle |\bar{p}|_\emptyset, \alpha_0 \rangle \simeq_t \langle |\bar{p}'|_\emptyset, \alpha_0 \rangle$ and vice versa. Suppose $a_j = a = a'_\ell$ for some $1 \leq j \leq m$ and $1 \leq \ell \leq m'$. Then from the hypothesis it follows that $p_j = p'_\ell$ as $p_j \simeq_t p'_\ell$. Because of the idempotence of action prefixed terms in the alternative composition, the correspondence must be one-to-one, so

we have that $m = m'$. Moreover, the summands can be renumbered such that $\underline{a}_i.p_i = \underline{a}'_i.p'_i$ for $1 \leq i \leq m$.

As the dependent delays must be identical in bisimilar terms, it follows that $D = D'$. In the normal form all races are resolved, so it is not possible to merge the timed delay transitions. Thus, to every timed delay of p , there corresponds exactly one timed delay from p' as the relation between the delays is given by the bijection r . Thus, $n = n'$ and there must be one-to-one correspondence between the timed delay transitions. The dependent delays are identical, so only the independent delays can be guided by variables with different names. However, one can use the α -conversion law A4.9 to rename this delays, such that r becomes an identity bijection. Thus, we can renumber the summands such that $W_j = W'_j$, $L_j = L'_j$, and $q_j = q'_j$ for $1 \leq j \leq n$, which completes the proof. ■

We proceed by introducing guarded recursion in the process theory, which enables the introduction of delayable actions and stochastic delay prefixes.

4.10 Guarded Recursive Specifications

We introduce guarded recursion in the process theory TCP^{drst} by means of guarded recursive specifications obtaining the process theory $\text{TCP}_{\text{rec}}^{\text{drst}}$. Guardedness is a well-known concept that typically guarantees unique solution of the recursive specifications. The prerequisite is that every recursion variable must be prefixed, which ensures well-defined (predictable) behavior of the process.

A guarded recursive equation is an equation of the form $A = p$, where $A \in \mathcal{R}$ is a recursion variable, and p is a term over the signature of TCP^{drst} that additionally contains variables from \mathcal{R} . Moreover, the term can be rewritten in such a way that the variables only appear in subterms prefixed by $\underline{a}.$ or $\sigma_L^w.$ for $a \in \mathcal{A}$ and $W, L \in \mathcal{V}$ provided that $W \cap L = \emptyset$. A guarded recursive specification $S \in \mathcal{G}$, \mathcal{G} denoting the set of guarded recursive specifications of our interest, is a set of guarded recursive equations with one equation for every variable. The set of recursion variables of a specification S is denoted by $\mathcal{R}(S)$.

The definitions of dependent racing, independent racing, dependence binding, and newly enabled independent delay names are straightforwardly extended to guarded recursive specifications as $I(A) = I(p)$, $D(A) = D(p)$, $B(A) = B(p)$, and $N(A) = N(p)$, respectively, assuming that $A = p$. For the renaming of delays we have that $A[X/Y] = A'$, where $A' = p[X/Y]$ provided

$$\begin{aligned}
\mu\mathbf{0}.S &= \mathbf{0} \\
\mu\mathbf{1}.S &= \mathbf{1} \\
\mu(\underline{a}.p).S &= \underline{a}.(\mu p.S) \\
\mu(\sigma_L^w.p).S &= \sigma_L^w.(\mu p.S) \\
\mu(\mu A.S).S &= \mu A.S \\
\mu(\partial_H(p)).S &= \partial_H(\mu p.S) \\
\mu(\theta_I(p)).S &= \theta_I(\mu p.S) \\
\mu(p_1 + p_2).S &= \mu p_1.S + \mu p_2.S \\
\mu(p_1 \parallel p_2).S &= \mu p_1.S \parallel \mu p_2.S
\end{aligned}$$

Table 4.4: Definition of $\mu p.S$

that $A = p$. For the notion of α -conversion we have:

$$\text{ccr}_{d,i}(A_1, E_1, A_2, E_2) \text{ if } \text{ccr}_{d,i}(p_1, E_1, p_2, E_2) \text{ for } A_1 = p_1 \text{ and } A_2 = p_2.$$

Solutions of recursive specifications in the term model are process terms that when replaced for the recursion variables give valid equations in the term model. By the constant $\mu A.S$ we denote a process term that is a solution for the recursion variable $A \in \mathcal{R}(S)$ defined by the guarded recursive specification S . Typically, a solution of a single variable is of interest, which we refer to as the solution of the guarded recursive specification. We extend the signature of $\mathbb{P}(\text{TCP}^{\text{drst}})$ with the constants $\mu A.S$ that are of our interest for $A \in \mathcal{R}(S)$ and $S \in \mathcal{G}$. The structural operational semantics is given in Table 4.5.

We can generalize the notation $\mu A.S$ to $\mu p.S$, for an arbitrary term $p \in \mathcal{C}(\text{TCP}_{\text{rec}}^{\text{drst}})$ that contains variables from $\mathcal{R}(S)$. The definition of $\mu p.S$ is given using structural induction in Table 4.4. It is supported by the operational semantics in Table 4.5.

It is straightforward from the structural operational semantics that every equation of some guarded recursive specification has a solution. Thus, the restrictive recursive definition principle, abbreviated as RDP^- , that every guarded recursive specification has a solution is sound in $\text{TCP}_{\text{rec}}^{\text{drst}}$. Also, it should come as no surprise that the bisimulation relation is a congruence for recursion and that the axioms and expansion laws are sound for $\mathbb{P}(\text{TCP}_{\text{rec}}^{\text{drst}})$. Thus, we have the following term model for $\text{TCP}_{\text{rec}}^{\text{drst}}$.

$$\begin{array}{c}
\text{4.1} \frac{\langle \mu p.S, \alpha \rangle \downarrow, A = p \in S}{\langle \mu A.S, \alpha \rangle \downarrow} \qquad \text{4.2} \frac{\langle \mu p.S, \alpha \rangle \xrightarrow{a} \langle p', \alpha' \rangle, A = p \in S}{\langle \mu A.S, \alpha \rangle \xrightarrow{a} \langle p', \alpha' \rangle} \\
\text{4.3} \frac{\langle \mu p.S, \alpha \rangle \xrightarrow{W}_L \langle p', \alpha' \rangle, A = p \in S}{\langle \mu A.S, \alpha \rangle \xrightarrow{W}_L \langle p', \alpha' \rangle}
\end{array}$$

Table 4.5: Operational rules for guarded recursion

Definition 4.10.1 The term model of $\text{TCP}_{\text{rec}}^{\text{drst}}$ is the quotient algebra $\mathbb{P}(\text{TCP}_{\text{rec}}^{\text{drst}})/\simeq_t$ for $\mathbb{P}(\text{TCP}_{\text{rec}}^{\text{drst}}) = (\mathcal{C}(\text{TCP}_{\text{rec}}^{\text{drst}}), \underline{0}, \underline{1}, \mu A.S$ for $S \in \mathcal{G}$ and $A \in \mathcal{R}(S), \underline{a}.$ for $a \in \mathcal{A}, \sigma_L^W.$ for $W, L \subseteq \mathcal{V}$ satisfying $W \cap L = \emptyset, |_{-}|_D$ for $D \subseteq \mathcal{V}, \partial_H(-)$ for $H \subseteq \mathcal{A}, \theta_I(-)$ for $I \subseteq \mathcal{A}, - + -, - \parallel -)$. \square

It is readily observed that $\text{TCP}_{\text{rec}}^{\text{drst}}$ is a conservative extension of TCP^{drst} [11, 8]. Additionally, it is not difficult to show that the head normal form of Corollary 4.8.1 is preserved. Now, by an adaptation of the proofs of [9] along the lines of [8] it can be shown that the recursive specification principle holds, relying on the existence of the head normal form. This principle, abbreviated as RSP, states that every guarded recursive specification has at most one solution in the model. As a consequence of the validity of the principles RDP^- and RSP in the model, all guarded recursive specifications have a unique solution in $\mathbb{P}(\text{TCP}_{\text{rec}}^{\text{drst}})/\simeq_t$.

4.11 Summary

We develop a sound and ground-complete equational theory for TCP^{drst} . The alternative composition is not associative, so we resort to normal forms that make the race condition explicit in order to provide for expansion laws. We also introduce guarded recursion by means of guarded recursive specifications. The guardedness assures unique solutions of the equations.

In the following section we employ guarded recursive specifications to embed delayable actions and stochastic delays into the theory.

Chapter 5

Process Theory $\text{DTCP}_{\text{rec}}^{\text{dst}}$

In this chapter we derive delayable action and stochastic delay prefixes by means of guarded recursive specifications comprising undelayable actions and timed delays as hinted in Section 2.4. The theory builds on the process theory TCP^{drst} , set up in Chapters 3 and 4. Afterwards, we analyze process specifications that comprise them. We will show that when dealing with such process specifications, we need not to resort to the specifications that comprise timed delay prefixes, but we can manipulate with the higher-order constructions directly. This gives rise to the ground-complete derived theory of communicating process with discrete stochastic time – $\text{DTCP}_{\text{rec}}^{\text{dst}}(\mathcal{A}, \mathcal{V}, \mathcal{R}, \gamma)$. We illustrate the approach by modeling and solving the $G/G/1/\infty$ queue as an example.

5.1 Delayable Action Prefix and Delayable Deadlock

We define the delayable action prefix scheme $\bar{a}.$ for $a \in \mathcal{A}$ by taking the approach of [11] and putting

$$\bar{a}.p = \mu A. \{A = \underline{a}.p + \sigma.A\}.$$

This process allows for the undelayable action \underline{a} at every time slice. If the action is taken, then the process continues to behave as p and, otherwise, the process is delayed one unit of time. As the semantics of the processes is given per time unit, the process captures the intuition of a delayable action.

Of interest is the application of the encapsulation and the maximal progress operator on the delayable action prefix. For the encapsulation one obtains

$$\partial_a(A) = \partial_a(\underline{a}.p + \sigma.A) = \underline{0} + \sigma.\partial_a(A) = \sigma.\partial_a(A).$$

So, the resulting process, can only delay arbitrary long. From the discussion on stochastic delays above, it should be clear that this process is not a stochastic delay as there are no winners. However, it plays a role in the theory as it occurs as an encapsulation of a delayable action. We represent this process in the theory as the constant process $\bar{0}$ called delayable deadlock where

$$\bar{0} = \mu B. \{B = \sigma.B\}.$$

It is a neutral element in the alternative composition for a delayable action. To see this, assume that the definitions of $\bar{a}.p$ and $\bar{0}$ are as above. Then for $\bar{a}.p + \bar{0}$ we have that

$$A + B = (\underline{a}.p + \sigma.A) + \sigma.B = \underline{a}.p + \sigma.(A + B),$$

i.e., $\bar{a}.p + \bar{0} = \bar{a}.p$.

Remark 5.1.1 We can also define a delayable termination process constant as $\bar{1} = \mu C. \{C = \underline{1} + \sigma.C\}$. It is a neutral element for the parallel composition. However, for the sake of clarity it is not included in the process algebra presented in this chapter. \square

For the application of the maximal progress we have

$$\theta_a(A) = \theta_a(\underline{a}.p + \sigma.A) = \underline{a}.\theta_a(p),$$

i.e., its application turns a delayable action prefix into an undelayable one.

Next, we analyze the interaction between undelayable action, delayable action, and stochastic delay prefixes.

5.2 Stochastic Delay Prefix

We specify stochastic delays as suggested in Section 2.4, i.e., as an expiration observed per unit of time in the same racing context.

Definition 5.2.1 The stochastic delay prefix ${}^W_L.p$ is defined as the solution of the following guarded recursive equation

$${}^W_L.p = \mu A. \{A = \sigma_L^W.p + \sigma_{W \cup L}.A\}. \quad \square$$

The solution of this guarded recursive specification is an infinite racing timed transition scheme. The ‘paths’ in the probabilistic timed transition system induced by this scheme that end in p represent the duration of the stochastic delay. The process is well-defined as the probability that a path of infinite

length is taken in the probabilistic timed transition system that is induced by some assignment of distributions is equal to zero. This is because the probability distributions of the racing delays are aged by 1 in every state by the expiration of the timed delay $\sigma_{w \cup L}$ from above and $\lim_{n \rightarrow \infty} F(n) = 1$ for every $F \in \mathcal{F}$.

We illustrate by means of an example how to specify the desired stochastic behavior in this fashion.

Example 5.2.2 Let

$$\begin{aligned} p_1 &= [X].p + [Y].q \text{ and} \\ p_2 &= [\bar{Y}].(|p|_\emptyset + [Y].q) + [X, Y].(p + q) + [\bar{X}].([X].p + |q|_\emptyset). \end{aligned}$$

We put $[X].p = \mu A_1.S$ for $A_1 = \sigma^x.p + \sigma_{x'}A_1 \in S$ and $[Y].q = \mu A_2.S$ for $A_2 = \sigma^y.q + \sigma_{y'}A_2 \in S$.

Let us put $[\bar{Y}].(|p|_\emptyset + [Y].q) = \mu A_3.S$, $[X, Y].(p + q) = \mu A_4.S$, and $[\bar{X}].([X].p + |q|_\emptyset) = \mu A_5.S$. Then,

$$\begin{aligned} S = \{ & A_1 = \sigma^x.p + \sigma_{x'}A_1, \\ & A_2 = \sigma^y.q + \sigma_{y'}A_2, \\ & A_3 = \sigma_{y'}^x.(|p|_\emptyset + A_2) + \sigma_{x, y'}A_3, \\ & A_4 = \sigma^{x, y}.(p + q) + \sigma_{x, y'}A_4, \\ & A_5 = \sigma_{x'}^y.(A_1 + |q|_\emptyset) + \sigma_{x, y'}A_5 \}. \end{aligned}$$

Now, we can write $p_1 = \mu(A_1 + A_2).S$ and $p_2 = \mu(A_3 + A_4 + A_5).S$. By using the expansion law A4.8 for the alternative compositions $A_1 + A_2$ and $A_3 + A_4 + A_5$, one calculates:

$$\begin{aligned} & A_1 + A_2 \\ = & (\sigma^x.p + \sigma_{x'}A_1) + (\sigma^y.q + \sigma_{y'}A_2) \\ = & \sigma^{x, y}.(p + q) + \sigma_{y'}^x.(|p|_\emptyset + A_2) + \sigma_{x'}^y.(A_1 + |q|_\emptyset) + \sigma_{x, y'}(A_1 + A_2) \\ & A_3 + A_4 + A_5 \\ = & (\sigma_{y'}^x.(|p|_\emptyset + A_2) + \sigma_{x, y'}A_3) + (\sigma^{x, y}.(p + q) + \sigma_{x, y'}A_4) + \\ & (\sigma_{x'}^y.(A_1 + |q|_\emptyset) + \sigma_{x, y'}A_5) \\ = & \sigma^{x, y}.(p + q) + \sigma_{y'}^x.(|p|_\emptyset + A_2) + \sigma_{x'}^y.(A_1 + |q|_\emptyset) + \sigma_{x, y'}(A_3 + A_4 + A_5) \end{aligned}$$

Now, by following the principles RDP⁻ and RSP for the solutions of guarded recursive specifications, p_1 and p_2 have the same solution. \square

Example 5.2.2 shows how to manipulate with stochastic delays by using guarded recursive specifications. However, we note that p_1 and p_2 do not

specify explicitly any recursive equations and use only a stochastic delay prefix of the form $[^W_L]_{\cdot}$ for $W, L \subseteq \mathcal{V}$ with $W \neq \emptyset$ and $W \cap L = \emptyset$. Actually, we can manipulate stochastic delay prefixed terms directly in any context without having to resort to the recursive specifications at all (as originally proposed in [69, 68]).

However, the interaction between timed and stochastic delays generally requires the representation of the stochastic delays in terms of the guarded recursive specifications. We give a simple example of the interaction between stochastic and timed delay prefixes.

Example 5.2.3 We consider the alternative composition $\theta_I(\sigma^3.\underline{a}.p + [^W_L]_{\cdot}\underline{b}.q)$ for $I = \{a, b\}$. Let $[^W_L]_{\cdot}q = \mu B.\{B = \sigma_L^w.\underline{b}.q + \sigma_{W \cup L}.B\}$. Then

$$\begin{aligned}
& \theta_I(\sigma^3.\underline{a}.p + B) \\
&= \theta_I(\sigma.\sigma^2.\underline{a}.p + (\sigma_L^w.\underline{b}.q + \sigma_{W \cup L}.B)) \\
&= \theta_I(\sigma_L^w.(\underline{b}.q + \sigma^2.\underline{a}.p) + \sigma_{W \cup L}.(\sigma^2.\underline{a}.p + B)) \\
&= \sigma_L^w.\theta_I(\underline{b}.q + \sigma^2.\underline{a}.p) + \sigma_{W \cup L}.\theta_I(\sigma.\sigma.\underline{a}.p + \sigma_L^w.\underline{b}.q + \sigma_{W \cup L}.B) \\
&= \sigma_L^w.\underline{b}.\theta_I(q) + \sigma_{W \cup L}.\theta_I(\sigma_L^w.(\sigma.\underline{a}.p + \underline{b}.q) + \sigma_{W \cup L}.(\sigma.\underline{a}.p + B)) \\
&= \sigma_L^w.\underline{b}.\theta_I(q) + \sigma_{W \cup L}(\sigma_L^w.\underline{b}.\theta_I(q) + \sigma_{W \cup L}.\theta_I(\sigma.\underline{a}.p + B)) \\
&= \sigma_L^w.\underline{b}.\theta_I(q) + \sigma_{W \cup L}(\sigma_L^w.\underline{b}.\theta_I(q) + \sigma_{W \cup L}(\sigma_L^w.(\underline{a}.\theta_I(p) + \underline{b}.\theta_I(q) + \\
&\qquad\qquad\qquad \sigma_{W \cup L}.\underline{b}.\theta_I(q))))).
\end{aligned}$$

In the last step of the derivation we unfold B one more time and apply the maximal progress operator. Even though no winner has expired, the maximal progress operator prohibits the expiration of the stochastic delay after time slice 3 as given by $\sigma_{W \cup L}.\underline{b}.\theta_I(q)$. \square

Such an interaction between the timed and stochastic delays can also be used to specify a probabilistic behavior after a passage of time. An example is given in Section 8.3, where we give the specification of concurrent alternating bit protocol in $\text{TCP}_{\text{rec}}^{\text{drst}}$. However, the theory cannot express a standard probabilistic choice between processes that do not allow passage of time.

Next, we take a closer look at the interaction between undelayable action, delayable action, and stochastic delay prefixes.

5.3 Interaction between the Prefix Operators

First, we investigate a common type of synchronization between delayable action and stochastic delay prefixes in the parallel composition by means of an example derivation.

Example 5.3.1 We consider the synchronization of the passage of time of the delayable action and a stochastic delay given by the term $\partial_H(\bar{a}.p \parallel [^W_L].q)$. We put $H = \{a\}$, i.e., we suppress the synchronizing action as in standard compositional modeling. Let $\bar{a}.p = \mu A.\{A = \underline{a}.p + \sigma.A\}$ and $[^W_L].q = \mu B.\{B = \sigma_L^w.q + \sigma_{w \cup L}.B\}$. Then,

$$\begin{aligned} \partial_H(A \parallel B) &= \partial_H((\underline{a}.p + \sigma.A) \parallel (\sigma_L^w.q + \sigma_{w \cup L}.B)) \\ &= \partial_H(\underline{a}.p \parallel B) + \sigma_L^w.(A \parallel |q|_L) + \sigma_{w \cup L}.(A \parallel B) \\ &= \sigma_L^w.\partial_H(A \parallel |q|_L) + \sigma_{w \cup L}.\partial_H(A \parallel B), \end{aligned}$$

i.e., $\partial_H(\bar{a}.p \parallel [^W_L].q) = [^W_L].\partial_H(\bar{a}.p \parallel |q|_L)$.

If $q = \bar{b}.q'$ and the synchronization of a and b is defined, i.e., $\gamma(a, b) = c$ for some $c \in \mathcal{A}$, then it is also common to prioritize this communication. For example, this can be a communication via a channel, so naturally one wants this communication to happen as soon as it is enabled. In that case, one typically has a specification of the form $\theta_I(\partial_H(\bar{a}.p \parallel [^W_L].\bar{b}.q'))$ for $H = \{a, b\}$ and $I = \{c\}$. Then by extending the previous derivation with $\bar{b}.q' = \mu C.\{C = \underline{b}.q' + \sigma.C\}$ one obtains:

$$\begin{aligned} \theta_I(\partial_H(A \parallel B)) &= \theta_I(\sigma_L^w.\partial_H(A \parallel |q|_L) + \sigma_{w \cup L}.\partial_H(A \parallel B)) \\ &= \sigma_L^w.\theta_I(\partial_H((\underline{a}.p + \sigma.A) \parallel (\underline{b}.q' + \sigma.C))) + \sigma_{w \cup L}.\theta_I(\partial_H(A \parallel B)) \\ &= \sigma_L^w.\theta_I(\partial_H(\underline{a}.p \parallel \bar{b}.q') + \underline{b}.(\underline{a}.p \parallel q') + \underline{c}.(p \parallel q') + \\ &\quad \sigma.(A \parallel C))) + \sigma_{w \cup L}.\theta_I(\partial_H(A \parallel B)) \\ &= \sigma_L^w.\theta_I(\underline{c}.\partial_H(p \parallel q') + \sigma.\partial_H(A \parallel C)) + \sigma_{w \cup L}.\theta_I(\partial_H(A \parallel B)) \\ &= \sigma_L^w.\underline{c}.\theta_I(\partial_H(p \parallel q')) + \sigma_{w \cup L}.\theta_I(\partial_H(A \parallel B)), \end{aligned}$$

i.e., $\theta_I(\partial_H(\bar{a}.p \parallel [^W_L].\bar{b}.q')) = [^W_L].\underline{c}.\theta_I(\partial_H(p \parallel q'))$. \square

The composition of a stochastic delay prefixed process and the delayable deadlock constant can also be resolved in terms of stochastic delay processes. Unlike the compositions with delayable actions, the delayable deadlock propagates through the stochastic delay. We show the case of the alternative composition where $\bar{0} = \mu C.\{C = \sigma.C\}$ and the stochastic delay prefixed term is defined as above:

$$B + C = (\sigma_L^w.q + \sigma_{w \cup L}.B) + \sigma.C = \sigma_L^w.(q + C) + \sigma_{w \cup L}.(B + C),$$

i.e., $[^W_L].q + \bar{0} = [^W_L].(q + \bar{0})$.

Example 5.3.1 and the previous discussion illustrate that the synchronization of passage of time of stochastic delay and delayable action prefixed

terms can be handled without resorting to guarded recursive specifications comprising timed delay prefixes. Together with Example 5.2.2 and the discussion in Section 5.1 involving delayable actions motivated us to develop a theory in the framework of $\text{TCP}_{\text{rec}}^{\text{drst}}$ that directly manipulates delayable action and stochastic delay prefixes.

5.4 Signature

The signature of $\text{DTCP}_{\text{rec}}^{\text{dst}}$ comprises separate delayable action and stochastic delay prefixes, but their semantics is based on the interpretation as guarded recursive specifications in $\text{TCP}_{\text{rec}}^{\text{drst}}$. The signature is given in the following definition.

Definition 5.4.1 The signature of $\text{DTCP}_{\text{rec}}^{\text{dst}}$ is given by

$$P ::= \underline{0} \mid \underline{1} \mid \bar{0} \mid \underline{a}.P \mid \bar{a}.P \mid [{}^W_L].P \mid |P|_D \mid \partial_H(P) \mid \theta_I(P) \mid P+P \mid P\|P \mid \mu A.S,$$

where $a \in \mathcal{A}$, $W, L, D \subseteq \mathcal{V}$ with $W \neq \emptyset$ and $W \cap L = \emptyset$, $H, I \subseteq \mathcal{A}$, $S \in \mathcal{G}$, and $A \in \mathcal{R}(S)$. The set of closed terms that do not contain term variables is denoted by $\mathcal{C}(\text{DTCP}_{\text{rec}}^{\text{dst}})$ and it is ranged over by p and q . \square

By the definition of the delayable deadlock constant, the delayable action, and the stochastic delay prefix, the process theory $\text{DTCP}_{\text{rec}}^{\text{dst}}$ is embedded in $\text{TCP}_{\text{rec}}^{\text{drst}}$. The semantics of closed $\text{DTCP}_{\text{rec}}^{\text{dst}}$ -terms is given by the racing timed transition scheme induced by the solutions of guarded recursive specifications that model the above constructs.

All auxiliary operations straightforwardly extend to the restriction of the theory to $\text{DTCP}_{\text{rec}}^{\text{dst}}$ by an application to the corresponding recursive specification. The renaming operation is extended as:

$$\begin{aligned} \bar{0}[Y/X] &= \bar{0} \\ (\bar{a}.p)[Y/X] &= \bar{a}.p \\ ([{}^W_L].p)[Y/X] &= [{}^W_L].p && \text{if } X \notin W \cup L \\ ([{}^W_L].p)[Y/X] &= [{}^{(W \setminus \{X\}) \cup \{Y\}}_L].p && \text{if } X \in W \\ ([{}^W_L].p)[Y/X] &= [{}^W_{(L \setminus \{X\}) \cup \{Y\}}].p[Y/X] && \text{if } X \in L. \end{aligned}$$

Next, we give the additional axioms for the dependence scope and the encapsulation operator.

$ \bar{0} _\emptyset = \bar{0}$	A5.1
$ \bar{a}.p _\emptyset = \bar{a}.p$	A5.2
$\bar{a}.p = \bar{a}. p _\emptyset$	A5.3
$[\overset{W}{L}].p = [\overset{W}{L}].p _{W \cup L}$	A5.4
$[\overset{L}{L}].p = [\overset{L}{L}]. p _L$	A5.5
$\partial_H([\overset{W}{L}].p) = [\overset{W}{L}].\partial_H(p)$	A5.6

Table 5.1: Axioms for the dependence scope encompassing stochastic delay prefixes

5.5 Dependence Scope and Encapsulation

The additional axioms that manage the dependence scope and encapsulation operator in $\text{DTCP}_{\text{rec}}^{\text{dst}}$ are given in Table 5.1. In the proof of the following theorem we show that the axioms are sound.

Theorem 5.5.1 *The axioms in Table 5.1 are sound.* □

PROOF We prove the soundness of the axioms by showing that both sides can be rewritten to recursive specifications that have the same solution.

[A5.1] Suppose $\bar{0} = \mu A.\{A = \sigma.A\}$ and $|\bar{0}|_\emptyset = \mu(|A|_\emptyset).\{A = \sigma.A\}$. Then, $|A|_\emptyset = |\sigma.A|_\emptyset = \sigma.A = A$.

[A5.2] Suppose $\bar{a}.p = \mu A.\{A = \underline{a}.p + \sigma.A\}$ and $|\bar{a}.p|_\emptyset = \mu(|A|_\emptyset).\{A = \underline{a}.p + \sigma.A\}$. Then,

$$|A|_\emptyset = |\underline{a}.p + \sigma.A|_\emptyset = |\underline{a}.p|_\emptyset + |\sigma.A|_\emptyset = \underline{a}.p + \sigma.A = A$$

[A5.3] Suppose $\bar{a}.p = \mu A.\{A = \underline{a}.p + \sigma.A\}$ and $\bar{a}.|p|_\emptyset = \mu B.\{B = \underline{a}.|p|_\emptyset + \sigma.B\}$. Then,

$$A = \underline{a}.p + \sigma.A = \underline{a}.|p|_\emptyset + \sigma.A.$$

Now, by the principles of RDP^- and RSP , the solutions of A and B coincide.

[A5.4] Suppose $[\overset{W}{L}].p = \mu A.\{A = \sigma_L^w.p + \sigma_{w \cup L}.A\}$ and $|[\overset{W}{L}].p|_{W \cup L} = \mu|A|_{W \cup L}.\{A = \sigma_L^w.p + \sigma_{w \cup L}.A\}$. Then,

$$A = \sigma_L^w.p + \sigma_{w \cup L}.A = |\sigma_L^w.p|_{W \cup L} + |\sigma_{w \cup L}.A|_{W \cup L} = |\sigma_L^w.p + \sigma_{w \cup L}.A|_{W \cup L} = |A|_{W \cup L}.$$

[A5.5] Suppose $[^W_L].p = \mu A.\{A = \sigma_L^w.p + \sigma_{w \cup L}.A\}$ and $[^W_L].|p|_L = \mu B.\{B = \sigma_L^w.|p|_L + \sigma_{w \cup L}.B\}$. Then,

$$A = \sigma_L^w.p + \sigma_{w \cup L}.A = \sigma_L^w.|p|_L + \sigma_{w \cup L}.A.$$

Now, by the principles of RDP^- and RSP , the solutions of A and B coincide.

[A5.6] Suppose $[^W_L].p = \mu A.\{A = \sigma_L^w.p + \sigma_{w \cup L}.A\}$ and $\partial_H([^W_L].p) = \mu \partial_H(B).\{B = \sigma_L^w.\partial_H(p) + \sigma_{w \cup L}.B\}$. Then

$$\begin{aligned} \partial_H(A) = \partial_H(\sigma_L^w.p + \sigma_{w \cup L}.A) &= \partial_H(\sigma_L^w.p) + \partial_H(\sigma_{w \cup L}.A) = \\ &= \sigma_L^w.\partial_H(p) + \sigma_{w \cup L}.\partial_H(A). \end{aligned}$$

Now, by the principles of RDP^- and RSP , the solutions of A and B coincide. \blacksquare

Next, we deal with the expansion laws of the rest of the operators.

5.6 Alternative Composition

We derive expansion laws for the alternative composition, α -conversion, the parallel composition, and the maximal progress operator for stochastic delays that deal only with undelayable action and stochastic delay prefixed terms along the lines of the expansion laws A4.8 for the alternative composition, A4.9 for the α -conversion, A4.17 for the parallel composition, and A4.18 and A4.19 for the maximal progress operator in the timed delay setting, respectively. Again, the laws are based on normal forms in which the stochastic delays are in resolved races. The normal forms have additional delayable action prefixes and the optional delayable deadlock constant. The constant is present if no summands prefixed by a delayable action or a stochastic delay exist because it is the neutral element for the delayable action prefix and it propagates through the stochastic delays prefix as shown above in Section 5.1.

A normal form of a term $p \in \text{DTCP}_{\text{rec}}^{\text{dst}}$ that is unique for the stochastic delays modulo commutativity, associativity, and naming of independent delays is given by

$$p = \left| \sum_{i=1}^u a_i.p_i + \sum_{j=1}^d \bar{b}_j.q_j + \sum_{k=1}^s [^W_{L_k}].r_k(+\bar{0})(+\underline{1})(+\underline{0}) \right|_D,$$

where $D \subseteq \text{R}(p) = W_k \cup L_k$, $\text{rr}([^W_{L_k}], [^W_{L_\ell}])$ holds for $1 \leq k < \ell \leq s$, the summand $\bar{0}$ may or may not exist provided that there are no delayable

action or stochastic delay prefixed summands, the summand $\underline{1}$ may or may not exist, and the summand $\underline{0}$ exists if none of the other summands does.

Next, we give the expansion law for the alternative composition $p + p'$, where

$$p' = \left| \sum_{i'=1}^{u'} \underline{a}_{i'} \cdot p'_{i'} + \sum_{j'=1}^{d'} \bar{b}_{j'} \cdot q'_{j'} + \sum_{k'=1}^{s'} \left[\begin{smallmatrix} W'_{k'} \\ L'_{k'} \end{smallmatrix} \right] \cdot r'_{k'} (+ \bar{0}) (+ \underline{1}) (+ \underline{0}) \right|_{D'}$$

with $D' \subseteq R(p') = W'_{k'} \cup L'_{k'}$ and $\text{rr}(\left[\begin{smallmatrix} W'_{k'} \\ L'_{k'} \end{smallmatrix} \right], \left[\begin{smallmatrix} W'_{\ell'} \\ L'_{\ell'} \end{smallmatrix} \right])$ holds for $1 \leq k' < \ell' \leq n'$.

The expansion is presented in three steps: (1) for the action prefixed terms, (2) for the stochastic delay prefixed terms that form a joint race, and (3) for the stochastic delay prefixed terms in resolved races. As for the standard semantics of the alternative composition, the action transitions from both terms are available, expressed by the term $\text{act}(p + p')$ given by

$$\text{act}(p + p') = \sum_{i=1}^u \underline{a}_i \cdot p_i + \sum_{i'=1}^{u'} \underline{a}'_{i'} \cdot p'_{i'} + \sum_{j=1}^d \bar{b}_j \cdot q_j + \sum_{j'=1}^{d'} \bar{b}'_{j'} \cdot q'_{j'}.$$

Recall that in a joint race of two stochastic delays $\left[\begin{smallmatrix} W_1 \\ L_1 \end{smallmatrix} \right]$ and $\left[\begin{smallmatrix} W_2 \\ L_2 \end{smallmatrix} \right]$ there are three possible outcomes: $\left[\begin{smallmatrix} W_1 \cup W_2 \\ L_1 \cup L_2 \end{smallmatrix} \right]$, $\left[\begin{smallmatrix} W_1 \cup W_2 \\ L_1 \cup L_2 \end{smallmatrix} \right]$, and $\left[\begin{smallmatrix} W_1 \cup W_2 \\ L_1 \cup L_2 \end{smallmatrix} \right]$. The existence of the outcomes depends on the relation between the losers and winners of the delays (cf. Section 2.1). If one term can only allow passage of time according to the delayable deadlock constant, then the stochastic delays synchronize on the passage of time, whereas the constant propagates through the prefixes. The term $\text{jrc}(p + p')$ gives the joint outcomes of the races between the racing delays of p and p' . It is given by $\text{jrc}(p + p') =$

$$\begin{aligned} & \sum_{k, k' : (W_k \cup W'_{k'}) \cap (L_k \cup L'_{k'}) = \emptyset} \left[\begin{smallmatrix} W_k \cup W'_{k'} \\ L_k \cup L'_{k'} \end{smallmatrix} \right] \cdot (|r_k|_{L_k} + |r'_{k'}|_{L'_{k'}} (+ \bar{0})) + \\ & \sum_{k : W_k \cap R(p') = \emptyset} \sum_{k'} \left[\begin{smallmatrix} W_k \\ L_k \cup R(p') \end{smallmatrix} \right] \cdot (|r_k|_{L_k} + \sum_{k'=1}^{s'} \left[\begin{smallmatrix} W'_{k'} \\ L'_{k'} \end{smallmatrix} \right] \cdot r'_{k'}) + \\ & \sum_k \sum_{k' : R(p) \cap W'_{k'} = \emptyset} \left[\begin{smallmatrix} W'_{k'} \\ R(p) \cup L'_{k'} \end{smallmatrix} \right] \cdot (\sum_{k=1}^s \left[\begin{smallmatrix} W_k \\ L_k \end{smallmatrix} \right] \cdot r_k + |r'_{k'}|_{L'_{k'}}). \end{aligned}$$

The first sum expresses the case when the winners from both delays win together. The optional $\bar{0}$ constant is propagated if there are no winners from one side, i.e., if the index set of either k or k' is empty. In that case the last two sums do not exist. If both summands do not have stochastic delay prefixed terms, then no sum exists. In the second sum the left delay

coming from the term p wins the race, which also means that it wins the race for every stochastic delay prefixed summand of p' . The third sum is the symmetric case of the second situation.

The racing delays of p and p' are in a resolved race in p and p' , respectively. Thus, a racing delay from p is in a resolved race in $p + p'$ if it is in a resolved race with every racing delay of p' . This is expressed by the term $\text{rsd}(p + p')$ given by:

$$\begin{aligned} \text{rsd}(p + p') = & \sum_{k: \text{rr}([L_k], [L_{k'}'])} [L_k] \cdot r_k + \\ & \sum_{k': \text{rr}([L_k], [L_{k'}'])} [L_{k'}'] \cdot r_{k'}'. \end{aligned}$$

Now, we have all the ingredients to state the expansion law of the alternative composition.

Theorem 5.6.1 *Let p and p' have the normal forms from above. If $\text{I}(p) \cap \text{R}(p') = \text{R}(p) \cap \text{I}(p') = \emptyset$, then the normal form of the alternative composition $p + p'$ is given by*

$$p + p' = |\text{act}(p + p') + \text{jrc}(p + p') + \text{rsd}(p + p') (+\bar{0})(+\underline{1})(+\underline{0})|_{D \cup D'} \quad \mathbf{A5.7}$$

where the summand $\bar{0}$ exists if p or p' contain it and both of them do not have delayable action or stochastic delay prefixed summands, the summand $\underline{1}$ exists if p or p' contain it, and $\underline{0}$ exists if none of the other summands does. \square

PROOF To see that $p + p'$ is again in normal form it is sufficient to observe that (1) $\text{rr}([L_k \cup \text{R}(p')], [L_\ell \cup L_{k'}'])$ holds for every $1 \leq k, \ell \leq s$ and $1 \leq k' \leq n'$ satisfying $W_k \cap \text{R}(p') = \emptyset$ and $(W_\ell \cup W_{k'}') \cap (L_\ell \cup L_{k'}') = \emptyset$, (2) $\text{rr}([L_{k'}'], [L_k \cup L_{\ell'}'])$ holds as the symmetric case of (1) for $1 \leq k \leq s$ and $1 \leq k', \ell' \leq s'$, (3) $\text{rr}([L_k \cup \text{R}(p')], [L_{k'}'])$ holds for every $1 \leq k \leq s$ and $1 \leq k' \leq s'$ satisfying $W_k \cap \text{R}(p') = \emptyset$ and $\text{R}(p) \cap W_{k'}' = \emptyset$, (4) $\text{rr}([L_k], [L_\ell \cup L_{k'}'])$ holds for every $1 \leq k, \ell \leq n$ and $1 \leq k' \leq n'$ satisfying $\text{rr}([L_k], [L_{k'}'])$ for all $1 \leq k' \leq n'$ and $(W_\ell \cup W_{k'}') \cap (L_\ell \cup L_{k'}') = \emptyset$, (5) $\text{rr}([L_k], [L_{k'}'])$ holds for every $1 \leq k \leq n$ and $1 \leq k' \leq n'$ satisfying $\text{rr}([L_k], [L_{k'}'])$ for all $1 \leq k' \leq n'$ and $\text{R}(p) \cap W_{k'}' = \emptyset$, (6) the symmetric

case of (4) holds, and (7) the symmetric case of (5) holds. For example, (3) holds because $W_k \cup L_k \cup \mathbf{R}(p') = \mathbf{R}(p) \cup \mathbf{R}(p') = \mathbf{R}(p) \cup W'_{k'} \cup L'_{k'}$, $\emptyset \neq W_k \cap \mathbf{R}(p) \subseteq W_k \cap (\mathbf{R}(p) \cup L'_{k'})$, and $\emptyset \neq W'_{k'} \cap \mathbf{R}(p') \subseteq W'_{k'} \cap (L_k \cup \mathbf{R}(p'))$.

Next, we show that the recursive specification of $p + p'$ in terms of timed delay prefixed terms and its expansion have the same solution. Suppose $\bar{0} = \mu A. \{A = \sigma.A\}$, $\bar{b}_j.q_j = \mu B_j. \{B_j = \underline{b}_j.q_j + \sigma.B_j\}$ for $1 \leq j \leq d$, $\bar{b}'_{j'}.q'_{j'} = \mu B'_{j'}. \{B'_{j'} = \underline{b}'_{j'}.q'_{j'} + \sigma.B'_{j'}\}$ for $1 \leq j' \leq d'$, $[L_k^k].r_k = \mu C_k. \{C_k = \sigma_{L_k^k}.r_k + \sigma_{\mathbf{R}(p)}.C_k\}$ for $1 \leq k \leq s$, and $[L_{k'}^{k'}].r'_{k'} = \mu C'_{k'}. \{C'_{k'} = \sigma_{L_{k'}^{k'}}.r'_{k'} + \sigma_{\mathbf{R}(p')}.C'_{k'}\}$ for $1 \leq k' \leq s'$.

First, we analyze the alternative composition of a delayable action and a stochastic delay prefixed term. By Theorem 4.3.3 for the alternative composition of $\bar{b}_j.q_j$ and $[L_k^k].r_k$ of p one calculates

$$\begin{aligned} B_j + C_k &= (\underline{b}_j.q_j + \sigma.B_j) + (\sigma_{L_k^k}.r_k + \sigma_{\mathbf{R}(p)}.C_k) + \\ &= \underline{b}_j.B_j + \sigma_{L_k^k}.r_k + \sigma_{\mathbf{R}(p)}.(B_j + C_k) \end{aligned}$$

for $1 \leq j \leq d$ and $1 \leq k \leq s$. It should not be difficult to see that such an alternative composition is associative.

Similarly, for the alternative composition of $[L_k^k].r_k$ and $[L_\ell^\ell].r_\ell$ one has:

$$\begin{aligned} C_k + C_\ell &= (\sigma_{L_k^k}.r_k + \sigma_{\mathbf{R}(p)}.C_k) + (\sigma_{L_\ell^\ell}.r_\ell + \sigma_{\mathbf{R}(p)}.C_\ell) + \\ &= \sigma_{L_k^k}.r_k + \sigma_{L_\ell^\ell}.r_\ell + \sigma_{\mathbf{R}(p)}.(C_k + C_\ell) \end{aligned}$$

for $1 \leq k, \ell \leq s$. Again, this type of alternative composition is associative. Similar results are obtained for the interaction between the delayable actions, and the interaction with the delayable deadlock constant.

Now, the normal forms of p and p' in terms of timed delay prefixed terms can be given as

$$\begin{aligned} p &= \mu \left(\left| \sum_{i=1}^u \underline{a}_i.p_i + \sum_{j=1}^d \underline{b}_j.q_j + \sum_{k=1}^s \sigma_{L_k^k}.(r_k (+ A)) + \right. \right. \\ &\quad \left. \left. \sigma_{\mathbf{R}(p)}.((A +) \sum_{j=1}^d B_j + \sum_{k=1}^s C_k) (+ \underline{1})(+ \underline{0}) \right|_D \right).S \\ p' &= \mu \left(\left| \sum_{i'=1}^{u'} \underline{a}'_{i'}.p'_{i'} + \sum_{j'=1}^{d'} \underline{b}'_{j'}.q'_{j'} + \sum_{k'=1}^{s'} \sigma_{L_{k'}^{k'}}.(r'_{k'} (+ A)) + \right. \right. \\ &\quad \left. \left. \sigma_{\mathbf{R}(p')}.((A +) \sum_{j'=1}^{d'} B'_{j'} + \sum_{k'=1}^{s'} C'_{k'}) (+ \underline{1})(+ \underline{0}) \right|_{D'} \right).S \end{aligned}$$

where the optional recursion variable A exists if the term contains the $\bar{0}$ summand and the guarded recursive specification S contains the equations for A , B_j , $B'_{j'}$, C_k , and $C'_{k'}$ for $1 \leq j \leq d$, $1 \leq j' \leq d'$, $1 \leq k \leq s$, and

$1 \leq k' \leq s'$. We can use this normal form to compute $p + p'$. Also, in a similar fashion one can rewrite the expanded alternative composition in terms of timed delays. Then, by the principles of RDP^- and RSP , it is easily derived that both specification have the same solution.

Here, we show only the derivation of $p + p'$, as the one for the expanded form is straightforward. By Theorem 4.3.3 the expansion of $p + p'$ is given by $p + p' =$

$$\begin{aligned}
& \left| \sum_{i=1}^u a_i \cdot p_i + \sum_{i'=1}^{u'} a_{i'} \cdot p_{i'} + \sum_{j=1}^d b_j \cdot q_j + \sum_{j'=1}^{d'} b_{j'} \cdot q_{j'} + \right. \\
& \quad \left. \sum_{k, k': (W_k \cup W_{k'}) \cap (L_k \cup L_{k'}) = \emptyset} \sigma_{L_k \cup L_{k'}}^{W_k \cup W_{k'}} \cdot (|r_k|_{L_k} + |r_{k'}|_{L_{k'}} (+ A)) + \right. \\
& \quad \sum_{k: W_k \cap R(p') = \emptyset} \sigma_{L_k \cup R(p')}^{W_k} \cdot (|r_k|_{L_k} + |(A +) \sum_{j'=1}^{d'} B_{j'} + \sum_{k'=1}^{s'} C_{k'}|_{R(p')}) + \\
& \quad \sum_{k': R(p) \cap W_{k'} = \emptyset} \sigma_{R(p) \cup L_{k'}}^{W_{k'}} \cdot (|(A +) \sum_{j=1}^d B_j + \sum_{k=1}^n C_k|_{R(p)} + |r_{k'}|_{L_{k'}}) + \\
& \quad \sum_{k: \text{rr}(\sigma_{L_k}^{W_k}, \sigma_{L_{k'}}^{W_{k'}}) \text{ for all } 1 \leq k' \leq n'} \sigma_{L_k}^{W_k} \cdot r_k + \\
& \quad \sum_{k': \text{rr}(\sigma_{L_k}^{W_k}, \sigma_{L_{k'}}^{W_{k'}}) \text{ for all } 1 \leq k \leq n} \sigma_{L_{k'}}^{W_{k'}} \cdot r_{k'} + \\
& \quad \sigma_{R(p) \cup R(p')} \cdot ((A +) \sum_{j=1}^d B_j + \sum_{j'=1}^{d'} B_{j'} + \sum_{k=1}^s C_k + \sum_{k'=1}^{s'} C_{k'}) (+ \underline{1})(+ \underline{0})|_{D \cup D'},
\end{aligned}$$

where the recursion variable A exists if p or p' contain it and they do not have delayable action or stochastic delay prefixed summands, the summand $\underline{1}$ exists if p or p' contain it, and $\underline{0}$ exists if none of the other summands does. Now, having in mind that $|C_k|_{R(p)} = C_k$ and $|C_{k'}|_{R(p')} = C_{k'}$, and along the lines of the derivations in Examples 5.2.2 and 5.3.1 it is straightforward, but meticulous, to calculate that the expansion A5.7 and the expansion of $p + p'$ using timed delay prefixed terms coincide, which completes the proof. \blacksquare

Next, we give the α -conversion in terms of stochastic delays.

5.7 α -conversion

Similarly to the α -conversion law A4.9 of Theorem 4.4.2, we have the following theorem for renaming independent racing stochastic delays.

Theorem 5.7.1 *Let p have the normal form*

$$p = \left| \sum_{i=1}^u \underline{a}_i \cdot p_i + \sum_{j=1}^d \bar{b}_j \cdot q_j + \sum_{k=1}^s [{}_{L_k}^{W_k}] \cdot r_k (+ \bar{0})(+ \underline{1})(+ \underline{0}) \right|_D$$

with $D \subseteq \mathbf{R}(p) = W_k \cup L_k$, $\text{rr}([{}_{L_k}^{W_k}], [{}_{L_\ell}^{W_\ell}])$ holds for $1 \leq k < \ell \leq s$. Then the independent racing delay $X \notin D$ can be renamed to Y as follows:

$$p = \left| \sum_{i=1}^u \underline{a}_i \cdot p_i + \sum_{j=1}^d \bar{b}_j \cdot q_j + \sum_{k: X \notin \mathbf{R}(p)} [{}_{L_k}^{W_k}] \cdot r_k + \sum_{k: X \in W_k} [{}_{L_k}^{(W_k \setminus \{X\}) \cup \{Y\}}] \cdot r_k + \sum_{k: X \in L_k} [{}_{(L_k \setminus \{X\}) \cup \{Y\}}^{W_k}] \cdot r_k [Y/X] (+ \bar{0})(+ \underline{1})(+ \underline{0}) \right|_D \quad \mathbf{A5.8.}$$

□

PROOF A direct consequence of Theorem 4.4.2 as the disjoint sums range over all stochastic delay prefixed terms. ■

We proceed with the resolution of the parallel composition.

5.8 Parallel Composition

As for the alternative composition, we split the expansion of the parallel composition in three parts: (1) resolution of an action prefix, (2) synchronization of action prefixes, and (3) resolution of the race condition. Again, unlike the alternative composition, the parallel composition is associative as a direct consequence of Theorem 4.6.3. Also, we assume that p and p' are in normal form as above.

By $\text{pre}(p \parallel p')$ we denote the term that takes the action prefixes out of the parallel composition. It is given by:

$$\begin{aligned} \text{pre}(p \parallel p') = & \sum_{i=1}^u \underline{a}_i \cdot (|p_i|_\emptyset \parallel p') + \sum_{i'=1}^{u'} \underline{a}'_{i'} \cdot (p \parallel |p'_{i'}|_\emptyset) + \\ & \sum_{j=1}^d \bar{b}_j \cdot (|q_j|_\emptyset \parallel q') + \sum_{j'=1}^{d'} \bar{b}'_{j'} \cdot (q \parallel |q'_{j'}|_\emptyset). \end{aligned}$$

As action transitions reset races, the racing delays of the summand that was prefixed by the action transition have to be made independent.

The synchronization of the action transitions is represented by the term $\text{syn}(p \parallel p')$. It is given by:

$$\begin{aligned} \text{syn}(p \parallel p') = & \sum_{i,i': \gamma(a_i, a'_{i'})=aa_{ii'}} \underline{aa}_{ii'} \cdot (|p_i|_\emptyset \parallel |p'_{i'}|_\emptyset) + \\ & \sum_{i,j': \gamma(a_i, b'_{j'})=ab_{ij'}} \underline{ab}_{ij'} \cdot (|p_i|_\emptyset \parallel |q'_{j'}|_\emptyset) + \\ & \sum_{j,i': \gamma(b_j, a'_{i'})=ba_{ji'}} \underline{ba}_{ji'} \cdot (|q_j|_\emptyset \parallel |p'_{i'}|_\emptyset) + \\ & \sum_{j,j': \gamma(b_j, b'_{j'})=bb_{jj'}} \underline{bb}_{jj'} \cdot (|q_j|_\emptyset \parallel |q'_{j'}|_\emptyset). \end{aligned}$$

Cross-synchronization of undelayable and delayable actions is possible, but in that case the resulting action must be undelayable.

The stochastic delay prefixes are merged in the same manner as for the alternative composition. The joint outcomes are given by the term $\text{std}(p \parallel p')$, where $\text{std}(p \parallel p') =$

$$\begin{aligned} & \sum_{k,k': (W_k \cup W_{k'}) \cap (L_k \cup L_{k'}) = \emptyset} \left[\begin{smallmatrix} W_k \cup W_{k'} \\ L_k \cup L_{k'} \end{smallmatrix} \right] \cdot ((|r_k|_{L_k} (+\bar{0})) \parallel (|r'_{k'}|_{L'_{k'}} (+\bar{0}))) + \\ & \sum_{k: W_k \cap \mathbf{R}(p') = \emptyset} \sum_{k'} \left[\begin{smallmatrix} W_k \\ L_k \cup \mathbf{R}(p') \end{smallmatrix} \right] \cdot (|r_k|_{L_k} \parallel \left[\begin{smallmatrix} W_{k'} \\ L'_{k'} \end{smallmatrix} \right] \cdot r'_{k'}) + \\ & \sum_k \sum_{k': \mathbf{R}(p) \cap W'_{k'} = \emptyset} \left[\begin{smallmatrix} W'_{k'} \\ W_k \cup L_k \cup L'_{k'} \end{smallmatrix} \right] \cdot (\left[\begin{smallmatrix} W_k \\ L_k \end{smallmatrix} \right] \cdot r_k \parallel |r'_{k'}|_{L'_{k'}}) \end{aligned}$$

The leading stochastic delay determines the set of losers in the term it prefixes as in the timed setting. The optional summand $\bar{0}$ exists if one of the components does not have stochastic delay prefixes as for the alternative composition above.

Similarly to the alternative composition, we combine the three parts from above to give an expansion law for the parallel composition.

Theorem 5.8.1 *Let p and p' have the normal forms as above. If $\mathbf{I}(p) \cap \mathbf{R}(p') = \mathbf{R}(p) \cap \mathbf{I}(p') = \emptyset$, then the normal form of the parallel composition of p and p' is given by*

$$p \parallel p' = |\text{pre}(p \parallel p') + \text{syn}(p \parallel p') + \text{std}(p \parallel p') (+\bar{0})(+1)(+0)|_{D \cup D'} \quad \mathbf{A5.9},$$

where the summands $\bar{0}$ and $\underline{1}$ exist if both p and p' contain them, respectively, and $\underline{0}$ exists if none of the other summands does. \square

PROOF Along the lines of the proof of Theorem 5.6.1 and using the expansion law A4.17 of Theorem 4.6.2 for expanding the timed delay prefix representations of p and p' . Note that if both p and p' have the $\bar{0}$ summand, then they cannot have stochastic delay or delayable action prefixed terms. \blacksquare

Next, we give the expansion of the maximal progress operator.

5.9 Maximal Progress

Unlike the timed delay prefixed processes for which it is not important to resolve the races in order to apply the maximal progress operator, when dealing with stochastic delay prefixes all races must be resolved. We illustrate the situation by an example.

Example 5.9.1 Let $p = [X].\underline{a}.\underline{0} + [Y].\underline{b}.\underline{0}$. If we directly apply $\theta_{a,b}(p)$ and assume that it propagates through stochastic delay prefixes as for timed delay prefixes, we have $\theta_{a,b}(p) = \theta_{a,b}([X].\underline{a}.\underline{0} + [Y].\underline{b}.\underline{0}) = [X].\theta_{a,b}(\underline{a}.\underline{0}) + [Y].\theta_{a,b}(\underline{b}.\underline{0}) = p$. Now, assume $[X].\underline{a}.\underline{0} = \mu A.\{A = \sigma^x.\underline{a}.\underline{0} + \sigma_{x'}A\}$ and $[Y].\underline{b}.\underline{0} = \mu B.\{B = \sigma^y.\underline{b}.\underline{0} + \sigma_{y'}B\}$. Then, by using Theorem 4.3.3 for the expansion of $\theta_{a,b}(p)$ one calculates

$$\begin{aligned} & \theta_{a,b}(A + B) \\ &= \theta_{a,b}((\sigma^x.\underline{a}.\underline{0} + \sigma_{x'}A) + (\sigma^y.\underline{b}.\underline{0} + \sigma_{y'}B)) \\ &= \theta_{a,b}(\sigma_{y'}^x(\underline{a}.\underline{0} + B) + \sigma^{x,y}(\underline{a}.\underline{0} + \underline{b}.\underline{0}) + \sigma_{x'}^y(A + \underline{b}.\underline{0}) + \sigma_{x,y'}(A + B)) \\ &= \sigma_{y'}^x\theta_{a,b}(\underline{a}.\underline{0} + B) + \sigma^{x,y}\theta_{a,b}(\underline{a}.\underline{0} + \underline{b}.\underline{0}) + \sigma_{x'}^y\theta_{a,b}(A + \underline{b}.\underline{0}) + \sigma_{x,y'}\theta_{a,b}(A + B) \\ &= \sigma_{y'}^x.\underline{a}.\underline{0} + \sigma^{x,y}(\underline{a}.\underline{0} + \underline{b}.\underline{0}) + \sigma_{x'}^y.\underline{b}.\underline{0} + \sigma_{x,y'}\theta_{a,b}(A + B). \end{aligned}$$

Thus, $\theta_{a,b}(p) = [\overset{x}{Y}].\underline{a}.\underline{0} + [X, Y].(\underline{a}.\underline{0} + \underline{b}.\underline{0}) + [\overset{y}{X}].\underline{b}.\underline{0}$. \square

Following the guidelines of Example 5.9.1 and having in mind that the maximal progress operator turns delayable actions to undelayable ones (cf. Section 5.1), as well as it disables passage of time (cf. Section 4.7), we have the following theorem.

Theorem 5.9.2 *Let the normal form of p be as above. Then the expansion law of the maximal progress $\theta_I(p)$ is given by*

$$\theta_I(p) = \left| \sum_{i=1}^u a_i.\theta_I(p_i) + \sum_{j=1}^d \bar{b}_j.\theta_I(q_j) (+\underline{1})(+\underline{0}) \right|_D$$

if $(\bigcup_{i=1}^u \{a_i\} \cup \bigcup_{j=1}^d \{b_j\}) \cap I \neq \emptyset$ **A5.10**

$$\theta_I(p) = \left| \sum_{i=1}^u a_i.\theta_I(p_i) + \sum_{j=1}^d \bar{b}_j.\theta_I(q_j) + \sum_{k=1}^s [{}_{L_k}^{W_k}].\theta_I(r_k) \right.$$

$(+\bar{0})(+\underline{1})(+\underline{0}) \Big|_D$ if $(\bigcup_{i=1}^u \{a_i\} \cup \bigcup_{j=1}^d \{b_j\}) \cap I = \emptyset$ **A5.11,**

where the conditions apply for the optional summands as in p . □

PROOF By direct application of Theorem 4.6.2 for the expansion of the maximal progress operator for timed prefixed delays. ■

Similarly to the timed process theory we can give head normal forms that pave the way for a ground-completeness result and unique solutions to the guarded recursive specifications.

5.10 Head Normal Form

It should come as no surprise that every closed $\text{DTCP}_{\text{rec}}^{\text{dst}}$ -term can be rewritten in a head normal form, in a similar way as the one of Corollary 4.8.1, as all operators can be expressed using an alternative composition of undelayable action, delayable action, and stochastic delay prefixed terms. However, to show this, we require two more idempotency axioms that deal with undelayable and delayable action prefixed terms. They can be stated as follows:

$$\underline{a}.p + \bar{a}.p = \bar{a}.p \quad \mathbf{A5.12},$$

$$\bar{a}.p + \bar{a}.p = \bar{a}.p \quad \mathbf{A5.13}.$$

To show the soundness of the axioms assume that $\bar{a}.p = \mu A.\{A = \underline{a}.p + \sigma.A\}$. Then by using axiom A4.20 and the expansion A4.8 of the alternative composition we have

$$[\text{A12}] \quad \underline{a}.p + A = \underline{a}.p + (\underline{a}.p + \sigma.A) = \underline{a}.p + \underline{a}.p + \sigma.A = \underline{a}.p + \sigma.A = A$$

$$[\text{A13}] \quad A + A = (\underline{a}.p + \sigma.A) + (\underline{a}.p + \sigma.A) = \underline{a}.p + \sigma.(A + A).$$

Now, by the principles of RDP^- and RSP , we have that A and $A + A$ have the same solution.

The head normal form is stated in the following corollary.

Corollary 5.10.1 *Every closed term $p \in \mathcal{C}(\text{DTCP}_{\text{rec}}^{\text{dst}})$ can be represented in a unique head normal form modulo commutativity, associativity, and naming of independent delays, viz.*

$$p = \left| \sum_{i=1}^u \underline{a}_i.p_i + \sum_{j=1}^d \bar{b}_j.q_j + \sum_{k=1}^s [{}_{L_k}^{W_k}].r_k(+\bar{0})(+\underline{1})(+\underline{0}) \right|_D$$

with $D \subseteq \mathbb{R}(p) = W_k \cup L_k$, $\underline{a}_i.p_i \neq \underline{a}_{i'}.p_i$ and $\underline{a}_i.p_i \neq \underline{b}_j.q_j$ for $1 \leq i, i' \leq u$ with $i \neq i'$ and $1 \leq j \leq d$, $\text{rr}([{}_{L_k}^{W_k}], [{}_{L_\ell}^{W_\ell}])$ holds for $1 \leq k < \ell \leq s$, the summand $\bar{0}$ may or may not exist provided that there are no delayable action or stochastic delay prefixed summands, the summand $\underline{1}$ may or may not exist, and the summand $\underline{0}$ exists if none of the other summands does. \square

PROOF The proof is analogous to the one of Corollary 4.8.1 by replacing the axioms and expansion laws that deal with timed delays with ones that deal with stochastic delays. By the axioms A4.1 – A4.4 and A4.7 in Table 4.2 and axioms A5.4 and A5.5 in Table 5.1 for manipulation with the dependence scope operator, the expansion law A5.7 of the alternative composition of Theorem 5.6.1, the α -conversion law A5.8 for renaming of independent delays of Theorem 5.7.1, axioms A4.10 – A4.13 and A4.16 in Table 4.3 and axiom A5.6 in Table 5.1 that deal with the encapsulation operator, the expansion law A5.9 of the parallel composition of Theorem 5.8.1, the expansion laws A5.10 and A5.11 of the maximal progress of Theorem 5.9.2 every closed $\text{TCP}_{\text{rec}}^{\text{drst}}$ term can be reduced to the temporary normal form that is unique only for timed delays modulo commutativity, associativity, and naming of independent delays. By using axioms A4.20, A5.12, and A5.13 for idempotence of the action prefixed terms in the alternative composition as a rewriting rule from left to right we also obtain uniqueness for the action prefixed terms modulo commutativity and associativity. \blacksquare

As before, since every term can be reduced in the head normal form given by Corollary 5.10.1, the equations form a ground-complete theory.

Theorem 5.10.2 *Axioms A4.1 – A4.4 and A4.7 in Table 4.2 and axioms A5.4 and A5.5 in Table 5.1, axioms A4.10 – A4.13 and A4.16 in Table 4.3 and axiom A5.6 in Table 5.1, the expansion laws A5.7–A5.11, and axioms A4.20, A5.12, and A5.13 are ground-complete for the term model $\mathbb{P}(\text{DTCP}_{\text{rec}}^{\text{dst}})/\simeq_t$.* \square

PROOF Analogous to the proof of Theorem 4.9.1 for the timed setting. ■

Next, we show the simplifications that can be applied in the case of race-complete process specifications that induce only races with all possible outcomes.

5.11 Race-Complete Process Specifications

Race-complete process specifications can be characterized as specifications that can be rewritten such that only stochastic delay prefixes of the form $[X]_{\cdot}$ for $X \in \mathcal{V}$ occur in the process term. This restriction assures that all possible outcomes of the race are given and it enables the associativity of the alternative composition. As a consequence, the equational theory becomes much more elegant as we do not have to resort to normal forms from the start. However, the expansion of the parallel composition still requires a (head) normal form in which the timed/stochastic delays are in resolved racing contexts. Also, to resolve the maximal progress either an additional operator or a normal form that makes explicit the undelayable action prefixes and the timed delays is required.

We present in Table 5.2 the alternative simplified axioms for the alternative composition and renaming of independent racing delays of race-complete process specifications.

Axiom A5.14 shows how to rename independent racing delays. It is applicable as only complete races can be formed, which in the beginning are formed by single stochastic delays. Axiom A5.15 is a simplified version of the merging of dependence scopes of the expansion law A4.8 of the alternative composition. The naming conflict condition remains the same. Axioms A5.16 and A5.17 are the standard axioms for the idempotence of the termination and the neutrality of the deadlock in the alternative composition. Axioms A5.18 and A5.19 state the commutativity and associativity of the alternative composition. Axiom A5.20 shows the resolution of the race when the winners from both terms have a common racing delay. Axiom A5.21 shows the resolution of the race when the winner comes from the left summand. In that case, the stochastic delays of the right summand must be made dependent on the winners of the first summand. The dependent racing delays of the remaining process of the left summand can come only from the set of losers L_1 . Finally, axiom A5.22 gives all possible outcomes when there are no restrictions on the merging of the racing delays. The axioms can be turned into a rewriting system to give the normal form

$$|[X].p|_{\emptyset} = |[Y].p|_{\emptyset} \text{ if } F_X = F_Y \quad \mathbf{A5.14}$$

$$|p_1 + p_2|_D = |p_1|_D + |p_2|_D \\ \text{if } I(|p_1|_D) \cap R(|p_2|_D) = R(|p_1|_D) \cap I(|p_2|_D) = \emptyset \quad \mathbf{A5.15}$$

$$\underline{1} + \underline{1} = \underline{1} \quad \mathbf{A5.16}$$

$$p + \underline{0} = p \quad \mathbf{A5.17}$$

$$p + q = q + p \quad \mathbf{A5.18}$$

$$(p + q) + r = p + (q + r) \quad \mathbf{A5.19}$$

$$[L_1^{W_1}].p_1 + [L_2^{W_2}].p_2 = [L_1 \cup L_2^{W_1 \cup W_2}].(|p_1|_{L_1} + |p_2|_{L_2}), \\ \text{if } W_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap L_2 = L_1 \cap W_2 = \emptyset \quad \mathbf{A5.20}$$

$$[L_1^{W_1}].p_1 + [L_2^{W_2}].|p_2|_{L_2} = [L_1 \cup W_2 \cup L_2^{W_1}].(|p_1|_{L_1} + [L_2^{W_2}].p_2), \\ \text{if } L_1 \cap W_2 \neq \emptyset \text{ and } W_1 \cap W_2 = W_1 \cap L_2 = \emptyset \quad \mathbf{A5.21}$$

$$[L_1^{W_1}].p_1 + [L_2^{W_2}].p_2 = [W_2 \cup L_2 \cup L_1^{W_1}].(|p_1|_{L_1} + [L_2^{W_2}].p_2) + \\ [L_1 \cup L_2^{W_1 \cup W_2}].(|p_1|_{L_1} + |p_2|_{L_2}) + [L_2 \cup W_1 \cup L_1^{W_2}].([L_1^{W_1}].p_1 + |p_2|_{L_2}), \\ \text{if } W_1 \cap W_2 = L_1 \cap W_2 = W_1 \cap L_2 = \emptyset \quad \mathbf{A5.22}$$

Table 5.2: Alternative simplified axioms in case of race-complete process specifications

from Section 4.3 and they replace the expansion laws A5.7 and A5.8 for the alternative composition and α -conversion, respectively.

To illustrate the features of the process theory $\text{DTCP}_{\text{rec}}^{\text{dst}}$, we specify the $G/G/1/\infty$ queue and solve its recursive specification.

5.12 The $G/G/1/\infty$ Queue

We proceed by specifying and solving the recursive specification of the $G/G/1/\infty$ queue, also discussed in [69]. The queue can be compactly modeled by a generalized semi-Markov process [48] given in Figure 5.1. Here, a denotes the event of an arrival job and s is an event of a processed job. The states are labeled by the clocks that correspond to events. In every state the clocks with which the state is labeled are reset, whereas the others are updated typically using spent-lifetime semantics. After an expiration of a clock, the transition labeled by the name of the event is taken. The model shows that jobs arrive constantly in the queue and the server processes one job at a time.

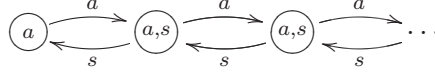


Figure 5.1: Generalized semi-Markov model of the $G/G/1/\infty$ queue

We specify the $G/G/1/\infty$ queue in our setting by using three components given by the recursive equations for A , Q_0 , and S .

$$\begin{aligned} A &= |[X].\bar{s}_1.A|_{\emptyset} \\ Q_0 &= \bar{r}_1.Q_1 \\ Q_{k+1} &= \bar{r}_1.Q_{k+2} + \bar{s}_2.Q_k \quad \text{if } k \geq 0 \\ S &= \bar{r}_2.[Y].\bar{s}_3.S \end{aligned}$$

The equation for A models the arrival process that is delayed by the stochastic delay $[X]$. This delay corresponds to the clock a in the generalized semi-Markov representation of the process in Figure 5.1. The process modeled by Q_0 is the standard representation of a queue. It comprises delayable actions and it is always able to receive a new job or to offer a job that has already been queued. Finally, the process given by S models the server that has processing time distributed according to Y . Its counterpart in Figure 5.1 is given by the clock s . It is always ready to accept a job when it is idle.

The specification of the $G/G/1/\infty$ queue itself is given by

$$Q = \theta_I(\partial_H(A \parallel Q_0 \parallel S)),$$

where $\gamma(r_1, s_1) = c_1$, $\gamma(r_2, s_2) = c_2$, $H = \{s_1, r_1, s_2, r_2\}$, and $I = \{c_1, c_2, s_3\}$. Along the lines of Examples 5.2.2 and 5.3.1 one calculates:

$$\begin{aligned} Q = S_0 &= \theta_I(\partial_H(A \parallel Q_0 \parallel S)) \\ &= \theta_I(\partial_H(|[X].\bar{s}_1.A|_{\emptyset} \parallel \bar{r}_1.Q_1 \parallel \bar{r}_2.[Y].\bar{s}_3.S)) \\ &= \theta_I(\partial_H(|[X].\bar{s}_1.A \parallel \bar{r}_1.Q_1 \parallel \bar{r}_2.[Y].\bar{s}_3.S|_{\emptyset})) \\ &= |[X].\underline{c}_1.\theta_I(\partial_H(A \parallel (\bar{r}_1.Q_2 + \bar{s}_2.Q_0) \parallel \bar{r}_2.[Y].\bar{s}_3.S))|_{\emptyset} \\ &= |[X].\underline{c}_1.\underline{c}_2.\theta_I(\partial_H(A \parallel Q_0 \parallel |[Y].\bar{s}_3.S|_{\emptyset}))|_{\emptyset}. \end{aligned}$$

Now, we put $S_1 = \theta_I(\partial_H(A \parallel Q_0 \parallel |[Y].\bar{s}_3.S|_{\emptyset}))$. Then, $Q = |[X].\underline{c}_1.\underline{c}_2.S_1|_{\emptyset}$.

We proceed with the following derivation for S_1 :

$$\begin{aligned}
S_1 &= \theta_I(\partial_H(A \parallel Q_0 \parallel |[Y].\bar{s}_3.S|_\emptyset)) \\
&= \theta_I(\partial_H(|[X].\bar{s}_1.A \parallel \bar{r}_1.Q_1 \parallel [Y].\bar{s}_3.S|_\emptyset)) \\
&= \theta_I(\partial_H(|[\bar{X}].(\bar{s}_1.A \parallel \bar{r}_1.Q_1 \parallel [Y].\bar{s}_3.S) + [\bar{X}].([X].\bar{s}_1.A \parallel \bar{r}_1.Q_1 \parallel \bar{s}_3.S) + \\
&\quad [X, Y].(\bar{s}_1.A \parallel \bar{r}_1.Q_1 \parallel \bar{s}_3.S)|_\emptyset)) \\
&= \theta_I(\partial_H(|[\bar{X}].\bar{c}_1.(A \parallel Q_1 \parallel [Y].\bar{s}_3.S) + [\bar{X}].\bar{s}_3.([X].\bar{s}_1.A \parallel \bar{r}_1.Q_1 \parallel S) + \\
&\quad [X, Y].(\bar{c}_1.\bar{s}_3.(A \parallel Q_1 \parallel S) + \bar{s}_3.\bar{c}_1.(A \parallel Q_1 \parallel S))|_\emptyset)) \\
&= |[\bar{X}].\underline{c}_1.\theta_I(\partial_H((A \parallel Q_1 \parallel [Y].\bar{s}_3.S))) + [\bar{X}].\underline{s}_3.\theta_I(\partial_H((A \parallel Q_0 \parallel S))) + \\
&\quad [X, Y].(\underline{c}_1.\underline{s}_3.\underline{c}_2.\theta_I(\partial_H(A \parallel Q_0 \parallel |[Y].\bar{s}_3.S|_\emptyset)) + \\
&\quad \underline{s}_3.\underline{c}_1.\underline{c}_2.\theta_I(\partial_H(A \parallel Q_0 \parallel |[Y].\bar{s}_3.S|_\emptyset))|_\emptyset) \\
&= |[\bar{X}].\underline{c}_1.S_2 + [\bar{X}].\underline{s}_3.S_0 + [X, Y].(\underline{c}_1.\underline{s}_3.\underline{c}_2.S_1 + \underline{s}_3.\underline{c}_1.\underline{c}_2.S_1),
\end{aligned}$$

where $S_2 = \theta_I(\partial_H((A \parallel Q_1 \parallel [Y].\bar{s}_3.S))$. Similarly, one can show that:

$$S_k = |[\bar{X}].\underline{c}_1.S_{k+1} + [\bar{X}.\bar{Y}].(\underline{c}_1.\underline{s}_3.\underline{c}_2.S_k + \underline{s}_3.\underline{c}_1.\underline{c}_2.S_k) + [\bar{X}].\underline{s}_3.\underline{c}_2.S_{k-1} \text{ for } k > 1$$

which completes the solution for the $G/G/1/\infty$ queue, where

$$S_{k+1} = \theta_I(\partial_H(A \parallel Q_k \parallel |[Y].\bar{s}_3.S|_\emptyset)) \text{ for } k > 1.$$

5.13 Summary

We derive the notions of delayable actions and stochastic delays as solutions of recursive equations comprising timed delays. We show that we need not resort to these specifications in order to manipulate processes prefixed by delayable actions and stochastic delays. This lead us to the derived theory of communicating processes with discrete stochastic time. Similarly to the timed setting, we develop a sound and ground-complete equational theory that is again based on normal forms in which the races between the stochastic delays are resolved. We illustrate the approach by specifying and solving the recursive specification of the $G/G/1/\infty$ queue.

Next, we take the opposite view and attempt to establish a stochastic process theory in such a way that it extends the standard real-time setting. However, first we need to introduce the notion of context-sensitive interpolation that represents a restriction of time additivity that conforms to the race condition.

Chapter 6

Extending Real Time with Stochastic Time

In this chapter we take the viewpoint of stochastic time and we attempt to mold real-time process algebras so that they can accommodate a stochastic extension. We give a simple example to illustrate the situation.

Suppose we wish to extend the term $\sigma^2.\sigma^3.p$ with stochastic time. If we make use of time additivity, i.e., only observe the accumulative delays, we may consider, e.g., the term $\sigma^5.p$ or even $\sigma^1.\sigma^3.\sigma^1.p$. Now, suppose that $X_1, X_2, X_3, X_5 \in \mathcal{V}$ are arbitrary distributed non-Dirac random variable suitably chosen to represent the delays of duration 1, 2, 3, and 5, respectively. Now, from the properties of the race condition (cf. Section 2) we have that $[X_2].[X_3].p$ is different from $[X_5].p$ and $[X_1].[X_3].[X_1].p$. The reason is that in a every racing context $[X_5]$ produces different probabilities and samples for the winning delays than $[X_2].[X_3]$ or $[X_1].[X_3].[X_1]$.

One solution is to consider timed delays as atomic, i.e., to explicitly state the delay that we want to model. In that way timed and stochastic delays are put on the same level and their expirations are viewed as discrete events. The motivation for such an approach stems from a discussion on the overlapping properties of prominent stochastic bisimulation relations.

6.1 Overview of Stochastic Bisimulation Relations

In general, timed bisimulation relations require that bisimilar processes delay the same amount of time. They typically employ time additivity, i.e., merging of subsequent timed delays into a joint single delay with the same accumulative duration, to compare the delays [84, 11]. For example, $\sigma^3.\sigma^2.p$ and $\sigma^5.p$ are typically considered to be equivalent.

On the contrary, stochastic bisimulation relations are set up as discrete event bisimulation relations (which is inherent to the underlying performance model), i.e., they consider passage of time per an atomic stochastic delay transition. To the best of our knowledge, with the exception of [65], all stochastic process theories consider stochastic bisimulation that is atomic in this sense: in [52] the actions are coupled with the stochastic clocks, in [42] there is an alternation between clocks and action transitions, whereas in [27, 26] the merging is impeded by the combination of the pre-selection policy and start-termination semantics. Although originally introduced as an atomic stochastic bisimulation [64], an effort is made in [65] to define a notion of weak stochastic bisimulation that merges subsequent stochastic delays. Unfortunately, such an approach is not compositional as merging of stochastic delays does not support the race condition. A simple example illustrates the problem. The process $[X].[Y].p$ intuitively has the same stochastic properties as the process $[Z].p$ provided that $F_Z = F_{X+Y}$. However, standard compositions involving these processes are not bisimilar. For example, $[X].[Y].p + [U].p$ is not bisimilar to $[Z].p + [U].p$ in a race condition setting. This is because the race of X and U induces a different probabilistic choice on the winner compared to the race between Z and U .

We conclude that from the viewpoint of stochastic process theories that employ the race condition, it is more convenient to treat timed delays as atomic, discrete event constructs, which levels the semantic differences with their stochastic counterpart.

6.2 Extending Real Time with Stochastic Time

The treatment of timed delays as atomic requires a new and more restrictive notion of time additivity. Again, we illustrate the situation by an example.

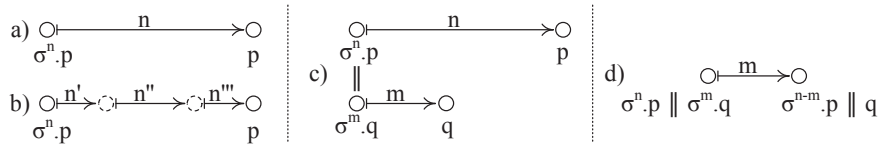


Figure 6.1: a) A timed delay prefix $\sigma^n.p$, b) arbitrary interpolation of σ^n into $\sigma^{n'}$, $\sigma^{n''}$, and $\sigma^{n'''}$, c) parallel composition of $\sigma^n.p$ and $\sigma^m.q$, and d) context-sensitive interpolation of σ^n in the context of the parallel composition with $\sigma^m.q$

Example 6.2.1 Figure 6.1b depicts arbitrary interpolation of the timed delay σ^n of the process $\sigma^n.p$ of Figure 6.1a to three timed delays $\sigma^{n'}$, $\sigma^{n''}$, and $\sigma^{n'''}$ satisfying $n' + n'' + n''' = n$. If interpreted as an atomic timed delay, the delay must be left intact, unless it is in a context of a composition that would induce a race. A race with another timed delay σ^m of the process $\sigma^m.q$ induced by a parallel composition is depicted in Figure 6.1c. Only then we can interpolate the longer delay (in this case $n > m$, as depicted in Figure 6.1d, conforming to race condition semantics. We note that the resulting process $(\sigma^{n-m}.p) \parallel q$ accounts for the remaining delay σ^{n-m} . \square

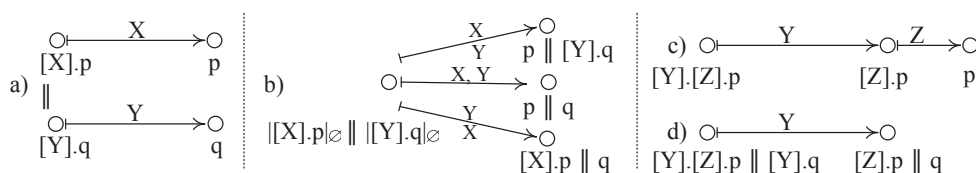


Figure 6.2: a) Stochastic extension of the composition in Figure 6.1c), b) independent race condition with every possible outcome, c) stochastic extension of $\sigma^n.p$ in accordance with the context-sensitive interpolation of Figure 6.1d), and d) dependent race condition synchronizing the dependent delays

In the stochastic setting of this thesis, such behavior can be interpreted both for the independent or dependent race condition as depicted in Figure 6.2. Suppose that the original timed delay σ^n of Example 6.2.1 is replaced by the stochastic delay $[X]$, obtaining $[X].p$ as depicted in Figure 6.2a, and $\sigma^m.q$ is extended to $[Y].q$. In Figure 6.2b we consider an independent race given by the term $[[X].p]_{\emptyset} \parallel [[Y].q]_{\emptyset}$, which results in all possible outcomes as discussed in Section 2. Here, we label the transitions with the winners on top and the losers below the arrow. This approach conveniently models independent components competing for the same resource.

Now, suppose that the components are considered dependent regarding their timing aspects. For example, $\sigma^n.p$ of Example 6.2.1 is a controller that has a timeout greater than the tolerated response time of the process that it controls. This can be represented in the timed model as $\sigma^m.q$ and conditioned by the fact that $n > m$. In such a situation the stochastic modeling using the independent race condition leads to undesirable behavior. For example, the premature expiration of the stochastic delay of the controller given by the outcome $[X]$ could introduce non-existent deadlock behavior as it did not wait for the result of the process that successfully finished its task. In this

case, relying on the context-sensitive interpolation, the correct modeling of $\sigma^n.p$ would be $[Y].[Z].p$ as depicted in Figure 6.2c. The idea is that both, the controller and the process, should synchronize on the dependent stochastic delay $[Y]$. The delay is followed by the short timeout $[Z]$ that models the extra timed delay σ^{n-m} in the context-sensitive interpolated representation $(\sigma^m.(\sigma^{n-m}.p) \parallel q)$ of $(\sigma^n.p) \parallel (\sigma^m.q)$. The situation is depicted in Figure 6.2d.

Another way of modeling the above system is to explicitly state that the stochastic delay $[Y]$ should be the winner of the race between $[X]$ and $[Y]$. This is done by specifying $\sigma^m.q$ in stochastic time as $[\overset{Y}{X}].q$. Such a specification expresses the result of the race between $[X]$ and $[Y]$. The parallel composition $[X].p \parallel [\overset{Y}{X}].q$ is resolved as $[\overset{Y}{X}].([X].p \parallel q)$. In this case, however, the race is incomplete, i.e., the other disjoint outcomes $[\overset{X}{\emptyset}.Y]$ and $[\overset{Y}{X}]$ are not present. As elaborated above, a major consequence is that the equational theory of terms exhibiting incomplete races is more intricate as the alternative composition is no longer associative and one must rely on normal form representations.

We conclude that the use of context-sensitive interpolation helps in identification of the nature of the stochastic delays by allowing the treatment of timed delays as atomic. However, it should be noted that its use cannot always reveal whether the delays should be interpreted as independent or dependent in stochastic time. This still remains the task of the designer.

6.3 Context-Sensitive Interpolation

From a process theoretical point of view, fundamental properties of time are time determinism and time additivity, i.e., passage of time does not make a choice by itself and subsequence timed delays can be merged together to the accumulative delay, respectively [84, 11]. They are captured by the following operational rules.

$$\begin{array}{lll}
 \mathbf{6.1} \frac{}{\sigma^n.p \xrightarrow{n} p} & \mathbf{6.2} \frac{p \xrightarrow{n} p'}{\sigma^m.p \xrightarrow{m+n} p'} & \mathbf{6.3} \frac{p_1 \xrightarrow{n} p'_1, p_2 \xrightarrow{n} p'_2}{p_1 + p_2 \xrightarrow{n} p'_1 + p'_2}.
 \end{array}$$

When treating timed delays as atomic, rule 6.1 holds again, but rule 6.2 for time additivity now fails. Therefore, we add instead of rule 6.2, two new rules similar to rule 6.3 for time determinism that enable context-sensitive interpolation when racing timed delays exhibit different durations:

$$\begin{array}{ll}
 \mathbf{6.4} \frac{p_1 \xrightarrow{m} p'_1, p_2 \xrightarrow{n} p'_2, m < n}{p_1 + p_2 \xrightarrow{m} p'_1 + \sigma^{n-m}.p'_2} & \mathbf{6.5} \frac{p_1 \xrightarrow{m} p'_1, p_2 \xrightarrow{n} p'_2, m > n}{p_1 + p_2 \xrightarrow{n} \sigma^{m-n}.p'_1 + p'_2}.
 \end{array}$$

Note the emphasis on performing the shortest winning duration first. Rules 6.4 and 6.5 give rise to the following axioms.

$$\sigma^m.p_1 + \sigma^m.p_2 = \sigma^m.(p_1 + p_2) \quad \mathbf{A6.1}$$

$$\sigma^m.p_1 + \sigma^{m+n}.p_2 = \sigma^m.(p_1 + \sigma^n.p_2) \quad \mathbf{A6.2}.$$

Axiom A6.1 enables time determinism, whereas Axiom A6.2 replaces the standard axiom for time additivity $\sigma^m.\sigma^n.p = \sigma^{m+n}.p$. Together with commutativity the latter allows for context-sensitive interpolation. If zero-time delays are allowed, then rule 6.3 and axiom A6.1 become obsolete. More details can be found in [85].

Remark 6.3.1 We note, however, that the coexistence of the rules 6.2, 6.4, and 6.5 is at best problematic. In that case, one must ensure that the context-sensitive interpolation has always been applied for every timed transition as in the opposite case some transitions may be ‘lost’. Consider the following process term $p = (\sigma^1.\underline{a}.\underline{0} + \sigma^2.\underline{b}.\underline{0}) + \sigma^1.\underline{c}.\underline{0}$. Using time additivity one derives the transition $\sigma^1.\underline{a}.\underline{0} + \sigma^2.\underline{b}.\underline{0} \xrightarrow{2} \underline{b}.\underline{0}$. Now, by applying the context-sensitive interpolation rule $p \xrightarrow{1} \sigma^1.\underline{b}.\underline{0} + \underline{c}.\underline{0}$ and the option of performing a transition labeled by a is lost. \square

To conclude, at first sight context-sensitive interpolation may seem too restrictive compared to time additivity. However, context-sensitive interpolation does exactly what time additivity is typically used for: merging of delays with the same duration by taking the shortest/minimal possible delay in a context with compositional operators. Moreover, context-sensitive interpolation fits naturally in the expansion of the parallel composition, which makes it a suitable candidate for a finer notion of time additivity in real-time process algebras. Finally, the bisimulation relation remains unchanged as context-sensitive interpolation is handled in the operational semantics on the model level. However, it is noted that the resulting process equivalence is finer. For example, $\sigma^2.\sigma^3.p$ and $\sigma^5.p$ are no longer related, though $\sigma^2.\sigma^3.p$ and $\sigma^5.p + \sigma^2.\underline{0}$ are.

Remark 6.3.2 For strong bisimilarity the elimination of the effect of time additivity can be achieved by representing every timed delay as atomic, i.e., by merging all subsequent timed delays. If the process terms are represented in such ‘normal forms’, then time additivity and context-sensitive interpolation have the same power of distinction. However, when using some weaker type of bisimilarity, e.g., weak timed or timed branching bisimilarity, context-sensitive interpolation is finer because of the effect of elimination of

silent steps between two timed delays. For example, in weak timed semantics with time additivity $\underline{a}.\sigma^3.\tau.\sigma^2.p$, where τ represents the silent step, is typically considered equivalent to $\underline{a}.\sigma^5.p$, whereas when using context-sensitive interpolation the same process is equivalent to $\underline{a}.\sigma^3.\sigma^2.p$, but not to the latter as discussed above. \square

We proceed by presenting a stochastic process theory that makes use of the concepts discussed above to deal with stochastic time as an extension of the real-time process theory. The gain lays in an expansion law that respects time determinism and the explicit treatment of the maximal progress.

6.4 Stochastic Process Theory $\text{TCP}_{\text{rec}}^{\text{st}}$

We proceed with the presentation of $\text{TCP}_{\text{rec}}^{\text{st}}(\mathcal{A}, \mathcal{V}, \mathcal{R}, \gamma)$, the theory of communicating processes with (discrete) stochastic time. It has the same signature as the process theory $\text{DTCP}_{\text{rec}}^{\text{dst}}$. However, unlike $\text{DTCP}_{\text{rec}}^{\text{dst}}$, which was constructed from the primitives of $\text{TCP}_{\text{rec}}^{\text{drst}}$ by deriving delayable action and stochastic delay prefixes using recursive equations, here, we give the semantics of closed $\text{TCP}_{\text{rec}}^{\text{st}}$ -terms from scratch. In return, we obtain a greater insight into the relationship between real time and the race condition, re-discovering the notion of context-sensitive interpolation. The semantics is given in terms of *stochastic transition schemes* that, in essence, represent stochastic automata with explicit symbolic representation of the race condition and passage of time as for the racing timed transitions schemes. We focus on the handling of the race condition, the expansion for the parallel composition, and the maximal progress operator. We note that we obtain the same equational theory as for $\text{DTCP}_{\text{rec}}^{\text{dst}}$, but the semantics is given in terms of finite objects as passage of time is observed on an atomic delay scale and not per unit of time.

For ease of reference, we repeat the signature of $\text{DTCP}_{\text{rec}}^{\text{dst}}$ as it is the signature of $\text{TCP}_{\text{rec}}^{\text{st}}$ as well. The processes are defined by the following grammar:

$$P ::= \underline{0} \mid \underline{1} \mid \bar{0} \mid \underline{a}.P \mid \bar{a}.P \mid [^W_L].P \mid |P|_D \mid \partial_H(P) \mid \theta_I(P) \mid P+P \mid P\|P \mid \mu A.S$$

where $a \in \mathcal{A}$, $W, L, D \subseteq \mathcal{V}$ with $W \neq \emptyset$ and $W \cap L = \emptyset$, $H, I \subseteq \mathcal{A}$, $S \in \mathcal{G}$, and $A \in \mathcal{R}(S)$.

As before, we use an environment to keep track of the dependencies between the racing delays. Recall, $[^W_L]$ denotes an outcome of a race that was won by W and lost by L for disjoint $W, L \subseteq \mathcal{V}$ with $W \neq \emptyset$. However,

in view of time determinism, time has passed equally for all racing delays in $W \cup L$. To denote that after a stochastic delay $[\frac{W}{L}]$, the same amount of time that has passed for the winners W has also passed for the losers L , we use an environment $\eta: \mathcal{V} \rightarrow 2^{\mathcal{V}}$. For each $X \in \mathcal{V}$, $\eta(X)$ is a set that contains one representative of the winners of every race that X lost. One representative suffices, because all winners share the same sample in the winning race. If $\eta(X) = \emptyset$, then X has never lost a race. We write \mathcal{H} for the set of all such environments. We illustrate the use of these environments by means of an example.

Example 6.4.1 The process term $[\frac{X;Y}{Z}].[\frac{U}{Z}].p$ has a stochastic delay transition in which X and Y are the winners and Z is the loser. In the resulting process $[\frac{U}{Z}].p$, the variable Z must be made dependent on the amount of time that has passed for X and Y before. This can be denoted either by $\eta(Z) = \{X\}$ or $\eta(Z) = \{Y\}$, assuming that initially $\eta(Z) = \emptyset$. As Z again loses a race, this time to U , the transition induced by $[\frac{U}{Z}]$ updates $\eta(Z)$ to $\eta(Z) = \{X, U\}$, provided X was chosen as a representative in the first race. \square

As in the timed setting, the environment does not affect the outgoing transitions. It is used only to calculate the correct distribution of the racing delays. However, it represents the effect of the races symbolically, so the exhibited samples of the expired winners are required to compute the age of the racing delays. Suppose that $\rho \in \mathcal{E}$ is used to keep track of the exhibited samples. Then the racing delay $[Y]$ in the environment η has an age, i.e., it participated in races that it lost with the total amount of time

$$\alpha_{\eta,\rho}(Y) = \sum_{X \in \eta(Y)} (\rho(X) + \alpha_{\eta,\rho}(X)).$$

By convention, $\alpha_{\eta,\rho}(Y) = 0$ if $\eta(Y) = \emptyset$. The distribution of Y at that point in time is $F_Y|\alpha_{\eta,\rho}(Y)$, provided that $F_Y(\alpha_{\eta,\rho}(Y)) < 1$. Thus, in order to compute the aged distribution of the racing delay $[Y]$, one has to know its complete racing history, i.e., the names of all delays that contribute in the derivation of its age $\alpha_{\eta,\rho}(Y)$. The *racing history* in a environment η of a set of racing delays R is defined by

$$H_\eta(R) = R \cup \bigcup_{X \in R} (\eta(X) \cup H_\eta(\eta(X))).$$

The racing history plays an important role as it must be maintained in the stochastic transition schemes and related by the corresponding bisimulation relation. It also introduces additional naming conflicts as there might be clashes between the racing delays names and the ones in the racing history.

6.5 Stochastic Transition Schemes

Closed $\text{TCP}_{\text{rec}}^{\text{st}}$ terms are given semantics by means of stochastic transition schemes that treat passage of time of stochastic delays as atomic. The stochastic transition schemes are based on the same idea as the racing timed transition schemes, i.e., keeping track of the aging of the racing delays. As passage of time in stochastic transition schemes is observed as a discrete event in terms of expired winners, then keeping track of the ages amounts to preserving the racing history. To represent passive (unbounded) passage of time for delayable actions we use an additional transition relation \rightsquigarrow . It also gives the semantics of the delayable deadlock constant $\bar{0}$ (cf. Section 5.1). Again, outgoing passive delay transitions exist only if the state does not have outgoing stochastic delay transitions.

Definition 6.5.1 A stochastic transition scheme $(S \times \mathcal{H}, \mathcal{L}, \mathcal{V}, \longrightarrow, \Longrightarrow, \rightsquigarrow, \downarrow, \text{I})$ is a tuple, where $u = \langle s, \eta \rangle \in S \times \mathcal{H}$ is a state in an environment η and

- $\longrightarrow \subseteq (S \times \mathcal{H}) \times \mathcal{L} \times (S \times \mathcal{H})$ is the labeled transition relation.
- $\Longrightarrow \subseteq (S \times \mathcal{H}) \times (2^{\mathcal{V}} \setminus \{\emptyset\}) \times 2^{\mathcal{V}} \times (S \times \mathcal{H})$ is a stochastic delay transition relation satisfying that for every $u \xrightarrow[L]{W} u'$ it holds that the winners and the losers are disjoint, i.e., $W \cap L = \emptyset$, and for every two different transitions originating from the same state $u \xrightarrow[L_1]{W_1} u_1 \neq u \xrightarrow[L_2]{W_2} u_2$ the predicate $\text{rr}([L_1^{W_1}], [L_2^{W_2}])$ holds.
- $\rightsquigarrow \subseteq (S \times \mathcal{H}) \times (S \times \mathcal{H})$ is the passive delay transition. It can relate two states $u \rightsquigarrow u'$ only if the state u does not have any outgoing stochastic delay transitions.
- $\downarrow \subseteq S \times \mathcal{H}$ is the termination option predicate.
- $\text{I}: S \rightarrow \mathcal{V}$ is the independent racing delays function. It satisfies that $\text{I}(s) \subseteq \bigcup_{\langle s, \eta \rangle \xrightarrow[L]{W} \langle s', \eta' \rangle} (W \cup L)$, for every $\eta \in \mathcal{H}$. \square

Again, we have that $W_1 \cup L_1 = W_2 \cup L_2$ for every $u \xrightarrow[L_1]{W_1} u_1$ and $u \xrightarrow[L_2]{W_2} u_2$ as $\text{rr}([L_1^{W_1}], [L_2^{W_2}])$ holds. Thus, for every state s there exists a set of racing delays $\text{R}(s)$ and $\text{I}(s) \subseteq \text{R}(s)$. Then, the set of dependent racing delays is given by $\text{D}(s) = \text{R}(s) \setminus \text{I}(s)$. For notational convenience, we also use $\text{R}(u)$, $\text{D}(u)$, and $\text{I}(u)$ when clear from the context.

Each stochastic transition scheme, coupled to an assignment of probability distributions to the stochastic delays induces a probabilistic timed

transition system. As the racing history is represented symbolically, we have to calculate the age of the racing delays using $\alpha_{\eta,\rho}(-)$ as given above. For that reason we also need an initial environment, standardly set to the zero sample environment ρ_0 . The action transitions and the termination predicate are adopted from the stochastic transition scheme as above in Definition 3.2.2. The probability measure of the probabilistic timed delay is induced by the winners, the losers, and the sample of the winning delay. We employ the notation

$$\text{RC}_n(W, L) = \text{P}(W = n, L > n),$$

extending the one of Definition 3.2.2. The formal definition is as follows.

Definition 6.5.2 Let $R = (S \times \mathcal{H}, A, V, \longrightarrow, \mapsto, \downarrow, \text{I})$ be a stochastic transition scheme, $d: V \rightarrow \mathcal{F}$ a distribution assignment function, and $\rho_0 \in \mathcal{E}$ an initial sample environment. If R does not have passive delay transitions, then (R, d, ρ_0) induces the probabilistic timed transition system $P = ((S \times \mathcal{H}) \times \mathcal{E}, A, \rightarrow, \mapsto, \downarrow)$, where the action transition and termination options \rightarrow and \downarrow of P are induced by \longrightarrow and \downarrow of R , respectively, and $\mapsto(v) = (\mathbb{N} \times (S \times \mathcal{H}) \times \mathcal{E}, \text{P})$ with $v = (\langle s, \eta \rangle, \rho)$, is the probability space induced by the race condition. The probability measure P is given by

$$\text{P}(n, v') = \frac{\text{RC}_n(W', L')}{\sum_{\langle s, \eta \rangle \xrightarrow[L]{W} \langle \bar{s}, \bar{\eta} \rangle} \text{P}(W < L)} \quad \text{if } \langle s, \eta \rangle \xrightarrow[L]{W'} \langle s', \eta' \rangle,$$

where $v' = (\langle s', \eta' \rangle, \rho')$, the distribution functions of $X \in \text{R}(s)$ are given by $F_X = d(X)|_{\alpha_{\eta,\rho}(X)}$, and $\rho' = \rho_0\{\rho\{n/W'\}/\text{H}_\eta(L')\}$. \square

The probabilistic transition system is built from the states of the stochastic transition scheme coupled with a sample environment. The samples are used to calculate the aged distributions of the racing delays. The race induces a probabilistic choice, which is normalized by the accumulative probability of the outgoing stochastic delay transitions. The normalization is required in case of incomplete races. Each probabilistic timed delay transition updates the sample environment by first assigning the winning sample to the winners. Afterwards, only the meaningful part of the environment given by the racing history of the losers is retained. We illustrate the situation by an example.

Example 6.5.3 We depict a stochastic transition scheme as in Figure 6.3a. It is very similar to the racing timed transition scheme, differing on the interpretation of the stochastic transitions. The environment now holds the

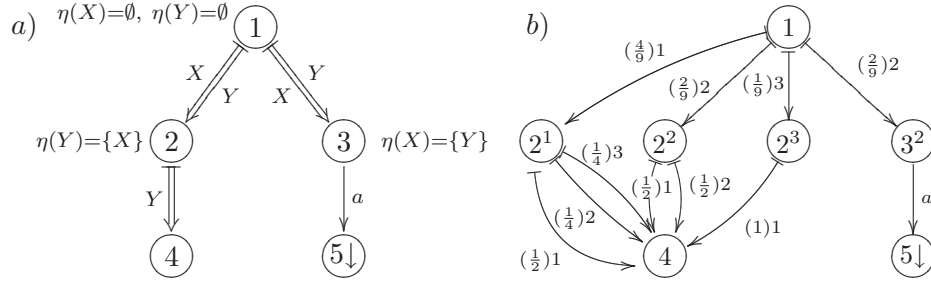


Figure 6.3: Stochastic transition scheme and an induced probabilistic timed transition system

racing history of the delays instead of the age of the racing delays. In Figure 6.3b we depict the induced probabilistic timed transition system, where the assignment of distributions to the delays X and Y is as in Example 2.1.1 and the initial age of the delays is assumed to be zero. We have suppressed the presentation of the environment for the sake of clarity of presentation.

The distributions of X and Y are given by $P(X = 1) = P(X = 2) = P(X = 3) = \frac{1}{3}$ and $P(Y = 2) = \frac{1}{2}, P(Y = 3) = P(Y = 4) = \frac{1}{4}$. Again the race in state 1 is incomplete and normalization is required. Recall that the probability that $P(X < Y) = \frac{7}{12}$ and $P(Y < X) = \frac{2}{12}$, so the probabilities are normalized to $\frac{7}{9}$ and $\frac{2}{9}$, respectively. These probabilities can be multiplied with the conditional probabilities that the winner X expires in 1, 2, or 3 time steps and, respectively, that Y wins the race in 2 time steps as depicted in Figure 6.3b. This is an alternative way of computing the probabilities in addition to the one given in Definition 6.5.2.

Stochastic delays generally introduce multiple and sometimes infinitely many timed transitions, depending on the support set of the distribution. In the superscript of the states, we put the duration of the stochastic delay with which that state has been reached. So, for example, state 2^2 is reached after the delay X won the race against the delay Y with a duration of 2 time units. This occurs with probability $\frac{2}{9}$ as stated on the transition between state 1 and 2^2 . In state 2^2 , the environment contains $\eta(Y) = \{X\}$ and $\rho(X) = 2$. Thus, the total age of Y is 2 and its residual distribution is computed as $Y' = \langle Y - 2 \mid X < Y, X = 2 \rangle$, i.e., $P(Y' = 1), P(Y' = 2) = \frac{1}{2}$. \square

We note that passive delay transitions can be denoted (by self-loops) of infinitely long timed transitions with probability 1. However, we do not typ-

ically consider such transitions due to prioritization of synchronized actions.

Next, we give the bisimulation relation in the vein of Definition 3.3.1 for the timed setting.

6.6 Bisimulation

We define a strong bisimulation relation on stochastic transition schemes that requires stochastic delays to have the same dependence history modulo names of the independent delays. It is the counterpart of the racing timed bisimulation relation for the stochastic time setting. As before, the condition for stochastic delays ensures that the induced races have the same probabilistic behavior by relating only independent delays with the same distributions. In the current setting, we must also account for the behavior of the passive delay transitions.

Definition 6.6.1 Let $R \subseteq (S \times \mathcal{H})^2 \times (\mathcal{V} \leftrightarrow \mathcal{V})$ be a symmetric relation. Then R is a stochastic bisimulation relation if for all $(\langle s_1, \eta_1 \rangle, \langle s_2, \eta_2 \rangle, r) \in R$ it holds that $r: H_{\eta_1}(R(s_1)) \leftrightarrow H_{\eta_2}(R(s_2))$ is a bijection with $r(I(s_1)) = I(s_2)$, and $F_X = F_{r(X)}$ and $r(\eta_1(X)) = \eta_2(r(X))$ for $X \in \text{dom}(r)$, and:

1. if $u_1 \downarrow$ then $u_2 \downarrow$;
2. if $u_1 \rightsquigarrow u'_1$ for some $u'_1 \in S \times \mathcal{H}$, then $u_2 \rightsquigarrow u'_2$ for some $u'_2 \in S \times \mathcal{H}$ such that $(u'_1, u'_2, r') \in R$ for some $r' \in \mathcal{V} \leftrightarrow \mathcal{V}$;
3. if $u_1 \xrightarrow{a} u'_1$ for some $u'_1 \in S \times \mathcal{H}$, then $u_2 \xrightarrow{a} u'_2$ for some $u'_2 \in S \times \mathcal{H}$ such that $(u'_1, u'_2, r') \in R$ for some $r' \in \mathcal{V} \leftrightarrow \mathcal{V}$; and
4. if $u_1 \xrightarrow[L_1]{W_1} u'_1$ for some $u'_1 = \langle u'_1, \eta'_1 \rangle \in S \times \mathcal{H}$, then $u_2 \xrightarrow[L_2]{W_2} u'_2$ for some $u'_2 = \langle u'_2, \eta'_2 \rangle \in S \times \mathcal{H}$ where $r(W_1) = W_2$, $r(L_1) = L_2$, and $(u'_1, u'_2, r') \in R$ for some $r' \in \mathcal{V} \leftrightarrow \mathcal{V}$ satisfying $r'(X) = r(X)$ for $X \in H_{\eta_1}(L_1 \cap D(s'_1))$.

We say that two states u_1 and u_2 are stochastic bisimilar, notation $u_1 \Leftrightarrow_s u_2$, if there exists a stochastic bisimulation relation R such that $(u_1, u_2, r) \in R$ for some $r \in \mathcal{V} \leftrightarrow \mathcal{V}$. \square

The bisimulation relation is adapted for the stochastic setting by generalizing the environment from an age of the distribution to a racing history of expired winning delays. As the history also depends on the names of the delays, the bisimulation also must cater for the consistency of the complete history of the losers. This is expressed by the last condition $r'(X) = r(X)$ for $X \in H_{\eta_1}(L_1 \cap D(s'_1))$. The extension is pretty straightforward and the

$$\begin{aligned}
C_\eta(\underline{1}) &= C_\eta(\underline{0}) = C_\eta(\underline{a}.p) = C_\eta(\bar{a}.p) = \emptyset \\
C_\eta([\frac{W}{L}].p) &= L \cap I(p) \\
C_\eta(|p|_D) &= C_\eta(\partial_H(p)) = C_\eta(\theta_I(p)) = C_\eta(p) \\
C_\eta(p_1 + p_2) &= C_\eta(p_1 \parallel p_2) = C_\eta(p_1) \cup C_\eta(p_2) \cup \\
&\quad ((I(p_1) \cup N(p_1)) \cap H_\eta(R(p_2))) \cup (H_\eta(R(p_1)) \cap (I(p_2) \cup N(p_2))).
\end{aligned}$$

Table 6.1: Set of conflicting names in an environment η

proofs from TCP^{drst} naturally extend to the new setting. The following theorem states without proof that stochastic bisimilarity is an equivalence relation.

Theorem 6.6.2 *Stochastic bisimilarity is an equivalence relation.* □

We continue with the presentation of the operational semantics.

6.7 Structural Operational Semantics

The operational semantics of $\text{TCP}_{\text{rec}}^{\text{st}}$ has the same impediments as the one of $\text{TCP}_{\text{rec}}^{\text{drst}}$. Again, for a closed term $p \in \mathcal{C}(\text{TCP}_{\text{rec}}^{\text{st}})$ to have proper semantics, the conflicting independent racing delay names have to be detected and renamed. We use the already established notions of dependent racing, independent racing, dependence binding, and newly enabled independent delay names to identify the conflicting names and set up α -conversion. As the environment holds the racing history in terms of stochastic delay names, the complete history has to be included in the detection of naming conflicts as well. We give a simple example to illustrate the situation.

Example 6.7.1 Let $[X].[Z].\underline{0} + [Y].\underline{0}$ be a term in an environment η with $\eta(Y) = \{Z\}$ and $\eta(Z) = \{U\}$. If $[X]$ wins the race, the resulting term is $[Z].\underline{0} + [Y].\underline{0}$ with $\eta(Y) = \{X, Z\}$ and $\eta(Z) = \{U\}$. Now, the conflict arises because $[Z]$ is a newly enabled independent delay, but because of the racing history of $[Y]$ it has been wrongly made dependent on the sample of U . □

We denote the environment as a subscript and we extract the set of conflicting names $C_\eta(p)$ of $p \in \mathcal{C}(\text{TCP}^{\text{drst}})$ in an environment η as in Table 6.1.

For notational convenience we write η_\emptyset for $\eta_\emptyset(X) = \emptyset$ for $X \in \mathcal{V}$. By $\eta + W$ we denote $(\eta + W)(X) = \eta(X) \cup \{Y\}$ for $X \in \mathcal{V}$, a non-empty set

$W \subseteq \mathcal{V}$, and a randomly chosen $Y \in W$. The notational conventions $\not\Rightarrow$ and $\not\rightsquigarrow$ express that the term does not have any outgoing stochastic delay or passive delay transitions, respectively. Now that we have all prerequisites we give the structural operational semantics in Table 6.2 for the constant processes, the prefix operators, and the dependence scope, Table 6.3 for the alternative composition, Table 6.4 for the parallel composition, and Table 6.5 for the encapsulation and maximal progress operator, and the recursion. We comment upon some of the rules that are different from the ones for $\text{TCP}_{\text{rec}}^{\text{drst}}$.

6.6 $\frac{}{\langle \underline{1}, \eta \rangle \downarrow}$	6.7 $\frac{}{\langle \bar{0}, \eta \rangle \rightsquigarrow \langle \bar{0}, \eta \rangle}$	
6.8 $\frac{}{\langle \underline{a}.p, \eta \rangle \xrightarrow{a} \langle p _{\emptyset}, \eta_{\emptyset} \rangle}$	6.9 $\frac{}{\langle \bar{a}.p, \eta \rangle \xrightarrow{a} \langle p _{\emptyset}, \eta_{\emptyset} \rangle}$	6.10 $\frac{}{\langle \bar{a}.p, \eta \rangle \rightsquigarrow \langle \bar{a}.p, \eta \rangle}$
6.11 $\frac{}{\langle [L].p, \eta \rangle \xrightarrow[L]{W} \langle p _L, \eta_{\emptyset} \{ \eta' / H_{\eta'}(L) \} \rangle}$ with $\eta' = \eta \{ (\eta + W) / L \}$		
6.12 $\frac{\langle p, \eta \rangle \downarrow}{\langle p _D, \eta \rangle \downarrow}$	6.13 $\frac{\langle p, \eta \rangle \xrightarrow{a} \langle p', \eta' \rangle}{\langle p _D, \eta \rangle \xrightarrow{a} \langle p', \eta' \rangle}$	
6.14 $\frac{\langle p, \eta \rangle \rightsquigarrow \langle p', \eta \rangle}{\langle p _D, \eta \rangle \rightsquigarrow \langle p', \eta \rangle}$	6.15 $\frac{\langle p, \eta \rangle \xrightarrow[L]{W} \langle p', \eta' \rangle}{\langle p _D, \eta \rangle \xrightarrow[L]{W} \langle p', \eta' \rangle}$	

Table 6.2: Structural operational semantics of $\text{TCP}_{\text{rec}}^{\text{st}}$ for the constants $\underline{1}$ and $\bar{0}$, the prefix operators, and the dependence scope operator

Rule 6.7 states that delayable deadlock constant has an outgoing passive delay transition. Rule 6.8 states that undelayable action prefixes perform only undelayable action transitions. Rules 6.9 and 6.10 state that delayable action prefixes induce both an undelayable action and a passive delay transition. Rule 6.11 enables stochastic delay transitions. The environment is updated in two phases. First, the dependence sets of the losers are updated resulting in the new environment η' . Afterwards, only the relevant dependence history of the losers, given by $H_{\eta'}(L)$, is retained. The losers in resulting term $|p|_L$ are treated as dependent as their names must be protected. Again, the dependence scope operator does not affect any transitions as illustrated by the rules 6.12 – 6.15 and it is only used to specify dependent and independent racing delay names.

6.16 $\frac{\langle p_1, \eta \rangle \downarrow}{\langle p_1 + p_2, \eta \rangle \downarrow}$	6.17 $\frac{\langle p_2, \eta \rangle \downarrow}{\langle p_1 + p_2, \eta \rangle \downarrow}$
6.18 $\frac{\langle p_1, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle}{\langle p_1 + p_2, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle}$	6.19 $\frac{\langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 + p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle}$
6.20 $\frac{\langle p_1, \eta \rangle \rightsquigarrow \langle p'_1, \eta \rangle, \langle p_2, \eta \rangle \not\Leftarrow, \langle p_2, \eta \rangle \not\Leftarrow}{\langle p_1 + p_2, \eta \rangle \rightsquigarrow \langle p'_1, \eta \rangle}$	6.21 $\frac{\langle p_1, \eta \rangle \not\Leftarrow, \langle p_2, \eta \rangle \not\Leftarrow, \langle p_2, \eta \rangle \rightsquigarrow \langle p'_2, \eta \rangle}{\langle p_1 + p_2, \eta \rangle \rightsquigarrow \langle p'_2, \eta \rangle}$
6.22 $\frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \not\Leftarrow, \langle p_2, \eta \rangle \not\Leftarrow}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle}$	6.23 $\frac{\langle p_1, \eta \rangle \not\Leftarrow, \langle p_2, \eta \rangle \not\Leftarrow, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}$
6.24 $\frac{\langle p_1, \eta \rangle \rightsquigarrow \langle p'_1, \eta \rangle, \langle p_2, \eta \rangle \rightsquigarrow \langle p'_2, \eta \rangle}{\langle p_1 + p_2, \eta \rangle \rightsquigarrow \langle p'_1 + p'_2, \eta \rangle}$	
6.25 $\frac{\langle p_1, \eta \rangle \rightsquigarrow \langle p'_1, \eta \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_1 + p'_2, \eta_2 \rangle}$	6.26 $\frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \rightsquigarrow \langle p'_2, \eta \rangle}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1 + p'_2, \eta_1 \rangle}$
6.27 $\frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle, W_1 \cap (W_2 \cup L_2) = \emptyset}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_1}_{L_1 \cup W_2 \cup L_2} \langle p'_1 + p_2, \eta_0 \{ \eta' / H_{\eta'} (L_1 \cup W_2 \cup L_2) \} \rangle}$, with $\eta' = \{ (\eta + W_1) / L_1 \cup W_2 \cup L_2 \}$	
6.28 $\frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle, (W_1 \cup L_1) \cap W_2 = \emptyset}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_1}_{W_1 \cup L_1 \cup L_2} \langle p_1 + p'_2, \eta_0 \{ \eta' / H_{\eta'} (W_1 \cup L_1 \cup L_2) \} \rangle}$, with $\eta' = \{ (\eta + W_2) / W_1 \cup L_1 \cup L_2 \}$	
6.29 $\frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle, (W_1 \cup W_2) \cap (L_1 \cup L_2) = \emptyset}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_1 \cup W_2}_{L_1 \cup L_2} \langle p'_1 + p'_2, \eta_1 \{ \eta_2 / L_2 \} \rangle}$	
6.30 $\frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \text{rr}([L_1^{W_1}], [L_2^{W_2}]) \text{ for } \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle}$	
6.31 $\frac{\text{rr}([L_1^{W_1}], [L_2^{W_2}]) \text{ for } \langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 + p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}$	

Table 6.3: Operational rules for $\text{TCP}_{\text{rec}}^{\text{st}}$ for the alternative composition

Rules 6.16–6.19 are as before. Rules 6.20 – 6.23 illustrates the default weak choice between action transitions and passage of time. Here, we have two transitions that denote passage of time, so both of them must be disabled in the other term. Rule 6.24 states that passive delay transitions merge. Rules 6.25 and 6.26 state that passive passage of time synchronizes with stochastic delays. Resolution of races is given by the rules 6.27 – 6.29. Unlike the timed delays that had predetermined racing context, stochastic delays resolve the races dynamically. The environment in rules 6.27 and 6.28 is again updated in two phases as for rule 6.11, but now with the joint set of losers obtained by resolving the race. When the delays have winners that exhibit the same sample, the resulting environment is a merger of the resulting environments as given by rule 6.29. As in the timed setting, rules 6.30 and 6.31 express that a stochastic delay transition of one summand is in a resolved race if its racing delays are in a resolved race with the ones of every outgoing stochastic delays of the other summand.

Rules 6.32 – 6.37 give the standard behavior for the parallel composition for termination and action transitions as for $\text{TCP}_{\text{rec}}^{\text{drst}}$. Rule 6.38 gives the synchronization of the passive delay transitions, whereas rules 6.39 and 6.40 give the synchronization of the passive and stochastic delay transitions as for the alternative composition. Rules 6.41 – 6.43 show the resolution of races analogous to the ones for the alternative composition. Again, resolved races are not possible as they represent disjoint events that cannot occur simultaneously.

Rules 6.44 – 6.47 express the standard behavior for the encapsulation operator. It suppresses only unwanted actions, whereas it simply propagates through the other transitions. Rules 6.48 – 6.51 show the behavior of the maximal progress operator. Rules 6.50 and 6.51 state that passage of time is enabled only if there are no prioritized outgoing action transitions. Finally, rules 6.52 – 6.55 are standard for guarded recursion, enabling the solution to have the same transitions as given by the specification.

The renaming of conflicting independent delay names is again performed by means of α -conversion, which is defined as for $\text{DTCP}_{\text{rec}}^{\text{dst}}$. The bisimilarity relation is given in the same vein as for $\text{TCP}_{\text{rec}}^{\text{drst}}$ requiring that dependent delay names are respected.

Definition 6.7.2 Two terms $p_1, p_2 \in \mathcal{C}(\text{TCP}_{\text{rec}}^{\text{st}})$ are stochastic bisimilar if there exists a stochastic bisimulation relation R with $(\langle p_1, \eta_\emptyset \rangle, \langle p_2, \eta_\emptyset \rangle, r) \in R$ for some $r \in \mathcal{V} \leftrightarrow \mathcal{V}$ satisfying $r(X) = X$ for $X \in \text{D}(p_1)$. \square

As before, the definition does not impose a restriction on the use of environments because a result analogous to Lemma 3.7.2 holds. It should also come

$$\begin{array}{c}
\mathbf{6.32} \frac{\langle p_1, \eta \rangle \downarrow, \langle p_2, \eta \rangle \downarrow}{\langle p_1 \parallel p_2, \eta \rangle \downarrow} \\
\mathbf{6.33} \frac{\langle p_1, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \not\Rightarrow}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{a_1} \langle p'_1 \parallel p_2, \eta_1 \rangle} \quad \mathbf{6.34} \frac{\langle p_1, \eta \rangle \not\Rightarrow, \langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{a_2} \langle p_1 \parallel p'_2, \eta_2 \rangle} \\
\mathbf{6.35} \frac{\langle p_1, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{a_1} \langle p'_1 \parallel p_2, \eta \rangle} \\
\mathbf{6.36} \frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{a_2} \langle p_1 \parallel p'_2, \eta \rangle} \\
\mathbf{6.37} \frac{\langle p_1, \eta \rangle \xrightarrow{a_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{a_2} \langle p'_2, \eta_2 \rangle, \gamma(a_1, a_2) = a_3}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{a_3} \langle p'_1 \parallel p'_2, \eta_0 \rangle} \\
\mathbf{6.38} \frac{\langle p_1, \eta \rangle \rightsquigarrow \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \rightsquigarrow \langle p'_2, \eta_2 \rangle}{\langle p_1 \parallel p_2, \eta \rangle \rightsquigarrow \langle p'_1 \parallel p'_2, \eta \rangle} \\
\mathbf{6.39} \frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \rightsquigarrow \langle p'_2, \eta \rangle}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1 \parallel p'_2, \eta_1 \rangle} \\
\mathbf{6.40} \frac{\langle p_1, \eta \rangle \rightsquigarrow \langle p'_1, \eta \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_1 \parallel p'_2, \eta_2 \rangle} \\
\mathbf{6.41} \frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle, W_1 \cap (W_2 \cup L_2) = \emptyset}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{W_1}_{L_1 \cup W_2 \cup L_2} \langle p'_1 \parallel p_2, \eta_0 \{ \eta' / H_{\eta'}(L_1 \cup W_2 \cup L_2) \} \rangle}, \\
\text{with } \eta' = \{(\eta + W_1) / L_1 \cup W_2 \cup L_2\} \\
\mathbf{6.42} \frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle, (W_1 \cup L_1) \cap W_2 = \emptyset}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{W_2}_{W_1 \cup L_1 \cup L_2} \langle p_1 \parallel p'_2, \eta_0 \{ \eta' / H_{\eta'}(W_1 \cup L_1 \cup L_2) \} \rangle}, \\
\text{with } \eta' = \{(\eta + W_2) / W_1 \cup L_1 \cup L_2\} \\
\mathbf{6.43} \frac{\langle p_1, \eta \rangle \xrightarrow{W_1}_{L_1} \langle p'_1, \eta_1 \rangle, \langle p_2, \eta \rangle \xrightarrow{W_2}_{L_2} \langle p'_2, \eta_2 \rangle, (W_1 \cup W_2) \cap (L_1 \cup L_2) = \emptyset}{\langle p_1 \parallel p_2, \eta \rangle \xrightarrow{W_1 \cup W_2}_{L_1 \cup L_2} \langle p'_1 \parallel p'_2, \eta_1 \{ \eta_2 / L_2 \} \rangle}
\end{array}$$

Table 6.4: Operational rules for the parallel composition

$$\begin{array}{c}
\begin{array}{cc}
\mathbf{6.44} \frac{\langle p, \eta \rangle \downarrow}{\langle \partial_H(p), \eta \rangle \downarrow} & \mathbf{6.45} \frac{\langle p, \eta \rangle \xrightarrow{a} \langle p', \eta' \rangle, a \notin H}{\langle \partial_H(p), \eta \rangle \xrightarrow{a} \langle \partial_H(p'), \eta' \rangle} \\
\mathbf{6.46} \frac{\langle p, \eta \rangle \rightsquigarrow \langle p', \eta \rangle}{\langle \partial_H(p), \eta \rangle \rightsquigarrow \langle \partial_H(p'), \eta \rangle} & \mathbf{6.47} \frac{\langle p, \eta \rangle \xrightarrow[L]{W} \langle p', \eta' \rangle}{\langle \partial_H(p), \eta \rangle \xrightarrow[L]{W} \langle \partial_H(p'), \eta' \rangle} \\
\mathbf{6.48} \frac{\langle p, \eta \rangle \downarrow}{\langle \theta_I(p), \eta \rangle \downarrow} & \mathbf{6.49} \frac{\langle p, \eta \rangle \xrightarrow{a} \langle p', \eta' \rangle}{\langle \theta_I(p), \eta \rangle \xrightarrow{a} \langle \theta_I(p'), \eta' \rangle} \\
\mathbf{6.50} \frac{\langle p, \eta \rangle \rightsquigarrow \langle p', \eta \rangle, \langle p, \eta \rangle \xrightarrow{a} \text{ for } a \in I}{\langle \theta_I(p), \eta \rangle \rightsquigarrow \langle \theta_I(p'), \eta \rangle} & \\
\mathbf{6.51} \frac{\langle p, \eta \rangle \xrightarrow[L]{W} \langle p', \eta' \rangle, \langle p, \eta \rangle \xrightarrow{a} \text{ for } a \in I}{\langle \theta_I(p), \eta \rangle \xrightarrow[L]{W} \langle \theta_I(p'), \eta' \rangle} & \\
\mathbf{6.52} \frac{\langle p, \eta \rangle \downarrow, A = p \in S}{\langle \mu A.S, \eta \rangle \downarrow} & \mathbf{6.53} \frac{\langle p, \eta \rangle \xrightarrow{a} \langle p', \eta' \rangle, A = p \in S}{\langle \mu A.S, \eta \rangle \xrightarrow{a} \langle p', \eta' \rangle} \\
\mathbf{6.54} \frac{\langle p, \eta \rangle \rightsquigarrow \langle p', \eta' \rangle, A = p \in S}{\langle \mu A.S, \eta \rangle \rightsquigarrow \langle p', \eta' \rangle} & \mathbf{6.55} \frac{\langle p, \eta \rangle \xrightarrow[L]{W} \langle p', \eta' \rangle, A = p \in S}{\langle \mu A.S, \eta \rangle \xrightarrow[L]{W} \langle p', \eta' \rangle}
\end{array}
\end{array}$$

Table 6.5: Structural operational semantics of $\text{TCP}_{\text{rec}}^{\text{st}}$ for the encapsulation operator, the maximal progress operator, and recursion

as no surprise that stochastic bisimilarity is a congruence for $\text{TCP}_{\text{rec}}^{\text{st}}$. The proof is along the same lines as the one for Theorem 3.9.1 and, therefore, it is omitted.

Theorem 6.7.3 *Stochastic bisimilarity \rightleftharpoons_s is a congruence on $\mathcal{C}(\text{TCP}_{\text{rec}}^{\text{st}})$.* \square

Supported by Theorem 6.7.3 we give a term model modulo stochastic bisimilarity.

Definition 6.7.4 The term model of $\text{TCP}_{\text{rec}}^{\text{st}}$ is the quotient algebra $\mathbb{P}(\text{TCP}_{\text{rec}}^{\text{st}})/\rightleftharpoons_s$, where $\mathbb{P}(\text{TCP}_{\text{rec}}^{\text{st}}) = (\mathcal{C}(\text{TCP}_{\text{rec}}^{\text{st}}), \underline{0}, \underline{1}, \overline{0}, \mu A.S \text{ for } S \in \mathcal{G} \text{ and } A \in \mathcal{R}(S), \underline{a}._ \text{ for } a \in \mathcal{A}, \overline{a}._ \text{ for } a \in \mathcal{A}, [^W_L]._ \text{ for } W, L \subseteq \mathcal{V}, \text{ satisfying } W \neq \emptyset \text{ and } W \cap L = \emptyset, | _ |_D \text{ for } D \subseteq \mathcal{V}, \partial_H(_) \text{ for } H \subseteq \mathcal{A}, \theta_I(_) \text{ for } I \subseteq \mathcal{A}, - + _, - \| _)$. \square

The equational theory of $\text{TCP}_{\text{rec}}^{\text{st}}$ coincides with the one of $\text{DTCP}_{\text{rec}}^{\text{dst}}$. Moreover, the main results carry over to the new setting and the theory is ground-

complete in $\text{TCP}_{\text{rec}}^{\text{st}}$ as well.

Next, we compare treatment of the passage of time with the other stochastic formalisms by discussing the expansion of the parallel composition.

6.8 Expansion of the Parallel Composition

First, we give an abstract description of the expansion of the parallel composition in clock-based approaches [42, 26, 60, 28] that employ start-termination semantics. There, the stochastic delay $[X]$ is split into a starting (X^+) and an ending (X^-) activity, which are then treated as normal undelayable action transitions. Intuitively $[X].p = X^+.X^-.p$, and the expansion of $[X].p \parallel [Y].q$ is given by

$$X^+.X^-.p \parallel Y^+.Y^-.q = X^+.Y^+.(X^-.p \parallel Y^-.q) + Y^+.X^+.(X^-.p \parallel Y^-.q).$$

This allows an expansion law that is much more elegant than our Theorem 5.8.1. For comparison purposes, we present the expansion of the parallel composition in SPADES, which employs clocks with residual lifetime semantics [42]. The treatment of the expansion for clocks with spent lifetimes and start-termination semantics is similar [26, 28].

To present the parallel composition in SPADES we give the normal form of two processes x and y : $x = \text{set } C \text{ in } x'$ and $y = \text{set } D \text{ in } y'$, for $x' = \sum_{i=1}^m (\text{when } C_i \mapsto a_i; p_i)$ and $y' = \sum_{j=1}^n (\text{when } D_j \mapsto b_j; q_j)$. The operator set sets the clocks, ' $a; _$ ' is the action prefix operator, and $\text{when } C \mapsto p$ is the guard that enables the process p when all clocks in the set C have expired. The expansion of CSP style parallel composition $x \parallel_A y$ for a synchronization set A , is given by $x \parallel_A y =$

$$\text{set } (C \cup D) \text{ in } \left(\sum_{a_i \notin A} \text{when } C_i \mapsto a_i; (p_i \parallel_A y') + \sum_{b_j \notin A} \text{when } D_j \mapsto b_j; (x' \parallel_A q_j) + \sum_{a_i = b_j \in A} \text{when } (C_i \cup D_j) \mapsto a_i; (p_i \parallel_A q_j) \right).$$

Such treatment only involves the setting of the joint sets of clocks, i.e., the enabling of the starting activities. There is no relation between the passage of time of the components as in standard real-time semantics, where the expansion of $t.p \parallel s.q$ is given by

$$t.p \parallel s.q = \min(t, s).((t - \min(t, s)).p \parallel (s - \min(t, s)).q)$$

provided that zero duration delays are allowed. As a consequence, the maximal progress operator cannot be handled explicitly as there is no knowledge

about the relationship between the samples of the clocks in the race. This leads to more complicated definitions of the bisimulation relations, which must account for the priority of the internal actions [42, 26, 64, 22].

Finally, the explicit treatment of the race condition in the stochastic transition schemes corresponds to the regional trees that are used in preliminary attempts to model check stochastic automata (albeit in residual lifetime semantics) [29]. Originally, the regional trees were obtained from stochastic automata [41] by explicitly ordering clock samples by their duration as symbolically represented by the stochastic delay prefix.

Next, we discuss the embedding of real time in a stochastic setting by means of Dirac stochastic delays.

6.9 Embedding Real Time as Dirac Stochastic Time

A natural embedding of real time in a stochastic setting is by means of Dirac (or degenerated) stochastic delays. These delays are guided by Dirac random variables X_n , where $P(X_n = n) = 1$. The Dirac delays can be included in the theory as separate stochastic delay prefixes. The duration of the Dirac delay is stated in the subscript.

Such direct inclusion of real time in the stochastic setting has a side effect, viz. the stochastic transition schemes may contain non-accessible transitions. For example, the transition

$$\langle [X_m].p + [X_{m+n}].q, \eta_0 \rangle \xrightarrow[X_m]{X_{m+n}} \langle [X_m].p + q, \eta_0 \{X_{m+n}/\{X_m\}\} \rangle$$

will never be observed in the probabilistic timed transition system for $m, n > 0$. Similarly, the only transition with non-zero probability of $[[X_n].p]_\emptyset + [[Y_n].q]_\emptyset$ is the joint stochastic delay transition with winners $\{X_n, Y_n\}$. Moreover, there is need to distinguish between independent and dependent Dirac delays because of the resolution of the race condition. For example, the age of the Dirac delay $[Y_n]$ in $[\overset{X}{Y_n}].[Y_n].\underline{0}$ is dependent on the sample of the winner $[X]$. In this case, the aged distribution of $[Y_n]$ in the subterm $[Y_n].\underline{0}$ is no longer Dirac.

Also, it should be clear that the concept of time additivity does not apply to the Dirac stochastic delays because of the race condition semantics (cf. Section 6.1). Actually, the treatment of timed delays as stochastic Dirac delays actually leads back to the notion of context-sensitive interpolation. Thus, the embedding of real time as Dirac stochastic time can be done by restricting the standard notion of time additivity by the new notion of

context-sensitive interpolation. In such a setting Dirac delays support time determinism, and moreover, the side-effects from above do not occur.

We will not develop the complete embedding of real-time delays as Dirac stochastic delays, but we only give and briefly discuss the fingerprint axioms for race-complete processes. The additional axioms for Dirac delays that enable context-sensitive interpolation are given in Table 6.6.

$$|[X_n].p_1|_\emptyset + |[X_n].p_2|_\emptyset = |[X_n].(p_1|_\emptyset + p_2|_\emptyset)|_\emptyset \quad \mathbf{A6.3}$$

$$|[X_n].p_1|_\emptyset + |[X_{n+m}].p_2|_\emptyset = |[X_n].(p_1|_\emptyset + |[X_m].p_2|_\emptyset)|_\emptyset \quad \mathbf{A6.4}$$

Table 6.6: Axioms for the context-sensitive interpolation of Dirac delays in race-complete process specifications

The axioms are very similar to their real-time counterparts given by the axioms A6.1 and A6.2. However, there is an extra condition that the Dirac delays must be independent. This condition plays an important role because it ensures that the age of the Dirac delays is zero. On the contrary, it is possible that the Dirac delay is dependent on a stochastic delay as in the term $[\tilde{v}_n^x].[Y_n].\underline{0}$ in the example above.

6.10 Summary

We take the viewpoint of stochastic time and attempt to interpret the concepts of time determinism and time additivity in race condition semantics. This leads us to the notion of context-sensitive interpolation that is a restriction of time additivity in race condition semantics. As timed delays can be interpreted as either dependent or independent stochastic delays, the resolution of timed delays employing context-sensitive interpolation turns out to be a valuable tool. Then, we develop a theory of communicating processes with stochastic time from scratch, following the guidelines set up in the previous chapter. We embed standard timed delays into the theory, which leads us again to context-sensitive interpolation due to the nature of stochastic time. Finally, we look closer at this new notion and we provide the identifying rules and axioms.

Next, we turn back to Markovian time and we give means to show that the reduction methods for elimination of probabilistic choices and nondeterministic (silent) transitions are correct. We also investigate the relational and compositional properties of two aggregation techniques based on lumping and reduction.

Chapter 7

Aggregation Methods for Markov Reward Chains with Fast and Silent Transitions

Compositionality is a central issue in the theory of concurrent processes. Discussing compositionality requires three ingredients: (1) a class of processes or models, (2) an operation to compose processes, and (3) a notion of behaviour, usually given by a semantic preorder or equivalence relation on the class of processes. For the purpose of this thesis, we will have semantic preorders and the parallel composition as operation. Therefore, the compositionality result can be stated as

$$P_1 \geq \bar{P}_1, P_2 \geq \bar{P}_2 \text{ implies } P_1 \parallel P_2 \geq \bar{P}_1 \parallel \bar{P}_2,$$

where $P_1, P_2, \bar{P}_1,$ and \bar{P}_2 are arbitrary processes and \parallel and \geq denote their parallel composition and the semantic preorder relation, respectively. Hence, compositionality enables the narrowing of a parallel composition by composing simplifications of its components, thus avoiding the construction of the actual parallel system. In this chapter, we study compositionality for augmented types of continuous-time Markov chains. Here, we note that even though the exponential distribution that guides the delays is continuous, the model has very close ties to and an alternative representation in discrete time [58].

Homogeneous continuous-time Markov chains, Markov chains for short, are among the most important and wide-spread analytical performance models. A Markov chain is given by a graph with nodes representing states and outgoing arrows labelled by exponential rates determining the stochastic behavior of each state. An initial probability vector indicates which states may act as starting ones. Markov chains often come equipped with rewards

that are used to measure their performance, such as throughput, utilization, etc. (cf. [57]). In this thesis, we focus on state rewards only, and we refer to a Markov chain with rewards as a Markov reward chain. Transition (impulse) rewards [57] can be dealt with similarly. A state reward is a number associated to a state, representing the rate at which gain is received while the process resides in the state.

To cope with the ever growing complexity of the systems, several performance modeling techniques have been developed to support the compositional generation of Markov reward chains. This includes stochastic process algebras [51, 55], (generalized) stochastic Petri nets [3, 38], probabilistic I/O automata [98, 36], stochastic automata networks [86], etc. The compositional modeling enables composing a bigger system from several smaller components. The size of the state space of the resulting system is in the range of the product of the sizes of the constituent state spaces. Hence, compositional modeling usually suffers from state space explosion.

In the process of compositional modeling, performance evaluation techniques produce intermediate constructs that are typically extensions of Markov chains featuring transitions with communication labels [51, 55, 3, 38, 98, 36, 86]. In the final modeling phase, all labels are discarded and communication transitions are assigned instantaneous behavior. Previous work [75, 78, 94] gave an account of handling these models by using Markov reward chains with fast transitions and Markov reward chains with silent transitions. The former present extensions of the standard Markov reward chains with transitions decorated with a real-valued linear parameter and in the latter the real-valued linear parameter is not specified. To capture the intuition that the labeled transitions are instantaneous, a limit for the parameter to infinity is taken. The resulting process is a generalization of the standard Markov reward chain that can perform infinitely many transitions in a finite amount of time. This model was initially studied in [45, 39] without rewards, and it is called a (stochastically) discontinuous Markov reward chain. The process exhibits stochastic discontinuity and is often considered pathological. However, as shown in [39, 5, 38], it proves very useful for the explanation of results.

Here, we consider discontinuous Markov reward chains, Markov reward chains with fast transitions, and Markov reward chains with silent transitions. These three models are intimately related: Markov reward chains with fast and silent transitions are used for modeling, but some notions for these processes are expressed asymptotically in terms of discontinuous Markov reward chains. A limiting process of a Markov reward chain with fast transitions is a discontinuous Markov reward chain; a Markov reward

chain with silent transitions is identified with an equivalence class of a relation \sim on Markov reward chains with fast transitions relating chains with the ‘same shape of fast transitions’. We define parallel composition of all models in the vein of standard Markov reward chains [31] using Kronecker products and sums.

As already mentioned, compositional modeling may lead to state space explosion. Current analytical and numerical methods can efficiently handle Markov reward chains with millions of states [32, 91]. However, they only alleviate the problem and many real world problems still cannot be feasibly solved. Several aggregation techniques have been proposed to reduce the state space of Markov reward chains. Ordinary lumping is the most prominent one [61, 31]. The method partitions the state space into partition classes. In each class, the states exhibit equivalent behavior for transiting to other classes, i.e., the cumulative probability of transiting to another class is the same for every state of the class. If non-trivial lumping exists, i.e., at least one partition contains more than one state, then the method produces a smaller Markov chain that retains the performance characteristics of the original one. For example, the expected reward rate at a given time is the same for the original as for the reduced, so-called lumped, process. Another lumping-based method is exact lumping [30, 31]. This method requires that each partition class of states has the same cumulative probability of transiting to every state of another class and, moreover, each state in the class has the same initial probability. The gain of exact lumping is that the probabilities of the original process can be computed for a special class of initial probability vectors by using the lumped Markov reward chain only.

A preliminary treatment of relational properties of lumping-based aggregations of Markov chains has been given in [89]. It has been shown that the notion of exact lumping is not transitive, i.e., there are processes which have exactly lumped versions that can be non-trivially exactly lumped again, but the original process cannot be exactly lumped directly to the resulting process. On the other hand, ordinary lumping of Markov reward chains is transitive and, moreover, it has a property of strict confluence. Strict confluence means that whenever a process can be lumped using two different partitions, there is always a smaller process to which the lumped processes can lump to. Coming back to our models of interest, ordinary lumping is defined for discontinuous Markov reward chains in [75, 78, 94]. Also, so-called τ -lumping is proposed for Markov reward chains with fast transitions in [75, 78, 94]. The two methods are in agreement and the situation can be pictured as in Figure 7.1.

For Markov reward chains with silent transitions, a lifting of τ -lumping to

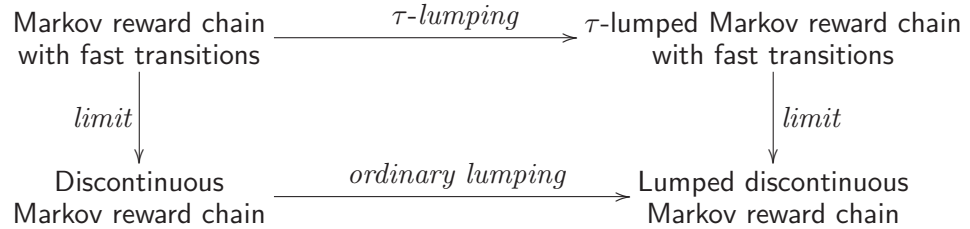


Figure 7.1: τ -lumping

the \sim -equivalence classes is proposed, referred to as τ_{\sim} -lumping [75, 78, 94]. The lifting idea is justified if the τ -lumped processes do not depend on the choice of the representative Markov reward chain with fast transitions, depicted in Figure 7.2.

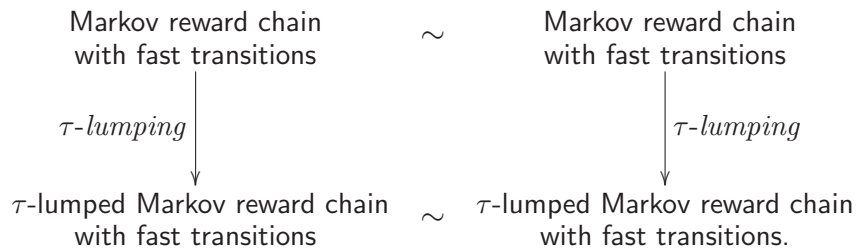


Figure 7.2: τ_{\sim} -lumping

In addition, [78, 94] study an aggregation method by reduction that eliminates the stochastic discontinuity and reduces a discontinuous Markov reward chain to a Markov reward chain. The reduction method is an extension of a well-known method in perturbation theory [44, 43, 39]. Its advantage is the ability to split states. The lumping method, in contrast, provides more flexibility: also states that do not exhibit discontinuous behavior can be aggregated. The reduction-based aggregation straightforwardly extends to τ -reduction of Markov reward chains with fast transitions [78, 94]. Therefore, we have the following situation depicted in Figure 7.3.

In the case of Markov reward chains with silent transitions, a direct lifting of the τ -reduction to equivalence classes does not aggregate many processes, as most of the time the reduced process depends on the actual fast transitions [78, 94]. In an attempt to remedy the effect of the fast transitions

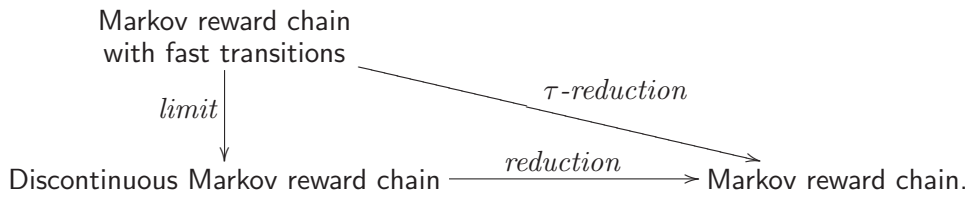


Figure 7.3: τ -reduction

we combine τ -reduction and standard ordinary lumping for Markov reward chains to obtain τ_{\sim} -reduction as depicted in Figure 7.4. We note that the method is called total τ_{\sim} -reduction in [78, 94], since there more τ_{\sim} -reduction methods are considered.

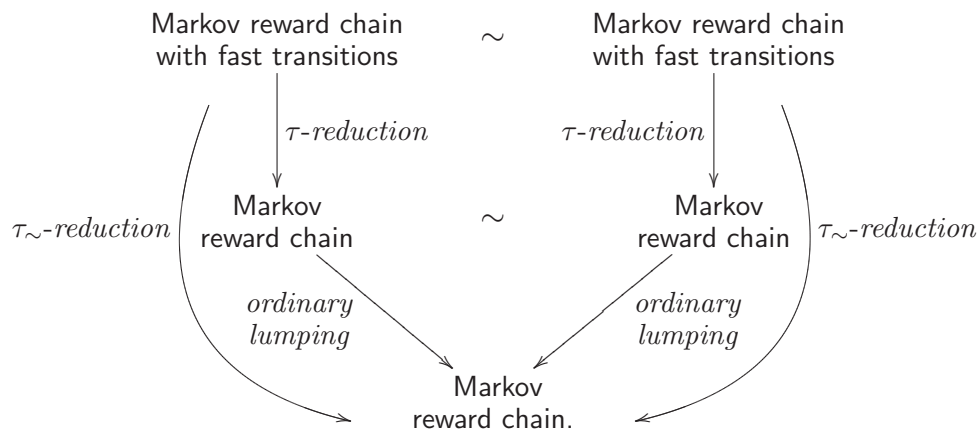


Figure 7.4: τ_{\sim} -reduction

Both the lumping-based and the reduction-based aggregation method induce semantic relations. Namely, for two processes P and \bar{P} we say that $P \geq \bar{P}$ if \bar{P} is an aggregated version of P . As already mentioned, compositionality is very important as it allows us to aggregate the smaller parallel components first, and then combine them into the aggregated complete system. We show that the relations induced by the lumping and reduction methods are indeed preorders, i.e., reflexive and transitive relations. Having all the ingredients in place, we show the compositionality of the aggregation preorders with respect to the defined parallel composition(s). We also show

continuity of the parallel composition(s). In short, the parallel operators preserve the diagrams above.

7.1 Extended Markovian Models

In this section we introduce the Markovian models studied here: discontinuous Markov reward chains as generalizations of standard Markov reward chains where infinitely many transitions can be performed in a finite amount of time; Markov reward chains with fast transitions as Markov reward chains parameterized by a real variable τ ; and Markov reward chains with silent transitions as equivalence classes of Markov reward chains with fast transitions with the same structure and unspecified ‘speeds’ of the fast transitions. The fast transitions explicitly model stochastic behavior, while the silent transitions model nondeterministic internal steps.

All vectors are column vectors if not indicated otherwise. By $\mathbf{1}^n$ we denote the vector of n 1’s; by $\mathbf{0}^{n \times m}$ the $n \times m$ zero matrix; by I^n the $n \times n$ identity matrix. We omit the dimensions n and m when they are clear from the context. By $A[i, j]$ we denote an element of the matrix $A \in \mathbb{R}^{m \times n}$ assuming $1 \leq i \leq m$ and $1 \leq j \leq n$. We write $A \geq 0$ when all elements of A are non-negative. The matrix A is called stochastic if $A \geq 0$ and $A \cdot \mathbf{1} = \mathbf{1}$. By A^T we denote the transpose of A .

Let \mathcal{S} be a finite set. A set $\mathcal{P} = \{S_1, \dots, S_N\}$ of N subsets of \mathcal{S} is called a partition of \mathcal{S} if $\mathcal{S} = S_1 \cup \dots \cup S_N$, $S_i \neq \emptyset$ and $S_i \cap S_j = \emptyset$ for all i, j , with $i \neq j$. The partitions $\{\mathcal{S}\}$ and $\Delta = \{\{i\} \mid i \in \mathcal{S}\}$ are the trivial partitions. Let $\mathcal{P}_1 = \{S_1, \dots, S_N\}$ be a partition of \mathcal{S} and $\mathcal{P}_2 = \{T_1, \dots, T_M\}$, in turn, a partition of \mathcal{P}_1 . The composition $\mathcal{P}_1 \circ \mathcal{P}_2$ of the partitions \mathcal{P}_1 and \mathcal{P}_2 is a partition of \mathcal{S} , given by $\mathcal{P}_1 \circ \mathcal{P}_2 = \{U_1, \dots, U_M\}$, where $U_i = \bigcup_{C \in T_i} C$.

In the standard theory (cf. [46, 37, 57]), Markov chains are assumed to be stochastically continuous. This means that when $t \rightarrow 0$, the probability of the process occupying at time t the same state as at time 0 is 1. As we include instantaneous transitions in our theory [39], this requirement must be dropped. Therefore, we work in the more general setting of discontinuous Markov chains originating from [45].

A discontinuous Markov reward chain is a time-homogeneous finite-state stochastic process with an associated state reward structure that satisfies the Markov property. It is completely determined by: (1) a stochastic *initial probability row vector* that gives the starting probabilities of the process for each state, (2) a *transition matrix function* $P : \mathbb{R}^+ \rightarrow \mathbb{R}^{n \times n}$ that defines the stochastic behavior of the transitions at time $t > 0$, and (3) a *state reward rate vector* that associates a number to each state representing the

gain of the process while spending time in the state. The transition matrix function gives a stochastic matrix $P(t) \geq 0$ at every time $t > 0$, and has the property $P(t+s) = P(t) \cdot P(s)$ [46, 37]. It has a convenient characterization independent of time as stated by the following proposition [39, 53, 94].

Proposition 7.1.1 *Let $(\Pi, Q) \in \mathbb{R}^{n \times n} \times \mathbb{R}^{n \times n}$ be such that*

1. $\Pi \geq 0$, $\Pi \cdot \mathbf{1} = \mathbf{1}$, $\Pi^2 = \Pi$,
2. $\Pi Q = Q \Pi = Q$,
3. $Q \cdot \mathbf{1} = \mathbf{0}$, and
4. $Q + c\Pi \geq 0$ for some $c \geq 0$.

Then $P(t) = \Pi e^{Qt}$ is a transition matrix. Moreover, for any transition matrix $P(t)$ there exists a unique pair (Π, Q) that satisfies conditions 1–4 such that $P(t) = \Pi e^{Qt}$. \square

In addition, it is known that $P(t)$ is continuous for $t > 0$ and the limit $\lim_{t \rightarrow \infty} P(t) = \Pi$ always exist [45, 46]. Then, it holds that $\Pi P(t) = P(t) \Pi = P(t)$ [39]. Proposition 7.1.1 enables us to give the following definition of a discontinuous Markov reward chain.

Definition 7.1.2 A discontinuous Markov reward chain D is a quadruple $D = (\sigma, \Pi, Q, \rho)$, where σ is a stochastic initial probability row vector, ρ is a state reward vector, and $\Pi \in \mathbb{R}^{n \times n}$ and $Q \in \mathbb{R}^{n \times n}$ satisfy the conditions of Proposition 7.1.1. The matrix function $P(t) = \Pi e^{Qt}$ is the transition matrix of D . \square

The transition matrix is continuous at zero if and only if $\Pi = I$. In this case, Q becomes the standard *generator matrix* [39, 75]. Otherwise, the matrix Q might contain negative non-diagonal entries. We note that, unlike for standard Markov reward chains, a meaningful graphical representation of discontinuous Markov reward chains when $\Pi \neq I$ is not common. The intuition behind the matrix Π is that $\Pi[i, j]$ denotes the probability that a process occupies two states via an instantaneous transition. Therefore, in case of no instantaneous transitions, i.e., when $\Pi = I$, we get a standard (stochastically continuous) Markov reward chain denoted by $M = (\sigma, Q, \rho)$.

For every discontinuous Markov reward chain $D = (\sigma, \Pi, Q, \rho)$, Π gets the following ‘ergodic’ form after a suitable renumbering of states [39], viz.

$$\Pi = \begin{pmatrix} \Pi_1 & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \dots & \Pi_M & \mathbf{0} \\ \bar{\Pi}_1 & \dots & \bar{\Pi}_M & \mathbf{0} \end{pmatrix}$$

where for all $1 \leq k \leq M$, $\Pi_k = \mathbf{1} \cdot \mu_k$ and $\bar{\Pi}_k = \delta_k \cdot \mu_k$ for a row vector $\mu_k > 0$ such that $\mu_k \cdot \mathbf{1} = 1$ and a vector $\delta_k \geq 0$ such that $\sum_{k=1}^M \delta_k = \mathbf{1}$.

The new numbering induces a partition $\mathcal{E} = \{E_1, \dots, E_M, T\}$ of the state space $\mathcal{S} = \{1, \dots, n\}$, where E_1, \dots, E_M are the ergodic classes, determined by Π_1, \dots, Π_M , respectively, and T is the class of transient states, determined by any $\bar{\Pi}_i$, $1 \leq i \leq M$. The partition \mathcal{E} is called the ergodic partition. For every ergodic class E_k , the vector μ_k is the vector of ergodic probabilities. If an ergodic class E_k contains exactly one state, then $\mu_k = (1)$ and the state is called regular. The vector δ_k contains the trapping probabilities from transient states to the ergodic class E_k .

We next discuss the behavior of a discontinuous Markov reward chain $D = (\sigma, \Pi, Q, \rho)$. It starts in a state with a probability given by the initial probability vector σ . In an ergodic class with multiple states the process spends a non-zero amount of time switching rapidly (infinitely many times) among the states. The probability that it is found in a specific state of the class is given by the vector of ergodic probabilities. The time the process spends in the class is exponentially distributed and determined by the matrix Q . In an ergodic class with a single state the row of Q corresponding to that state has the form of a row in a generator matrix, and $Q[i, j]$ for $i \neq j$ is interpreted as the rate from i to j . In a transient state the process spends no time (with probability one) and it immediately becomes trapped in some ergodic class. The process in $i \in T$ can be trapped in E_k if and only if the trapping probability $\delta_k[i] > 0$.

The *expected reward rate* at time $t > 0$, notation $R(t)$, is obtained as $R(t) = \sigma P(t) \rho$. It is required in the calculation of the most important performance measure, the *expected accumulated reward* up to time t , given by $\int_0^t R(s) ds$. We have that the expected reward remains unchanged if the reward vector ρ is replaced by $\Pi \rho$. To see this, we use that $P(t) = P(t) \Pi$, so $\sigma P(t) \Pi \rho = \sigma P(t) \rho = R(t)$. Intuitively, the reward in a transient state can be replaced by the sum of the rewards of the ergodic states that it can get trapped in as the process gains no reward while transiting through transient states. The reward of an ergodic state is the sum of the rewards of all states

inside its ergodic class weighted according to their ergodic probabilities. This alternative representation of the reward vector alleviates the presentation of some aggregation methods in later sections. We give an illustration in the following example.

Example 7.1.3 Let $D = (\sigma, \Pi, Q, \rho)$ be defined as:

$$\sigma^\top = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \Pi = \begin{pmatrix} 0 & p & q & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad Q = \begin{pmatrix} 0 & -p\lambda & -q\mu & p\lambda + q\mu \\ 0 & -\lambda & 0 & \lambda \\ 0 & 0 & -\mu & \mu \\ \nu & 0 & 0 & -\nu \end{pmatrix} \quad \rho = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{pmatrix}$$

for $0 < p, q < 1$, where $p + q = 1$ and $\lambda, \mu, \nu > 0$. The ergodic partitioning is $\mathcal{E} = \{E_1, E_2, E_3, T\}$ where $E_1 = \{2\}$, $E_2 = \{3\}$, $E_3 = \{4\}$, and $T = \{1\}$. We have $\mu_i = (1)$ for all $i = 1, 2, 3$, and $\delta_1 = (p)$, $\delta_2 = (1-p)$, and $\delta_3 = (0)$. When the process is in state 1, then with probability p , respectively $1-p$, it is trapped in the ergodic class E_1 , respectively E_2 . Note that $R(t)$ does not depend on r_1 . This is confirmed when ρ is replaced by $\Pi\rho = (pr_2 + (1-p)r_3 \ r_2 \ r_3 \ r_4)^\top$. \square

A Markov reward chain with fast transitions is obtained by adding parameterized, so-called fast, transitions to a standard Markov reward chain. The remaining standard transitions are referred to as slow. The behavior of a Markov reward chain with fast transitions is determined by two generator matrices S and F , which represent the rates of the slow transitions and the rates (called speeds) of the fast transitions, respectively.

Definition 7.1.4 A Markov reward chain with fast transitions $F = (\sigma, S, F, \rho)$ is a function assigning to each $\tau > 0$, the parameterized Markov reward chain

$$M_\tau = (\sigma, S + \tau F, \rho)$$

where $\sigma \in \mathbb{R}^{1 \times n}$ is an initial probability vector, $S, F \in \mathbb{R}^{n \times n}$ are two generator matrices, and $\rho \in \mathbb{R}^{n \times 1}$ is the reward vector. \square

By taking the limit when $\tau \rightarrow \infty$, fast transitions become instantaneous. Then, a Markov reward chain with fast transitions behaves as a discontinuous Markov reward chain [39, 75, 78, 94].

Definition 7.1.5 Let $F = (\sigma, S, F, \rho)$ be a Markov reward chain with fast transitions. The discontinuous Markov reward chain $D = (\sigma, \Pi, Q, \Pi\rho)$ is the limit of F , where the matrix Π is the so-called ergodic projection at zero of F , i.e., $\Pi = \lim_{t \rightarrow \infty} e^{Ft}$, and $Q = \Pi S \Pi$. We write $F \rightarrow_\infty D$. \square

The ergodic projection of a generator matrix also has an alternative characterization given by the following proposition [46, 2].

Proposition 7.1.6 *Let $Q \in \mathbb{R}^{n \times n}$. The matrix $\Pi \in \mathbb{R}^{n \times n}$ is its ergodic projection at zero if and only if*

$$\Pi \geq 0, \Pi \cdot \mathbf{1} = \mathbf{1}, \Pi^2 = \Pi, \Pi Q = Q\Pi = \mathbf{0}$$

and $\text{rank}(\Pi) + \text{rank}(Q) = n$. □

We note that the initial probability vector in Definition 7.1.5 is not affected by the limit construction. We will later motivate the choice of using the reward vector $\Pi\rho$ instead of just ρ . In addition, we define the ergodic partition of a Markov reward chain with fast transitions to be the ergodic partition of its limit discontinuous Markov reward chain.

The ergodic partition can also be obtained in an alternative manner. We write $i \rightarrow j$ if $F[i, j] > 0$ and denote the reflexive-transitive closure of \rightarrow by \twoheadrightarrow . If $i \twoheadrightarrow j$ we say that j is τ -reachable from i . If $i \twoheadrightarrow j$ and $j \twoheadrightarrow i$ we say that i and j τ -communicate. In a slightly different context, it has been shown (see, e.g. [46]) that every ergodic class is actually a closed class of τ -communicating states. Moreover, for all states i and all ergodic states j , $i \twoheadrightarrow j$ iff $\Pi[i, j] > 0$. Now, by $\text{erg}(i) = \{E \in \mathcal{E} \mid i \twoheadrightarrow j, j \in E\}$ we denote the set of ergodic classes which are τ -reachable from the state i . If i is a transient state, i.e., $i \in T$, then $\text{erg}(i)$ is the set of ergodic classes to which it is trapped.

We depict Markov reward chains with fast transitions as in Figure 7.5. The initial probabilities are depicted left above, and the reward rates right above each state. Here, a, b , and c are speeds, whereas λ, μ, ν , and ξ are rates of slow transitions. As in the definition, τ denotes the real parameter.

As an example, the limit of the Markov reward chain with fast transitions in Figure 7.5c is given by the discontinuous Markov reward chain in Example 7.1.3 for $p = \frac{a}{a+b}$ and $q = \frac{b}{a+b}$.

We define a Markov reward chain with silent transitions as a Markov reward chain with fast transitions in which the speeds of the fast transitions are left unspecified. To abstract away from the speeds of the fast transitions we introduce a suitable equivalence relation on Markov reward chains with fast transitions that is induced by the following equivalence relation of matrices.

Definition 7.1.7 Two matrices $A, B \in \mathbb{R}^{n \times n}$ have the same shape (also called a grammar), notation $A \sim B$, if and only if

$$A[i, j] = 0 \text{ if and only if } B[i, j] = 0,$$

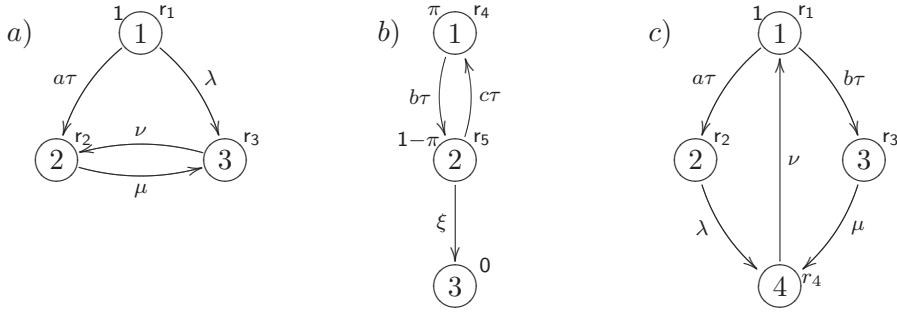


Figure 7.5: Markov reward chains with fast transitions

for all $1 \leq i, j \leq n$. □

It is obvious that \sim is an equivalence on matrices of the same order. The abstraction from speeds is achieved by identifying generator matrices of fast transitions with the same shape. Thus, silent transitions are modeled by equivalence classes of \sim .

Definition 7.1.8 A Markov reward chain with silent transitions \mathbf{S} is a quadruple $\mathbf{S} = (\sigma, S, \mathcal{F}, \rho)$ where \mathcal{F} is an equivalence class of \sim and for every $F \in \mathcal{F}$, $F = (\sigma, S, F, \rho)$ is a Markov reward chain with fast transitions. □

We write $F \in \mathbf{S}$ if $\mathbf{S} = (\sigma, S, \mathcal{F}, \rho)$, and $F = (\sigma, S, F, \rho)$ with $F \in \mathcal{F}$. Furthermore, we lift the relation \sim to Markov reward chains with fast transitions and write $F \sim F'$ if $F, F' \in \mathbf{S}$. The notion of an ergodic partition is speed independent, i.e., if $F \sim F'$, then they have the same ergodic partition. This is because the ergodic partition depends only on the existence of fast transitions, but not on the actual speeds. Hence we can define the ergodic partition of a Markov reward chain with silent transitions \mathbf{S} to be the ergodic partition of any Markov reward chain with fast transitions F with $F \in \mathbf{S}$.

We depict Markov reward chains with silent transitions as in Figure 7.6 by omitting the speeds of the fast transitions. The depicted Markov reward chains with silent transitions are induced by the Markov reward chains with fast transitions in Figure 7.5.

In Figure 7.6, τ can be understood as a label of internal action transitions, as it is common in transition system modeling and process algebra [80, 13]. In this way we formalize the notion of performance analysis for Markov reward chains with nondeterministic internal steps.

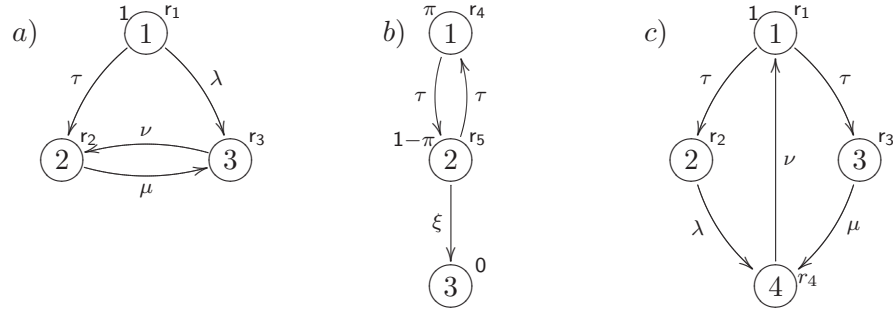


Figure 7.6: Markov reward chains with silent transitions

7.2 Aggregation Methods

In this section we introduce lumping methods for the Markovian models of the previous section following [75, 78, 94]. First, we generalize ordinary lumping of [61] to discontinuous Markov reward chains. Then, we define τ -lumping for Markov reward chains with fast transitions based on ordinary lumping of discontinuous Markov reward chains. Finally, we lift the τ -lumping to τ_{\sim} -lumping of Markov reward chains with silent transitions.

We define aggregation by lumping in terms of matrices. Every partition $\mathcal{P} = \{C_1, \dots, C_N\}$ of $\mathcal{S} = \{1, \dots, n\}$ can be associated with a so-called collector matrix $V \in \mathbb{R}^{n \times N}$ defined by $V[i, k] = 0$ if $i \notin C_k$, $V[i, k] = 1$ if $i \in C_k$, and vice versa. The k -th column of V has 1's for elements corresponding to states in C_k and 0's otherwise. Note that $V \cdot \mathbf{1} = \mathbf{1}$. A distributor matrix $U \in \mathbb{R}^{N \times n}$ for \mathcal{P} is defined as a matrix $U \geq 0$, such that $UV = I^N$. To satisfy these conditions, the elements of the k -th row of U , which correspond to states in the class C_k , sum up to one, whereas the other elements of the row are 0.

An ordinary lumping is a partition of the state space of a discontinuous Markov reward chain into classes such that the states that are lumped together have equivalent behavior for transiting to other classes and, additionally, have the same reward.

Definition 7.2.1 A partition \mathcal{L} of $\{1, \dots, n\}$ is an ordinary lumping, or lumping for short, of a discontinuous Markov reward chain $D = (\sigma, \Pi, Q, \rho)$ if and only if the following hold: (1) $VU\Pi V = \Pi V$, (2) $VUQV = QV$, and (3) $VU\rho = \rho$, where V is the collector matrix and U is any distributor matrix for \mathcal{L} . \square

The lumping conditions only require that the rows of ΠV (respectively QV and ρ) that correspond to the states of the same partition class are equal. We have the following property [75, 78, 94].

Proposition 7.2.2 *Let $D = (\sigma, \Pi, Q, \rho)$ be a discontinuous Markov reward chain and let \mathcal{L} be its ordinary lumping. Define (1) $\bar{\sigma} = \sigma V$, (2) $\bar{\Pi} = U\Pi V$, (3) $\bar{Q} = UQV$, and (4) $\bar{\rho} = U\rho$, for the collector matrix V of \mathcal{L} and any distributor U . Then $\bar{D} = (\bar{\sigma}, \bar{\Pi}, \bar{Q}, \bar{\rho})$ is a discontinuous Markov reward chain. Moreover, $\bar{P}(t) = UP(t)V$ where $\bar{P}(t)$ and $P(t)$ are the transition matrices of \bar{D} and D , respectively. \square*

Using Proposition 7.2.2 we define the lumped process.

Definition 7.2.3 *If the conditions of Proposition 7.2.2 are satisfied, then $D = (\sigma, \Pi, Q, \rho)$ lumps to $\bar{D} = (\bar{\sigma}, \bar{\Pi}, \bar{Q}, \bar{\rho})$, called the lumped discontinuous Markov reward chain with respect to \mathcal{L} . We write $D \xrightarrow{\mathcal{L}} \bar{D}$. \square*

It can readily be seen that neither the definition of a lumping, nor the definition of the lumped process depends on the choice of a distributor matrix U . For example, if $VUQV = QV$, then $VU'QV = VU'VUQV = VUQV = QV$, for any other distributor U' . In the continuous case, when $\Pi = I$ we have $\bar{\Pi} = I$, so \bar{Q} is a generator matrix and our notion of ordinary lumping coincides with the standard definition [61, 83]. The expected reward is preserved by ordinary lumping, since:

$$\bar{R}(t) = \sigma VUP(t)VU\rho = \sigma P(t)VU\rho = \sigma P(t)\rho = R(t).$$

Similarly, as in [61], one can show that other performance measures are also preserved by lumping. We illustrate the situation by an example.

Example 7.2.4 Consider again the discontinuous Markov reward chain D from Example 7.1.3, but assume that $\lambda = \mu$ and $r_2 = r_3$. Then, the partition $\{\{1\}, \{2, 3\}, \{4\}\}$ is an ordinary lumping. Now, D can be lumped to the discontinuous Markov reward chain $\bar{D} = (\bar{\sigma}, \bar{\Pi}, \bar{Q}, \bar{\rho})$ given by

$$\bar{\sigma} = (1 \ 0 \ 0) \quad \bar{\Pi} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \bar{Q} = \begin{pmatrix} 0 & -\lambda & \lambda \\ 0 & -\lambda & \lambda \\ \nu & 0 & -\nu \end{pmatrix} \quad \bar{\rho} = \begin{pmatrix} r_1 \\ r_2 \\ r_4 \end{pmatrix}.$$

Furthermore, the partition $\{\{\{1\}, \{2, 3\}\}, \{4\}\}$ is an ordinary lumping of \bar{D} . So, it can be lumped all the way to the Markov reward chain $M = (\bar{\bar{\sigma}}, \bar{\bar{Q}}, \bar{\bar{\rho}})$ given by

$$\bar{\bar{\sigma}} = (1 \ 0) \quad \bar{\bar{Q}} = \begin{pmatrix} -\lambda & \lambda \\ \nu & -\nu \end{pmatrix} \quad \bar{\bar{\rho}} = \begin{pmatrix} r_2 \\ r_4 \end{pmatrix}.$$

Note that M could also be obtained directly from D by using the ordinary lumping $\{\{1, 2, 3\}, \{4\}\}$. \square

The notion of τ -lumping is based on ordinary lumping for discontinuous Markov reward chains. The aim is that the limit of a τ -lumped Markov reward chain with fast transitions is an ordinary lumped version of the limit of the original Markov reward chain with fast transitions.

Definition 7.2.5 A partition \mathcal{L} of the state space of a Markov reward chain with fast transitions F is called a τ -lumping, if it is an ordinary lumping of its limiting discontinuous Markov reward chain D with $F \rightarrow_{\infty} D$. \square

Note that since we defined the reward of the limit by $\Pi\rho$, a τ -lumping may identify states with different rewards.

Like for ordinary lumping, we define the τ -lumped process by multiplying σ , S , F , and ρ with a collector matrix and a distributor matrix. However, unlike for ordinary lumping, not all distributors are allowed because the lumping condition does not hold for F , but only for D . We give a special class of distributors, called τ -distributors, that give a τ -lumped Markov reward chain with fast transitions which limit is lumped version of the limit of the original Markov reward chain with fast transitions [75, 78, 94].

Before we define the class of τ -distributors, we need a proposition that gives the connection between the τ -lumping, the transient states, and the ergodic classes [76, 94]. Intuitively, if two lumping classes share states from at least one ergodic class, then they both share states from the same ergodic classes. Moreover, if a lumping class contains transient and ergodic states, then it must also contain states from every ergodic class to which these transient states are trapped.

Proposition 7.2.6 Let (σ, S, F, ρ) be a Markov reward chain with fast transitions. Let $\mathcal{E} = \{E_1, \dots, E_m, T\}$ be its ergodic partitioning and let $\mathcal{P} = \{C_1, \dots, C_N\}$ be a τ -lumping. Then, for all $1 \leq I, J \leq M$ and all $1 \leq K, L \leq N$, if $E_I \cap C_K \neq \emptyset$, $E_J \cap C_K \neq \emptyset$, and $E_I \cap C_L \neq \emptyset$, then $E_J \cap C_L \neq \emptyset$. Moreover, if there exists $i \in C_K \cap T$, then $C_K \cap E \neq \emptyset$ for every $E \in \text{erg}(i)$. \square

Next, we give the definition of a τ -distributor.

Definition 7.2.7 Let $F = (\sigma, S, F, \rho)$ be a Markov reward chain with fast transitions. Let $\mathcal{P} = \{C_1, \dots, C_N\}$ be its τ -lumping and $\mathcal{E} = \{E_1, \dots, E_M, T\}$ its ergodic partitioning. Let Π be the ergodic projection of F . Put $e(K) =$

$\{E \in \mathcal{E} \mid C_K \cap E \neq \emptyset\}$. Let $\alpha_{KL} > 0$ if $E_L \in e(K)$ be arbitrary, subject only to $\sum_{L: E_L \in e(K)} \alpha_{KL} = 1$ and $\alpha_{KL} = \alpha_{K'L}$ for K, K' , and L such that $E_L \in e(K)$ and $E_L \in e(K')$. Let $\beta_{Ki} > 0$ for $i \in C_K$ and $e(K) = \emptyset$ be also arbitrary, subject only to $\sum_{i \in C_K} \beta_{Ki} = 1$. Then, a τ -distributor $W \in \mathbb{R}^{N \times n}$ is defined as

$$W[K, i] = \begin{cases} 0, & i \notin C_K \\ \alpha_{KL} \cdot |e(K)| \cdot \frac{\Pi[i, i]}{\sum_{k \in C_K} \Pi[k, k]}, & i \in C_K \cap E_L \\ 0, & i \in C_K \cap T, e(K) \neq \emptyset \\ \beta_{Ki}, & i \in C_K, e(K) = \emptyset. \end{cases} \quad \square$$

Note that if we restrict $\alpha_{KL} = 1/|e(K)|$ and $\beta_{Ki} = 1/|C_K|$, then we obtain as a special case the τ -distributor of [75]. As a distributor, it assigns weights to the rows of SV and FV , and then sums them up. The lumping and the ergodic classes can be grouped such that lumping classes share states only with the ergodic classes of the same group. The set of ergodic classes that have common states with C_K is given by $e(K)$. The weights $\alpha_{KL} > 0$, for $L \in e(K)$, can be arbitrarily distributed among such classes. They must sum up to one to ensure the form of a distributor. The condition $\alpha_{KL} = \alpha_{K'L}$ assures that the states from the same ergodic class are treated in the same way. The weights are multiplied by $|e(K)|$ as the normalization constant $\sum_{k \in C_K} \Pi[k, k]$ is a sum over all states of the $|e(K)|$ shared ergodic classes. As transient states have no ergodic probabilities, they are assigned weight 0 when lumped together with ergodic states. We assign arbitrary weights β_{Ki} when lumping only transient states as their lumped trapping probabilities must be equal [78, 94].

The class of τ -distributors in Definition 7.2.7 has an alternative characterization given by the following proposition.

Proposition 7.2.8 *A matrix W is a τ -distributor for V if and only if (1) it is a distributor for V , (2) $\Pi V W \Pi = \Pi V W$, and (3) the entries of W corresponding to states in classes of transient states are positive. \square*

Having defined τ -distributors, we can now explicitly define a τ -lumped process.

Definition 7.2.9 Let $F = (\sigma, S, F, \rho)$ and let \mathcal{L} be a lumping with a collector matrix V , and a corresponding τ -distributor W . The τ -lumped Markov reward chain with fast transitions $\bar{F} = (\bar{\sigma}, \bar{S}, \bar{F}, \bar{\rho})$ is defined as $\bar{\sigma} = \sigma V$, $\bar{S} = W S V$, $\bar{F} = W F V$, $\bar{\rho} = W \rho$. We say that F τ -lumps to \bar{F}

with respect to W and write $F \overset{\mathcal{L}}{\rightsquigarrow}_W \bar{F}$. We write $F \overset{\mathcal{L}}{\rightsquigarrow} \bar{F}$ if $F \overset{\mathcal{L}}{\rightsquigarrow}_W \bar{F}$ for some τ -distributor W . \square

In general, when lumping a Markov reward chain with fast transitions F using a collector V and a distributor U , USV and UFV are not uniquely determined, i.e., they depend on the choice of the distributor. The restriction to τ -distributors does not change this. Subsequently, the τ -lumped process depends on the choice of the τ -distributor. In order to make the τ -distributor used explicit, we sometimes write $F \overset{\mathcal{L}}{\rightsquigarrow}_{\alpha,\beta} \bar{F}$ in order to emphasize the parameter sets such that $W = W_{\alpha,\beta}$.

The motivation for restricting to τ -distributors, despite that they do not ensure a unique τ -lumped process, is that all τ -lumped processes are equivalent in the limit. This is stated in the following proposition, which gives the precise connection of ordinary lumping and τ -lumping [75, 94].

Proposition 7.2.10 *The following diagram commutes*

$$\begin{array}{ccc} F & \overset{\mathcal{L}}{\rightsquigarrow} & \bar{F} \\ \infty \downarrow & & \downarrow \infty \\ D & \xrightarrow{\mathcal{L}} & \bar{D} \end{array}$$

that is, if $F \overset{\mathcal{L}}{\rightsquigarrow} \bar{F} \rightarrow_{\infty} \bar{D}$ and if $F \rightarrow_{\infty} D \xrightarrow{\mathcal{L}} \bar{D}'$, then $\bar{D} = \bar{D}'$, for F and \bar{F} Markov reward chains with fast transitions, and D , \bar{D} , and \bar{D}' discontinuous Markov reward chains. \square

Moreover, the τ -lumped processes that originate from the same Markov reward chain with fast transitions become exactly the same, once all fast transitions are eliminated [78, 94].

We depict in Figure 7.7 the lumped versions of the Markov reward chains with fast transitions of Figure 7.5. The partitions are indicated by the state labels. We assume that $\lambda = \mu$ and $r_2 = r_3$ for the Markov reward chain with fast transitions in Figure 7.5c. We note that the lumped Markov reward chain with fast transitions show that reward rates of transient states play no role, whereas the ones of the ergodic classes are weighted by the ergodic probabilities. The rate ξ of the Markov reward chain with fast transitions depicted in Figure 7.7 is adjusted by the ergodic probability $\frac{b}{b+c}$ of state 2.

We lift τ -lumping to equivalence classes of \sim to obtain τ_{\sim} -lumping for Markov reward chains with silent transitions. Intuitively, a partition is a τ_{\sim} -lumping of S , if it is a τ -lumping for every $F \in S$ and, moreover, the limit of the τ -lumped process of F does not depend on the parameters chosen for

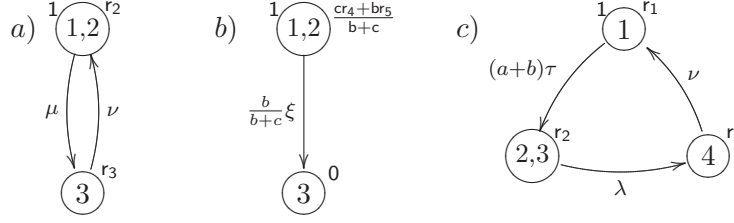


Figure 7.7: τ -lumped Markov reward chains with fast transitions

the τ -distributor. Recall that the parameter set α affects ergodic states, whereas the parameter set β affects only transient states.

Definition 7.2.11 Let S be a Markov reward chain with silent transitions and let \mathcal{L} be its partition. Then \mathcal{L} is a τ_{\sim} -lumping if and only if it is a τ -lumping for every Markov reward chain with fast transitions $F \in S$ and, moreover, for every $F, F' \in S$ if $F \overset{\mathcal{L}}{\underset{\alpha, \beta}{\rightsquigarrow}} \bar{F}$ and $F' \overset{\mathcal{L}}{\underset{\alpha, \beta}{\rightsquigarrow}} \bar{F}'$, then $\bar{F} = \bar{F}'$. \square

The motivation behind the use of the same parameter set β in Definition 7.2.11 is that there may be slow transitions originating from transient states which will depend on β in the lumped process. If we do not restrict to the same parameter set β , then τ_{\sim} -lumpings will only exist in rare cases in which transient states have no slow transitions [78, 94].

Now we can define a τ_{\sim} -lumped process which is unique for a given τ_{\sim} -lumping \mathcal{L} and a parameter set β .

Definition 7.2.12 Let S be a Markov reward chain with silent transitions and \mathcal{L} its τ_{\sim} -lumping. Let $F \in S$ be such that $F \overset{\mathcal{L}}{\underset{\alpha, \beta}{\rightsquigarrow}} \bar{F}$ and let \bar{S} be the Markov reward chain with silent transitions with $\bar{F} \in \bar{S}$. Then S τ_{\sim} -lumps to \bar{S} , with respect to \mathcal{L} and β , notation $S \overset{\mathcal{L}}{\underset{\beta}{\rightsquigarrow}} \bar{S}$. We write $S \overset{\mathcal{L}}{\rightsquigarrow} \bar{S}$ if $S \overset{\mathcal{L}}{\underset{\beta}{\rightsquigarrow}} \bar{S}$ for some parameter set β . \square

As for Markov reward chains with fast transitions, all lumped processes coincide with a unique Markov reward chain, once all silent transitions are eliminated [78, 94]. We depict in Figure 7.8 the τ_{\sim} -lumped versions of the Markov reward chains with silent transitions given in Figure 7.6a and Figure 7.6c, again under the assumption that $\lambda = \mu$ and $r_2 = r_3$. The Markov reward chain with silent transitions depicted in Figure 7.6b has only trivial τ_{\sim} -lumpings because the representative τ -lumped Markov reward chains

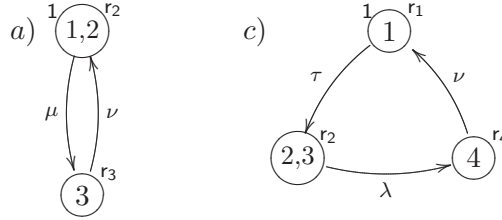


Figure 7.8: τ_{\sim} -lumped Markov reward chains with silent transitions

with fast transitions depend on the speed of the fast transitions as depicted in Figure 7.7b.

Reduction is a specific aggregation method for transforming a discontinuous Markov chain into a standard Markov chain, originally studied in [44, 43, 39]. Extended to reward processes, the method reduces a discontinuous Markov reward chain to a Markov reward chain by eliminating instantaneous states, while retaining the behavior of the regular states. In the same spirit, we define reduction methods that reduce Markov reward chains with fast and silent transitions to Markov reward chains following [78, 94], called τ -reduction and τ_{\sim} -reduction, respectively.

The reduction-based aggregation method masks the stochastic discontinuity of a discontinuous Markov reward chain and transforms it into a Markov reward chain [44, 39, 78, 94]. The underlying idea is to abstract away from the behavior of individual states in an ergodic class. The method is based on the notion of a canonical product decomposition.

Definition 7.2.13 Let $D = (\sigma, \Pi, Q, \rho)$ and assume that $\text{rank}(\Pi) = M$, i.e., that there are M ergodic classes. A canonical product decomposition of Π is a pair of matrices (L, R) with $L \in \mathbb{R}^{M \times n}$ and $R \in \mathbb{R}^{n \times M}$ such that $L \geq 0$, $R \geq 0$, $\text{rank}(L) = \text{rank}(R) = M$, $L \cdot \mathbf{1} = \mathbf{1}$, and $\Pi = RL$. \square

A canonical product decomposition always exists and it can be constructed from the ergodic form of Π as follows:

$$L = \begin{pmatrix} \mu_1 & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mu_2 & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mu_M & \mathbf{0} \end{pmatrix} \quad R = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{1} \\ \delta_1 & \delta_2 & \dots & \delta_M \end{pmatrix}$$

Moreover, it can be shown that any other canonical product decomposition is permutation equivalent to this one. Since a canonical product decomposition (L, R) of Π is a full-rank decomposition, and since Π is idempotent, we also have that $LR = I^M$. Thus, we have $L\Pi = LRL = L$ and $\Pi R = RLR = R$. Next, we present the reduction method.

Definition 7.2.14 Let $D = (\sigma, \Pi, Q, \rho)$ be a discontinuous Markov reward chain. Then, it reduces to the Markov reward chain $M = (\bar{\sigma}, \bar{Q}, \bar{\rho})$, given by $\bar{\sigma} = \sigma R$, $\bar{Q} = LQR$, and $\bar{\rho} = L\rho$, where (L, R) is a canonical product decomposition of Π . We write $D \rightarrow_r M$. \square

If $\bar{P}(t)$ and $P(t)$ are the transition matrices of the reduced and the original chain, respectively, then one can show that $\bar{P}(t) = LP(t)R$ [39, 43].

The reduced process is unique up to a permutation of the states, since the canonical product decomposition is. The states of the reduced process are given by the ergodic classes of the original process, while the transient states are ‘ignored’. Intuitively, the transient states are split probabilistically between the ergodic classes according to their trapping probabilities. In case a transient state is also an initial state, its initial probability is split according to its trapping probabilities. The reward rate is calculated as the sum of the individual reward rates of the states of the ergodic class weighted by their ergodic probabilities. Like lumping, the reduction also preserves the expected reward rate at time t :

$$\bar{R}(t) = \sigma RLP(t)RL\rho = \sigma\Pi P(t)\Pi\rho = \sigma P(t)\rho = R(t).$$

In case the original process has no stochastic discontinuity, i.e., $\Pi = I$, the reduced process is equal to the original. We illustrate the situation by an example.

Example 7.2.15 Recall again the discontinuous Markov reward chain D from Example 7.1.3. The canonical decomposition of Π is given by:

$$\Pi = \begin{pmatrix} 0 & p & q & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad L = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R = \begin{pmatrix} p & q & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Now, D can be reduced to the Markov reward chain $M' = (\bar{\sigma}', \bar{Q}', \bar{\rho}')$ given by

$$\bar{\sigma}' = (p \ q \ 0) \quad \bar{Q}' = \begin{pmatrix} -\lambda & 0 & \lambda \\ 0 & -\mu & \mu \\ p\nu & q\nu & -\nu \end{pmatrix} \quad \bar{\rho}' = \begin{pmatrix} r_2 \\ r_3 \\ r_4 \end{pmatrix}.$$

We note that the Markov reward chain M' cannot be obtained as a τ -lumped version of D . However, M' can be ordinary lumped to the Markov reward chain M of Example 7.2.4 using the partition $\{\{1, 2\}, \{3\}\}$ under the assumption that $\lambda = \mu$ and $r_2 = r_3$. \square

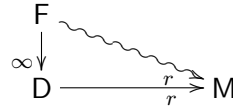
In general, the aggregation methods by ordinary lumping and reduction are not comparable, but the reduction method enjoys the advantage of splitting transient states as in Example 7.2.15. On the other hand, reduction-based aggregation reduces the discontinuous Markov reward chain to a Markov reward chain in one step and it cannot produce the intermediate aggregated Markov reward chains with fast transitions.

We now define a reduction-based aggregation method called τ -reduction. It aggregates a Markov reward chain with fast transitions to an asymptotically equivalent Markov reward chain.

Definition 7.2.16 Let $F = (\sigma, S, F, \rho)$ be a Markov reward chain with fast transitions. Then, it τ -reduces to the Markov reward chain $M = (\bar{\sigma}, \bar{Q}, \bar{\rho})$, given by (1) $\bar{\sigma} = \sigma R$, (2) $\bar{Q} = LSR$, and (3) $\bar{\rho} = L\rho$, where $F \xrightarrow{\infty} (\sigma, \Pi, Q, \Pi\rho)$ and (L, R) is a canonical product decomposition of Π . We write $F \rightsquigarrow_r M$. \square

The following simple property relates τ -reduction to reduction. It holds since $LQR = L\Pi S\Pi R = LSR$ and $L\Pi\rho = L\rho$.

Proposition 7.2.17 *The following diagram commutes*



that is, if $F \rightsquigarrow_r M$ and $F \xrightarrow{\infty} D \xrightarrow[r]{} M'$, then $M = M'$, for F a Markov reward chain with fast transitions, D a discontinuous Markov reward chain and M and M' (continuous) Markov reward chains. \square

We depict in Figure 7.9 the τ -reduced versions of the Markov reward chains with fast transitions given in Figure 7.5. Remarkably, the τ -reduced versions of the Markov reward chains with fast transitions depicted in Figure 7.5a and Figure 7.5b coincide with the τ -lumped ones. However, different from τ -lumping in the τ -reduced process Figure 7.9a the transient state 1 is eliminated. The approach is equivalent when abstracting from whole ergodic classes as illustrated in Figure 7.9b. Figure 7.5c shows the property of reduction to probabilistically split transient states and, consequently, their

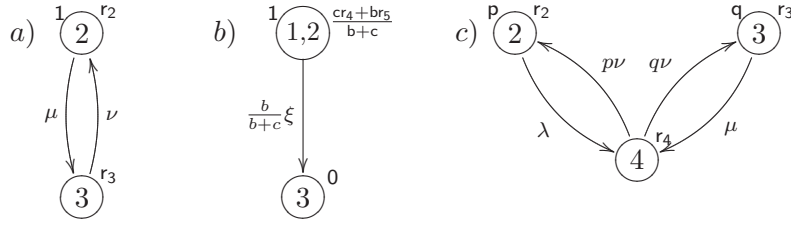


Figure 7.9: τ -reduced Markov reward chains with fast transitions

incoming slow transitions. Note that the initial probability vector is also adjusted according to the trapping probabilities.

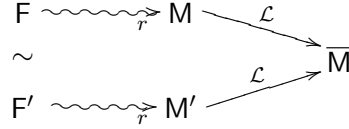
As illustrated by the example, the aggregation methods by τ -lumping and τ -reduction are, in general, incomparable [78, 94]. The reduction-based method retains the feature of splitting transient states according to their trapping probabilities. The combination of τ -reduction followed by ordinary lumping of the resulting Markov reward chain aggregates more than τ -lumping alone [78, 94]. It presents the core of the reduction-based aggregation of Markov reward chains with silent transitions.

By combining τ -reduction with ordinary lumping of Markov reward chains, we can eliminate the effect of the speeds and obtain a reduction-like aggregation method for Markov reward chains with silent transitions. Here, we refer to this method as τ_{\sim} -reduction.

Naturally, one could define a reduction-based method for Markov reward chains with silent transitions by direct lifting of τ -reduction, i.e., by saying that a Markov reward chain with silent transitions S reduces to a Markov reward chain M if all Markov reward chains with fast transitions $F \in S$ τ -reduce to M . However, such is not an efficient reduction method as it is applicable only in a few special cases when all Markov reward chains with fast transitions in a \sim -equivalence class τ -reduce to the same Markov reward chain [78, 94]. For this reason we combine τ -reduction and lumping.

Similarly as for τ_{\sim} -lumping, the result of the τ_{\sim} -reduction should not depend on the representative Markov reward chain with fast transitions. Therefore, a Markov reward chain with silent transitions can be τ_{\sim} -reduced if all Markov reward chains with fast transitions in its equivalence class τ -reduce to Markov reward chains that can be ordinary lumped to the same

Markov reward chain, as depicted below:



This is captured by the following definition.

Definition 7.2.18 Let S be a Markov reward chain with silent transitions, let $\mathcal{E} = \{E_1, \dots, E_M, T\}$ be its ergodic partition, and \mathcal{L} a partition of $\{E_1, \dots, E_M\}$. Then S can be τ_{\sim} -reduced according to \mathcal{L} if and only if there exists a Markov reward chain \bar{M} , such that for every $F \in S$, we have that $F \rightsquigarrow_r M \xrightarrow{\mathcal{L}} \bar{M}$ for some Markov reward chain M . We write $S \rightsquigarrow_r^{\mathcal{L}} \bar{M}$. We may also write $S \rightsquigarrow_r \bar{M}$ if a partition \mathcal{L} exists such that $S \rightsquigarrow_r^{\mathcal{L}} \bar{M}$. \square

We note that both τ_{\sim} -lumping and τ_{\sim} -reduction produce the same process when all silent transitions are eliminated, cf. [78, 94] for details. Consequently, the Markov reward chains with silent transitions depicted in Figure 7.6 τ_{\sim} -reduce in the same way as they can be τ_{\sim} -lumped. Again, the Markov reward chain with silent transitions depicted in Figure 7.6b cannot be non-trivially τ_{\sim} -reduced as its τ -reduced version, given in Figure 7.9b, depends on the speeds of the fast transitions and has only trivial lumpings.

7.3 Relational Properties

Next, we investigate the relational properties of the lumping-based aggregation methods. For ordinary lumping, the combination of transitivity and strict confluence ensures that iterative application yields a uniquely determined process. In the case of τ -lumping, by Proposition 7.2.10, only the limit of the final reduced process is uniquely determined, unless the final process contains no fast transitions. Similarly, for τ_{\sim} -lumping the reduced process is uniquely determined only if it does not contain any silent transitions. There is no need to investigate the relational properties of the reduction-based methods, since they act in one step (no iteration is possible), in a unique way, between different types of models.

First, we investigate the properties of the relation \succcurlyeq on discontinuous Markov reward chains defined by

$$D_1 \succcurlyeq D_2 \text{ if and only if there exists } \mathcal{L} \text{ such that } D_1 \xrightarrow{\mathcal{L}} D_2.$$

The above relation is clearly reflexive, since the trivial partition Δ is always an ordinary lumping, i.e., $D \xrightarrow{\Delta} D$ for any discontinuous Markov reward chain D . Transitivity enables replacement of repeated application of ordinary lumping by a single application using an ordinary lumping that is a composition of the individual lumpings.

Theorem 7.3.1 *Let D be a discontinuous Markov reward chain such that $D \xrightarrow{\mathcal{L}} \bar{D}$ and $\bar{D} \xrightarrow{\bar{\mathcal{L}}} \bar{\bar{D}}$. Then $D \xrightarrow{\mathcal{L} \circ \bar{\mathcal{L}}} \bar{\bar{D}}$. \square*

PROOF Let $D = (\sigma, \Pi, Q, \rho)$, $\bar{D} = (\bar{\sigma}, \bar{\Pi}, \bar{Q}, \bar{\rho})$, and $\bar{\bar{D}} = (\bar{\bar{\sigma}}, \bar{\bar{\Pi}}, \bar{\bar{Q}}, \bar{\bar{\rho}})$. Let V and \bar{V} denote the collector matrices for \mathcal{L} and $\bar{\mathcal{L}}$, respectively. The collector matrix for $\mathcal{L} \circ \bar{\mathcal{L}}$ is $V\bar{V}$. The following lumping conditions hold: $VU\Pi V = \bar{\Pi}V$, $VUQV = \bar{Q}V$, and $VU\rho = \bar{\rho}$. Also $\bar{\Pi} = U\Pi V$, $\bar{Q} = UQV$, and $\bar{\rho} = U\rho$ for any distributor U for V . Similarly, it holds that: $\bar{V}\bar{U}\bar{\Pi}\bar{V} = \bar{\bar{\Pi}}\bar{V}$, $\bar{V}\bar{U}\bar{Q}\bar{V} = \bar{\bar{Q}}\bar{V}$, and $\bar{V}\bar{U}\bar{\rho} = \bar{\bar{\rho}}$. Moreover $\bar{\Pi} = \bar{U}\bar{\Pi}\bar{V}$, $\bar{Q} = \bar{U}\bar{Q}\bar{V}$, and $\bar{\rho} = \bar{U}\bar{\rho}$ for any distributor \bar{U} for \bar{V} .

The iterative application of the ordinary lumping method can be replaced by the ordinary lumping given by the partition $\mathcal{L} \circ \bar{\mathcal{L}}$, that corresponds to the collector matrix $\bar{\bar{V}} = V\bar{V}$. A corresponding distributor is $\bar{\bar{U}} = \bar{U}U$, because $\bar{\bar{U}}\bar{\bar{V}} = \bar{U}U\bar{V} = I$. That the partition is indeed an ordinary lumping follows from the observation

$$\bar{\bar{V}}\bar{\bar{U}}\bar{\Pi}\bar{\bar{V}} = V\bar{V}\bar{U}U\Pi V\bar{V} = V\bar{V}\bar{U}\bar{\Pi}\bar{V} = V\bar{\Pi}\bar{V} = VU\Pi V\bar{V} = \bar{\Pi}V\bar{V} = \bar{\bar{\Pi}}\bar{\bar{V}}.$$

Similarly, one gets the condition for Q , and

$$\bar{\bar{V}}\bar{\bar{U}}\bar{\rho} = V\bar{V}\bar{U}U\rho = V\bar{V}\bar{U}\bar{\rho} = V\bar{\rho} = VU\rho = \bar{\rho}. \quad \blacksquare$$

The relation \geq on Markov reward chains with fast transitions, defined by

$$F_1 \geq F_2 \text{ if and only if there exists } \mathcal{L} \text{ such that } F_1 \xrightarrow{\mathcal{L}} F_2$$

is a preorder as well. It is reflexive via the trivial lumping Δ . The following theorem shows the transitivity of the τ -lumping relation.

Theorem 7.3.2 *Let F be a Markov reward chain with fast transitions, such that $F \xrightarrow{\mathcal{L}} \bar{F}$ and $\bar{F} \xrightarrow{\bar{\mathcal{L}}} \bar{\bar{F}}$. Then $F \xrightarrow{\mathcal{L} \circ \bar{\mathcal{L}}} \bar{\bar{F}}$. \square*

PROOF Let $F = (\sigma, F, S, \rho)$ and $\bar{F} = (\bar{\sigma}, \bar{F}, \bar{S}, \bar{\rho})$. Denote by V and \bar{V} the collector matrices for \mathcal{L} and $\bar{\mathcal{L}}$, respectively. The collector matrix for $\mathcal{L} \circ \bar{\mathcal{L}}$ is then $\bar{\bar{V}} = V\bar{V}$. Let W and \bar{W} be the corresponding τ -distributors used

for $F \xrightarrow{\mathcal{L}} \bar{F}$ and $\bar{F} \xrightarrow{\bar{\mathcal{L}}} \bar{\bar{F}}$, respectively. Since τ -lumping is defined in terms of ordinary lumping, it is sufficient to show that $\bar{\bar{W}} = \bar{W}W$ is a τ -distributor. From Theorem 7.3.1 it follows that it is a distributor. The condition requiring positive entries corresponding to transient states that lump only with other transient states, can be checked using the explicit description of τ -distributors as in [94]. It remains to verify the third condition.

Let Π and $\bar{\Pi}$ be the ergodic projections of F and \bar{F} . Then, $\Pi V W \Pi = \Pi V W$ and $\bar{\Pi} \bar{V} \bar{W} \bar{\Pi} = \bar{\Pi} \bar{V} \bar{W}$. We have that

$$\begin{aligned} \Pi \bar{\bar{V}} \bar{\bar{W}} \Pi &= \Pi V \bar{V} \bar{W} W \Pi = V W \Pi V \bar{V} \bar{W} W \Pi = V \bar{\Pi} \bar{V} \bar{W} W \Pi \\ &= V \bar{\Pi} \bar{V} \bar{W} \bar{\Pi} W \Pi = V \bar{\Pi} \bar{V} \bar{W} W \Pi V W \Pi = V \bar{\Pi} \bar{V} \bar{W} W \Pi V W \\ &= \dots \\ &= \Pi V \bar{V} \bar{W} W = \Pi \bar{\bar{V}} \bar{\bar{W}}. \end{aligned}$$

■

Similarly, τ_{\sim} -lumping induces a preorder on Markov reward chains with silent transitions defined by

$$S_1 \geq S_2 \text{ if and only if there exists } \mathcal{L} \text{ such that } S_1 \xrightarrow{\mathcal{L}} S_2.$$

Reflexivity again holds due to the trivial partition Δ , while transitivity is a direct consequence of Theorem 7.3.2 and the definition of τ_{\sim} -lumping, Definition 7.2.11. Thus, we have the following theorem.

Theorem 7.3.3 *Let S be a Markov reward chain with silent transitions. Suppose $S \xrightarrow{\mathcal{L}} \bar{S}$ and $\bar{S} \xrightarrow{\bar{\mathcal{L}}} \bar{\bar{S}}$. Then $S \xrightarrow{\mathcal{L} \circ \bar{\mathcal{L}}} \bar{\bar{S}}$. □*

The lumping preorders also have the strict confluence property. In case of lumping this means that if $P \xrightarrow{\mathcal{L}_1} P_1$ and $P \xrightarrow{\mathcal{L}_2} P_2$, then there exist two partitions $\bar{\mathcal{L}}_1$ and $\bar{\mathcal{L}}_2$ such that $P_1 \xrightarrow{\mathcal{L}_1 \circ \bar{\mathcal{L}}_1} \bar{P}$ and $P_2 \xrightarrow{\mathcal{L}_2 \circ \bar{\mathcal{L}}_2} \bar{P}$. One can prove the strict confluence property by adapting the proof for Markov reward chains, e.g., from [89].

7.4 Parallel Composition and Compositionality

In this section, we define parallel composition for each of the models, and prove the compositionality results. The definitions are based on Kronecker products and sums, as for standard Markov reward chains [31, 33].

The intuition behind the Kronecker sum is that it represents interleaving, whereas the Kronecker product represents synchronization. Let us first recall the definition of Kronecker product and sum.

Definition 7.4.1 Let $A \in \mathbb{R}^{n_1 \times n_2}$ and $B \in \mathbb{R}^{m_1 \times m_2}$. The *Kronecker product* of A and B is a matrix $(A \otimes B) \in \mathbb{R}^{n_1 m_1 \times n_2 m_2}$ defined by

$$(A \otimes B)[(i-1)m_1 + k, (j-1)m_2 + \ell] = A[i, j]B[k, \ell]$$

for $1 \leq i \leq n_1$, $1 \leq j \leq n_2$, $1 \leq k \leq m_1$, and $1 \leq \ell \leq m_2$.

The Kronecker sum of two square matrices $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{m \times m}$ is a matrix $(A \oplus B) \in \mathbb{R}^{nm \times nm}$ defined by $A \oplus B = A \otimes I^m + I^n \otimes B$. \square

Next, we list some basic properties of the Kronecker product and sum [49].

Proposition 7.4.2 *The following equations hold:*

1. $(A \otimes B)(C \otimes D) = AC \otimes BD$,
2. $(A + B) \otimes (C + D) = A \otimes C + A \otimes D + B \otimes C + B \otimes D$,
3. $c(A \otimes B) = (cA \otimes B) = (A \otimes cB)$,
4. $c(A \oplus B) = (cA \oplus cB)$,
5. $e^{A \oplus B} = e^A \otimes e^B$,
6. $\text{rank}(A \otimes B) = \text{rank}(A) \text{rank}(B)$. \square

We also need the notion of a Kronecker product of two partitions. Let \mathcal{L}_1 and \mathcal{L}_2 be two partitions with corresponding collector matrices V_1 and V_2 , respectively. Then $\mathcal{L}_1 \otimes \mathcal{L}_2$ denotes the partition corresponding to the collector matrix $V_1 \otimes V_2$.

First, we present the definition of parallel composition of discontinuous Markov reward chains. The intuition is that ‘rates’ interleave, and the probabilities of the instantaneous transitions synchronize, i.e., they are independent.

Definition 7.4.3 Let $D_1 = (\sigma_1, \Pi_1, Q_1, \rho_1)$ and $D_2 = (\sigma_2, \Pi_2, Q_2, \rho_2)$ be discontinuous Markov reward chains. Then, their parallel composition is defined as:

$$D_1 \parallel D_2 = (\sigma_1 \otimes \sigma_2, \Pi_1 \otimes \Pi_2, Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2, \rho_1 \otimes \mathbf{1}^{|\rho_2|} + \mathbf{1}^{|\rho_1|} \otimes \rho_2). \quad \square$$

The following theorem shows that the parallel composition of two discontinuous Markov reward chains is well defined.

Theorem 7.4.4 *Let D_1 and D_2 be two discontinuous Markov reward chains. Then $D_1 \parallel D_2$ is a discontinuous Markov reward chain.* \square

PROOF Let $D_1 = (\sigma_1, \Pi_1, Q_1, \rho_1)$ and $D_2 = (\sigma_2, \Pi_2, Q_2, \rho_2)$. The initial probability vector $\sigma_1 \otimes \sigma_2$ is a stochastic vector and the reward vector is well defined. Using Proposition 7.4.2(1)-(3), it is easy to check that the matrices $\Pi_1 \otimes \Pi_2$ and $Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2$ satisfy the conditions of Definition 7.1.2:

1. $(\Pi_1 \otimes \Pi_2) \geq 0$, $(\Pi_1 \otimes \Pi_2) \cdot \mathbf{1} = \mathbf{1}$, and $(\Pi_1 \otimes \Pi_2)^2 = \Pi_1 \otimes \Pi_2$;
2. $(\Pi_1 \otimes \Pi_2) \cdot (Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2) = (Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2) \cdot (\Pi_1 \otimes \Pi_2) = Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2$;
3. $(Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2) \cdot \mathbf{1} = \mathbf{0}$; and
4. $Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2 + (c_1 + c_2) \cdot (\Pi_1 \otimes \Pi_2) = (Q_1 + c_1 \Pi_1) \otimes \Pi_2 + \Pi_1 \otimes (Q_2 + c_2 \Pi_2) \geq 0$ for $c_1, c_2 \geq 0$ with $Q_1 + c_1 \Pi_1, Q_2 + c_2 \Pi_2 \geq 0$. ■

In the special case, when both discontinuous Markov reward chains are continuous, their parallel composition is again a Markov reward chain as defined in [31]. Moreover, the following property shows that the parallel composition of two discontinuous Markov reward chains has a transition matrix that is the Kronecker product of the individual transition matrices, corresponding to the intuition that the Kronecker product represents synchronization. This justifies the definition of the parallel composition.

Theorem 7.4.5 *Let D_1 and D_2 be two discontinuous Markov reward chains with transition matrices $P_1(t)$ and $P_2(t)$, respectively. Then the transition matrix of $D_1 \parallel D_2$ is given by $P_1(t) \otimes P_2(t)$. □*

PROOF Let $D_1 = (\sigma_1, \Pi_1, Q_1, \rho_1)$ and $D_2 = (\sigma_2, \Pi_2, Q_2, \rho_2)$. As the matrices $Q_1 \otimes \Pi_2$ and $\Pi_1 \otimes Q_2$ commute, and $P_i(t)\Pi_i = \Pi_i P_i(t) = P_i(t)$, we derive:

$$\begin{aligned}
& (\Pi_1 \otimes \Pi_2) e^{(Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2)t} \\
&= (\Pi_1 \otimes \Pi_2) (e^{(Q_1 \otimes \Pi_2)t} e^{(\Pi_1 \otimes Q_2)t}) \\
&= (\Pi_1 \otimes \Pi_2) \left(\sum_{n=0}^{\infty} \frac{(Q_1 \otimes \Pi_2)^n t^n}{n!} \right) \left(\sum_{n=0}^{\infty} \frac{(\Pi_1 \otimes Q_2)^n t^n}{n!} \right) \\
&= (\Pi_1 \otimes \Pi_2) \left(I \otimes I + \sum_{n=1}^{\infty} \frac{(Q_1 \otimes \Pi_2)^n t^n}{n!} \right) \left(I \otimes I + \sum_{n=1}^{\infty} \frac{(\Pi_1 \otimes Q_2)^n t^n}{n!} \right) \\
&= (\Pi_1 \otimes \Pi_2) \left(I \otimes I + \sum_{n=1}^{\infty} \frac{(Q_1^n \otimes \Pi_2^n) t^n}{n!} \right) \left(I \otimes I + \sum_{n=1}^{\infty} \frac{(\Pi_1^n \otimes Q_2^n) t^n}{n!} \right) \\
&= (\Pi_1 \otimes \Pi_2) \left(I \otimes I + \sum_{n=1}^{\infty} \frac{(Q_1^n \otimes \Pi_2) t^n}{n!} \right) \left(I \otimes I + \sum_{n=1}^{\infty} \frac{(\Pi_1 \otimes Q_2^n) t^n}{n!} \right) \\
&= (\Pi_1 \otimes \Pi_2) \left(I \otimes I + \left(\sum_{n=1}^{\infty} \frac{Q_1^n t^n}{n!} \right) \otimes \Pi_2 \right) \left(I \otimes I + \Pi_1 \otimes \left(\sum_{n=1}^{\infty} \frac{Q_2^n t^n}{n!} \right) \right)
\end{aligned}$$

$$\begin{aligned}
&= (\Pi_1 \otimes \Pi_2)(I \otimes I + (e^{Q_1 t} - I) \otimes \Pi_2)(I \otimes I + \Pi_1 \otimes (e^{Q_2 t} - I)) \\
&= (\Pi_1 \otimes \Pi_2)(I \otimes I + e^{Q_1 t} \otimes \Pi_2 - I \otimes \Pi_2)(I \otimes I + \Pi_1 \otimes e^{Q_2 t} - \Pi_1 \otimes I) \\
&= (\Pi_1 \otimes \Pi_2 + P_1(t) \otimes \Pi_2 - \Pi_1 \otimes \Pi_2)(I \otimes I + \Pi_1 \otimes e^{Q_2 t} - \Pi_1 \otimes I) \\
&= (P_1(t) \otimes \Pi_2)(I \otimes I + \Pi_1 \otimes e^{Q_2 t} - \Pi_1 \otimes I) \\
&= (P_1(t) \otimes \Pi_2 + P_1(t) \otimes P_2(t) - P_1(t) \otimes \Pi_2) \\
&= P_1(t) \otimes P_2(t),
\end{aligned}$$

which completes the proof. \blacksquare

Remark 7.4.6 We motivate Definition 7.4.3 also from another perspective. By the standard probabilistic, i.e., non-matrix representation of discontinuous Markov reward chain the same notion can be obtained by the following analysis. Let $\{X(t) \mid t \geq 0\}$ and $\{Y(t) \mid t \geq 0\}$ be two discontinuous Markov reward chains defined on state spaces S_X and S_Y respectively. Their parallel composition can be defined as the stochastic process $\{(X \parallel Y)(t) \mid t \geq 0\}$ with the state space $S_X \times S_Y$, such that $(X \parallel Y)(t) = (x, y)$ if and only if $X(t) = x$ and $Y(t) = y$. One can show that this process is again a discontinuous Markov reward chain with transition matrix equal to the Kronecker product of the transition matrices of $\{X(t) \mid t \geq 0\}$ and $\{Y(t) \mid t \geq 0\}$. It is known that the matrices Π and Q characterizing a transition matrix $P(t)$ are obtained as

$$\Pi = \lim_{t \rightarrow 0} P(t) \quad \text{and} \quad Q = \lim_{h \rightarrow 0} \frac{P(h) - \Pi}{h} \quad [39].$$

Applying this result on the transition matrix of $\{(X \parallel Y)(t) \mid t \geq 0\}$ and using the definition of $(X \parallel Y)(0)$ we obtain the first three components of the quadruple from Definition 7.4.3. The reward vector for the parallel composition encodes the assumption that the reward rate in (x, y) is the sum of the reward rates in x and y . \square

It is easy to see that the expected reward of the parallel composition is the sum of the expected rewards of the components. Using Proposition 7.4.2(1) and (2) we have

$$\begin{aligned}
&(\sigma_1 \otimes \sigma_2)(P_1(t) \otimes P_2(t))(\rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \rho_2) \\
&= \sigma_1 P_1(t) \rho_1 \otimes \sigma_1 P_1(t) \mathbf{1} + \sigma_2 P_2(t) \mathbf{1} \otimes \sigma_2 P_2(t) \rho_2 \\
&= R_1(t) \otimes \mathbf{1} + \mathbf{1} \otimes R_2(t) \\
&= R_1(t) + R_2(t).
\end{aligned}$$

The following theorem shows that both lumping and reduction are compositional with respect to the parallel composition of discontinuous Markov reward chains.

Theorem 7.4.7 *If $D_1 \xrightarrow{\mathcal{L}_1} \bar{D}_1$ and $D_2 \xrightarrow{\mathcal{L}_2} \bar{D}_2$, then $D_1 \parallel D_2 \xrightarrow{\mathcal{L}_1 \otimes \mathcal{L}_2} \bar{D}_1 \parallel \bar{D}_2$. Also, if $D_1 \rightarrow_r M_1$ and $D_2 \rightarrow_r M_2$, then $D_1 \parallel D_2 \rightarrow_r M_1 \parallel M_2$. \square*

PROOF Let $D_1 = (\sigma_1, \Pi_1, Q_1, \rho_1)$, $\bar{D}_1 = (\bar{\sigma}_1, \bar{\Pi}_1, \bar{Q}_1, \bar{\rho}_1)$, $D_2 = (\sigma_2, \Pi_2, Q_2, \rho_2)$, and $\bar{D}_2 = (\bar{\sigma}_2, \bar{\Pi}_2, \bar{Q}_2, \bar{\rho}_2)$. We first prove the compositionality of lumping. We show that $\mathcal{L}_1 \otimes \mathcal{L}_2$ is an ordinary lumping of

$$D_1 \parallel D_2 = (\sigma_1 \otimes \sigma_2, \Pi_1 \otimes \Pi_2, Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2, \rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \rho_2).$$

Let U_1 , U_2 , and $U_1 \otimes U_2$ be distributors and V_1 , V_2 , and $V_1 \otimes V_2$ be the collectors for \mathcal{L}_1 , \mathcal{L}_2 , and $\mathcal{L}_1 \otimes \mathcal{L}_2$, respectively. By using the lumping conditions and Proposition 7.4.2(1) and (2) we have that

$$\begin{aligned} & (V_1 \otimes V_2)(U_1 \otimes U_2)(\Pi_1 \otimes \Pi_2)(V_1 \otimes V_2) \\ &= V_1 U_1 \Pi_1 V_1 \otimes V_2 U_2 \Pi_2 V_2 \\ &= \Pi_1 V_1 \otimes \Pi_2 V_2 \\ &= (\Pi_1 \otimes \Pi_2)(V_1 \otimes V_2) \\ & \\ & (V_1 \otimes V_2)(U_1 \otimes U_2)(Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2)(V_1 \otimes V_2) \\ &= V_1 U_1 Q_1 V_1 \otimes V_2 U_2 \Pi_2 V_2 + V_1 U_1 \Pi_1 V_1 \otimes V_2 U_2 Q_2 V_2 \\ &= Q_1 V_1 \otimes \Pi_2 V_2 + \Pi_1 V_1 \otimes Q_2 V_2 \\ &= (Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2)(V_1 \otimes V_2) \\ & \\ & (V_1 \otimes V_2)(U_1 \otimes U_2)(\rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \rho_2) \\ &= V_1 U_1 \rho_1 \otimes V_2 U_2 \mathbf{1} + V_1 U_1 \mathbf{1} \otimes V_2 U_2 \rho_2 \\ &= \rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \rho_2. \end{aligned}$$

Next, we prove that the lumped parallel composition is the parallel composition of the lumped components. We easily get, by Proposition 7.4.2(1) and (2),

$$\begin{aligned} (U_1 \otimes U_2)(\Pi_1 \otimes \Pi_2)(V_1 \otimes V_2) &= \bar{\Pi}_1 \otimes \bar{\Pi}_2 \text{ and} \\ (U_1 \otimes U_2)(Q_1 \otimes \Pi_2 + \Pi_1 \otimes Q_2)(V_1 \otimes V_2) &= \bar{Q}_1 \otimes \bar{\Pi}_2 + \bar{\Pi}_1 \otimes \bar{Q}_2. \end{aligned}$$

Next, we consider reduction. Let $\Pi_1 = R_1 L_1$ and $\Pi_2 = R_2 L_2$ be some canonical product decompositions. Put $L = L_1 \otimes L_2$ and $R = R_1 \otimes R_2$. Note that $L \geq 0$ and $R \geq 0$ because $L_1, L_2, R_1, R_2 \geq 0$. We also have $L \cdot \mathbf{1} = (L_1 \otimes L_2) \cdot (\mathbf{1} \otimes \mathbf{1}) = L_1 \cdot \mathbf{1} \otimes L_2 \cdot \mathbf{1} = \mathbf{1} \otimes \mathbf{1} = \mathbf{1}$. Since $\text{rank}(A \otimes B) = \text{rank}(A) \cdot \text{rank}(B)$ by Proposition 7.4.2(6), we get that (L, R) is a canonical product decomposition of $\Pi = \Pi_1 \otimes \Pi_2$. Reducing $D_1 \parallel D_2$ using the canonical product decomposition (L, R) gives us $M_1 \parallel M_2$. \blacksquare

We now present the definition of the parallel composition of Markov reward chains with fast transitions. It comprises Kronecker sums of the generator matrices, i.e., interleaving of the rates for both slow and fast transitions.

Definition 7.4.8 Let $F_1 = (\sigma_1, S_1, F_1, \rho_1)$ and $F_2 = (\sigma_2, S_2, F_2, \rho_2)$ be two Markov reward chains with fast transitions. Then their parallel composition is defined as

$$F_1 \parallel F_2 = (\sigma_1 \otimes \sigma_2, S_1 \oplus S_2, F_1 \oplus F_2, \rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \rho_2). \quad \square$$

It is not difficult to see that the parallel composition of Markov reward chains with fast transitions is well defined. In Figure 7.10a and Figure 7.10b we recall the two Markov reward chains with fast transitions of Figure 7.5a and Figure 7.5b, respectively. Their parallel composition is depicted in Figure 7.10c.

Having defined parallel composition for both models, we show how they are related: the limit of the parallel composition of two Markov reward chains with fast transitions is the parallel composition of the limits of the components (that are discontinuous Markov reward chains). Hence, a continuity property of the parallel composition holds as stated in the next result.

Theorem 7.4.9 *If $F_1 \rightarrow_{\infty} D_1$ and $F_2 \rightarrow_{\infty} D_2$, then $F_1 \parallel F_2 \rightarrow_{\infty} D_1 \parallel D_2$.* \square

PROOF Let $F_1 = (\sigma_1, S_1, F_1, \rho_1)$ and $F_2 = (\sigma_2, S_2, F_2, \rho_2)$, and let their corresponding limits be $D_1 = (\sigma_1, \Pi_1, Q_1, \Pi_1 \rho_1)$ and $D_2 = (\sigma_2, \Pi_2, Q_2, \Pi_2 \rho_2)$. Using Proposition 7.4.2(4) and (5), we get that $\Pi_1 \otimes \Pi_2$ is the ergodic projection of $F_1 \oplus F_2$, i.e., $\lim_{t \rightarrow \infty} e^{(F_1 \oplus F_2)t} = \Pi_1 \otimes \Pi_2$. As before, using the distributivity of the Kronecker product and the fact that Π_1 is a stochastic matrix, we derive $Q_1 \otimes \Pi_2 + \Pi_2 \otimes Q_1 = (\Pi_1 \otimes \Pi_2)(S_1 \oplus S_2)(\Pi_1 \otimes \Pi_2)$ and $(\Pi_1 \otimes \Pi_2)(\rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \rho_2) = \Pi_1 \rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \Pi_2 \rho_2$. \blacksquare

Next we show that τ -lumping and τ -reduction are compositional as well, with respect to the parallel composition of Markov reward chains with fast transitions.

Theorem 7.4.10 *If $F_1 \xrightarrow{\mathcal{L}_1} \bar{F}_1$ and $F_2 \xrightarrow{\mathcal{L}_2} \bar{F}_2$, then $F_1 \parallel F_2 \xrightarrow{\mathcal{L}_1 \otimes \mathcal{L}_2} \bar{F}_1 \parallel \bar{F}_2$. Also, if $F_1 \rightsquigarrow_r M_1$ and $F_2 \rightsquigarrow_r M_2$, then $F_1 \parallel F_2 \rightsquigarrow_r M_1 \parallel M_2$.* \square

PROOF Let $F_1 = (\sigma_1, S_1, F_1, \rho_1)$, $F_2 = (\sigma_2, S_2, F_2, \rho_2)$, $\bar{F}_1 = (\bar{\sigma}_1, \bar{S}_1, \bar{F}_1, \bar{\rho}_1)$, and $\bar{F}_2 = (\bar{\sigma}_2, \bar{S}_2, \bar{F}_2, \bar{\rho}_2)$. By Theorem 7.4.7 and the continuity result Theorem 7.4.9, we get that $\mathcal{L}_1 \otimes \mathcal{L}_2$ is a τ -lumping for $F_1 \parallel F_2$. Let W_1 and W_2

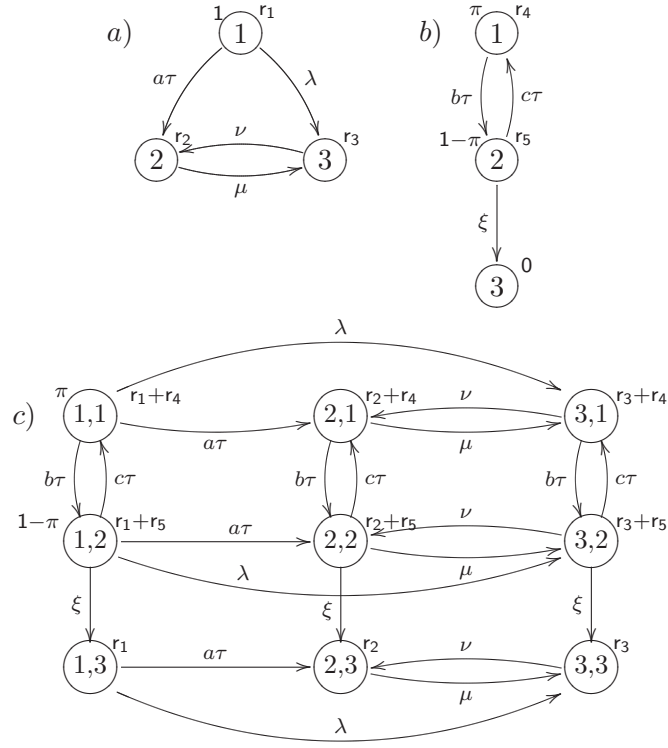


Figure 7.10: Parallel composition of Markov reward chains with fast transitions

be the τ -distributors used for the τ -lumped processes in the assumption, respectively. By Definition 7.2.8, Theorem 7.4.9, and Definition 7.4.3 for the parallel composition of discontinuous Markov reward chains, we have that $W_1 \otimes W_2$ is a τ -distributor for $F_1 \parallel F_2$. The τ -lumped process corresponding to $W_1 \otimes W_2$ is exactly $\bar{F}_1 \parallel \bar{F}_2$.

We next show the compositionality of τ -reduction. Let $\Pi_1 = R_1 L_1$ and $\Pi_2 = R_2 L_2$ be the canonical product decompositions of $\Pi_1 = \lim_{t \rightarrow \infty} e^{F_1 t}$ and $\Pi_2 = \lim_{t \rightarrow \infty} e^{F_2 t}$, respectively. Put $L = L_1 \otimes L_2$ and $R = R_1 \otimes R_2$. Then (L, R) is a canonical product decomposition of $\Pi = \Pi_1 \otimes \Pi_2$, as in the proof of Theorem 7.4.7. This canonical product decomposition applied to $F_1 \parallel F_2$ produces $M_1 \parallel M_2$ as the τ -reduced process. ■

In Figure 7.11a and Figure 7.11b we repeat the aggregated versions of the Markov reward chains with fast transitions from Figure 7.10a and Fig-

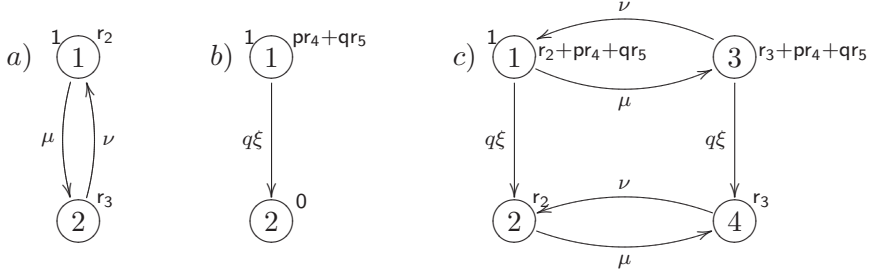


Figure 7.11: Aggregated Markov reward chains with fast transitions

ure 7.10b. The Markov reward chain with fast transitions in 7.11c is the parallel composition of the Markov reward chains with fast transitions in 7.11a and 7.11b with $p = \frac{c}{b+c}$ and $q = \frac{b}{b+c}$. By Theorem 7.4.10, we have that the Markov reward chain in 7.11c, is in fact the lumped version of the parallel composition given in Figure 7.10c.

We define the parallel composition of two Markov reward chains with silent transitions via the equivalence class of the parallel composition of the representative Markov reward chains with fast transitions.

Definition 7.4.11 Let $S_1 = (\sigma_1, S_1, \mathcal{F}_1, \rho_1)$ and $S_2 = (\sigma_2, S_2, \mathcal{F}_2, \rho_2)$ be two Markov reward chains with silent transitions. Then their parallel composition is defined as

$$S_1 \parallel S_2 = (\sigma_1 \otimes \sigma_2, S_1 \oplus S_2, \mathcal{F}_1 \oplus \mathcal{F}_2, \rho_1 \otimes \mathbf{1} + \mathbf{1} \otimes \rho_2),$$

where $\mathcal{F}_1 \oplus \mathcal{F}_2$ denotes the equivalence class of $F_1 \oplus F_2$ with respect to \sim , for some $F_1 \in \mathcal{F}_1$ and $F_2 \in \mathcal{F}_2$. \square

The parallel composition of Markov reward chains with silent transitions is well defined as the Kronecker sum respects the equivalence \sim . Next we state the compositionality result for τ_{\sim} -lumping and τ_{\sim} -reduction. It is a direct consequence of Theorem 7.4.10 for compositionality of τ -lumping and τ -reduction, and compositionality of ordinary lumping for standard Markov reward chain as a special case of Theorem 7.4.7.

Theorem 7.4.12 Let S_1 and S_2 be two Markov reward chains with silent transitions. If $S_1 \xrightarrow{\mathcal{L}_1} \bar{S}_1$ and $S_2 \xrightarrow{\mathcal{L}_2} \bar{S}_2$, then $S_1 \parallel S_2 \xrightarrow{\mathcal{L}_1 \otimes \mathcal{L}_2} \bar{S}_1 \parallel \bar{S}_2$. Also, if $S_1 \xrightarrow{\mathcal{L}_1}_r M_1$ and $S_2 \xrightarrow{\mathcal{L}_2}_r M_2$, then $S_1 \parallel S_2 \xrightarrow{\mathcal{L}_1 \otimes \mathcal{L}_2}_r M_1 \parallel M_2$. \square

7.5 Summary

We consider three types of performance models. Markov reward chains with fast transitions are our central model used for analyzing systems with stochastic and instantaneous probabilistic transitions. Their limits are the discontinuous Markov reward chains. Their quotients are the Markov reward chains with silent transitions which can be used for the analysis of systems with stochastic transitions and nondeterministic (internal) τ steps.

For each type of models, we present two aggregation methods: lumping and reduction for discontinuous Markov reward chains, τ -lumping and τ -reduction for Markov reward chains with fast transitions, and τ_{\sim} -lumping and τ_{\sim} -reduction for Markov reward chains with silent transitions.

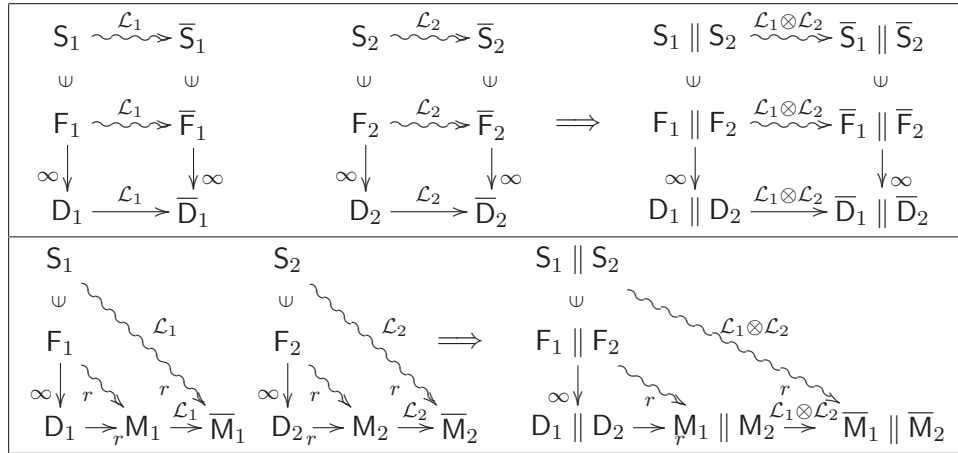


Figure 7.12: Summary compositionality results

In short, the contributions of this chapter are: (1) A definition of parallel composition of Markov rewards chains with stochastic discontinuity, fast, and silent transitions. (2) Identification of preorder properties of the aggregation methods for all types of models. (3) Compositionality theorems for each type of models and each corresponding aggregation preorder, and a continuity property of the parallel compositions. The results on compositionality are summarized by Figure 7.12 which is justified by the Theorems 7.3.1–7.4.12, as well as by Proposition 7.2.10 and Proposition 7.2.17.

Next, we illustrate the features of the process theories developed so far by specifying and analyzing the concurrent alternating bit protocol with real timeouts and stochastically distributed lossy channels.

Chapter 8

Analyzing the Concurrent Alternating Bit Protocol

In this chapter, we illustrate the theories and methods developed so far by analyzing a version of the concurrent alternating bit protocol with lossy channels. The protocol comprises real timeouts and generally-distributed channels, which makes it suitable for specification in TCP^{drst} . For the purpose of performance analysis, we choose the framework of the language χ as it provides means for Markovian analysis and simulation for generally-distributed delays from the same specification. To enable performance evaluation in discrete real time as well, we augment the environment with a prototype extension that supports the analysis of models comprising immediate probabilistic choices and deterministic delays. We refer to the model as a discrete-time probabilistic reward graph and we develop two methods for its analysis by translating it to a discrete-time Markov reward chain.

8.1 The Language χ

The language χ is a modeling language for control and analysis of industrial systems (machines, manufacturing lines, warehouses, factories, etc.) [16]. It has been successfully applied to a large number of industrial cases, such as a car assembly line, a multi-product multi-process wafer fab [34], a fruit juice blending and packaging plant [47], and process industry factories [17]. Initially, χ came equipped with features for the modeling of discrete-event systems only, and was not supported by a formal semantics. Later, it was redesigned and converted to a formal timed specification language [25]. At present, χ can be characterized as a process algebra with data. In addition, it was extended to handle both discrete-event and continuous aspects, allowing for the modeling of hybrid systems [16].

Originally, simulation was the only means to analyze χ models. For the verification of functional requirements, however, simulation is not sufficient. Although it can, for instance, reveal that a system has a deadlock or that the system may exhibit a specific behavior, it cannot show that the system is deadlock-free nor that it will persist having the specific behavior. Therefore, a new approach has been taken, connecting χ to state-of-the-art verification tools and techniques. Currently, a χ model can be compiled to the input language of a number of model checkers, including SPIN [56, 93], μ CRL [21, 97] and UPPAAL [63, 24] (cf. Figure 1.11). The translated model can subsequently be checked against the functional properties formulated in the target setting.

Successful verification is usually succeeded by performance analysis and design optimization. At present, performance analysis of a χ model can be carried out either by simulation, or by analysis of the underlying continuous-time Markov (reward) chain (cf. Figure 1.11). Simulation is a powerful method for performance analysis, but its disadvantages in comparison to analytical methods are well-known [15]. The approach based on Markov chain turns χ into a powerful stochastic process algebra in the vein of [51, 55]. It is analytical, and builds on a vast and well-established theory. However, the generation of a Markov chain from a χ model requires that all delays in the system are exponentially distributed. This is a serious drawback since in industrial systems, particularly in controllers, delays are often closer to being deterministic. Although it is possible to approximate deterministic delays by sequences of exponential delays, i.e. to model them by so-called phase-type distributions [82], this approach suffers from the state explosion problem. Many states are needed to correctly approximate these delays, and the generated Markov chain becomes large due to the full interleaving of stochastic transitions in parallel contexts.

In this chapter, we propose a model in which time delays are discrete and deterministic, while random behavior is expressed in terms of immediate probabilistic choices. This model is referred to as *discrete-time probabilistic reward graphs*. We define a method for obtaining performance measures of a discrete-time probabilistic reward graph by transforming it to a discrete-time Markov reward chain [61]. We augment the χ environment so that for a given χ specification, the corresponding discrete-time probabilistic reward graph can be obtained automatically. Usually, in contrast to the Markov chain approach, the discrete-time probabilistic reward graph generated from a χ -model is considerably smaller (more than threefold for our case study). In a discrete-time probabilistic reward graph, time itself does not decide a choice and, as such, interleaving of timed transitions does not occur as

in typical timed process algebras [10]. As an illustration, a case study is discussed on the performance of a turntable drilling system. Although compact, this system incorporates many complex modeling issues. The case has been studied previously to illustrate the verification techniques of functional requirements [25, 23]. We put the new performance results exploiting discrete-time probabilistic reward graphs in perspective, by comparing them to results from simulation and the approach exploiting Markov chains.

8.2 Discrete-Time Probabilistic Reward Graphs

In this section we introduce the notion of a discrete-time probabilistic reward graph, and give, regarding performance, two equivalent Markovian interpretations: one straightforward and general, the other more specific, but computationally more efficient.

Discrete-time probabilistic reward graphs are transition systems with two types of states: (1) probabilistic, which have finitely many probabilistic outgoing transitions and (2) timed, which have only one outgoing transition. This is formalized in the following definition.

Definition 8.2.1 A discrete-time probabilistic reward graph is a tuple $G = (\sigma, S, \dashrightarrow, \mapsto, \rho)$, where

1. $\sigma \in \mathbb{R}^{1 \times |S|}$ is an *initial state probability row vector*;
2. S is the set of states partitioned as $\{S_p, S_t\}$, where S_p and S_t are the sets of probabilistic and timed states, respectively;
3. $\dashrightarrow \subseteq S_p \times (0, 1] \times S$ is an (immediate) *probabilistic transition relation*;
4. $\mapsto \subseteq S_t \times \mathbb{N}^+ \times S$ is a *timed transition relation* such that $s \xrightarrow{n} s'$ and $s \xrightarrow{m} s''$ (in infix notation) implies $n = m$ and $s' = s''$; and
5. $\rho \in \mathbb{R}^{|S| \times 1}$ is a *state reward rate vector*. □

The interpretation of a discrete-time probabilistic reward graph is as follows. In probabilistic states the process spends no time, and it jumps to a next state chosen according to the probabilistic transition relation. In a timed state the process spends as many time units as specified by the timed transition relation, and jumps to the unique subsequent state. The uniqueness requirement is to support the time-determinism property [84, 11, 10]. A reward is gained per time unit, as determined by the reward rate assigning function. Although we allow reward rates to be assigned also to probabilistic

states, the process actually gains no reward as it spends no time in them. This statement will also be supported by the aggregation method used below (cf. also Section 7.2).

We visualize a discrete-time probabilistic reward graph as in Figure 8.1a. Here, states 1, 2, and 3 are timed, whereas states 4 and 5 are probabilistic. The reward rates are put in sans-serif at the top right corner of each state; the reward rate of the state i is r_i , for $1 \leq i \leq 5$.

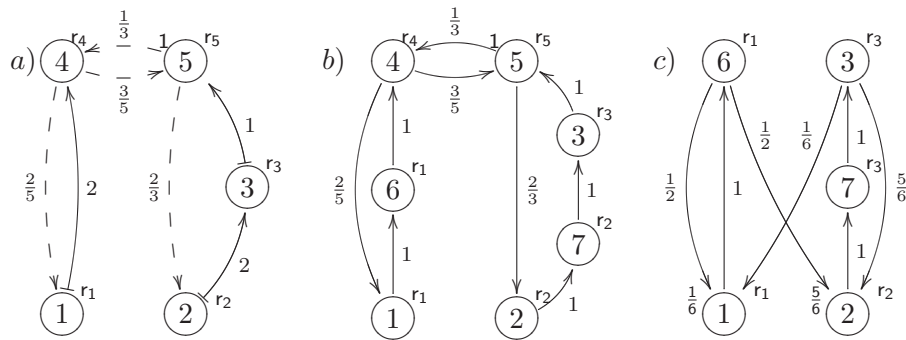


Figure 8.1: a) A discrete-time probabilistic reward graph, b) its unfolding, and c) aggregated unfolding

To obtain the performance measures of a discrete-time probabilistic reward graph we exploit a relation with discrete-time Markov reward chains, as the latter are well-established models for performance analysis. We show how to represent a discrete-time probabilistic reward graph to an equivalent discrete-time Markov reward chain, which is then analyzed, in the end to interpret the results back to the discrete-time probabilistic reward graph setting. The translation is performed in two steps: first the discrete-time probabilistic reward graph is transformed to a transition system that can be interpreted as a discrete-time Markov reward chain, and afterwards the discrete-time Markov reward chain is adapted to truthfully represent the semantics of the original process by an aggregation that eliminates the immediate probabilistic transitions. We need to interchangeably treat discrete-time Markov reward chains both as transition systems and in matrix terms. First, we give the notion of a discrete-time Markov reward chain in terms of transition systems.

Definition 8.2.2 A discrete-time Markov reward chain $M = (\sigma, S, \longrightarrow, \rho)$ is a tuple where

- $\sigma \in \mathbb{R}^{1 \times |S|}$ is the initial state probability row vector;
- S is a finite set of states;
- $\longrightarrow \subseteq S_p \times (0, 1] \times S$ is the *probabilistic transition relation*; and
- $\rho \in \mathbb{R}^{|S| \times 1}$ is the state reward vector. □

Operationally, a discrete-time Markov reward chain is considered to wait one time unit in a state, gain the reward for this state determined by the reward vector ρ , and immediately jumps to another state with a probability specified by the relation \longrightarrow .

When required by the context, we will have occasion to represent a discrete-time Markov reward chain as a pair (σ, P, ρ) , where P is the probability transition matrix, i.e., the matrix representation of the probability transition relation, and ρ is the state reward vector. It is known that $P(n)$, the transition probabilities after $n > 0$ time steps are given by $P(n) = P^n$. Also, the long-run probability vector $\pi \in \mathbb{R}^{|S|}$, i.e., the average probability that the process resides in a given state after the system stabilizes, satisfies $\pi P = \pi$. For more details, we refer to the standard literature (e.g., [61, 37]).

The main idea behind the translation from a discrete-time probabilistic reward graph G to a discrete-time Markov reward chain M is to represent a timed transition of duration n of G as a sequence of n states in M , connected by probabilistic transitions with probability 1, all having the same reward. The immediate probabilistic transitions of G remain unchanged by this transformation. Thus, the immediate probabilistic transitions of G are wrongly transformed to probabilistic transitions of M that last one time unit. We come back to this problem later. First, we give the naive transformation to a discrete-time Markov reward chain, which we refer to as the *unfolding* of a discrete-time probabilistic reward graph.

Definition 8.2.3 Let $G = (\sigma_G, S_G, \dashrightarrow, \dashrightarrow, \rho_G)$ be a discrete-time probabilistic reward graph with $S_G = \{s_1, \dots, s_n\}$. Associate with every state $s_i \in S_G$ a number $m_i \in \mathbb{N}^+$ as follows: if s_i is a probabilistic state, then $m_i = 1$; if s_i is a timed state, then $m_i = m$ for the unique m such that $s_i \xrightarrow{m} s_k$, for some $s_k \in S_G$.

Then, the *unfolding* of G is the discrete-time Markov reward chain $U = (\sigma_U, S_U, \longrightarrow, \rho_U)$ where $S_U = \{s_{ij} \mid 1 \leq i \leq n, 1 \leq j \leq m_i\}$ and

1. $\sigma_U(s_{i1}) = \sigma_G(s_i)$ and $\sigma_U(s_{ij}) = 0$ for $1 < j \leq m_i$;
2. $s_{ij} \xrightarrow{1} s_{i,j+1}$ for $1 \leq j \leq m_i - 1$, and $s_{im_i} \xrightarrow{1} s_{k1}$ if $s_i \xrightarrow{m} s_k$ or $s_{i1} \xrightarrow{p} s_{k1}$ if $s_i \dashrightarrow^p s_k$; and

3. $\rho_U(s_{ij}) = \rho_G(s_i)$ for $1 < j \leq m_i$.

The set of probabilistic states of U is given by $S_{U,p} = \{s_{i1} \mid s_i \in S_{G,p}\}$ and the set of timed states is given by $S_{U,t} = S_U \setminus S_{U,p}$. The *unfolding set* of s_i is given by $US(s_i) = \{s_{ij} \mid 1 \leq j \leq m_i\}$. The starting state of the unfolding of s_i is given by $us(US(s_i)) = s_{i1}$. \square

Remark 8.2.4 The states of an unfolding of a discrete-time probabilistic reward graph can be partitioned to probabilistic and timed states as given by Definition 8.2.3. In the matrix representation of the unfolding $U = (\sigma_U, P, \rho_U)$, the transition matrix P induces two transition matrices P_t and P_p . The transition matrix P_t represents the unfolded timed transitions that originate from the timed states of $S_{G,t}$, whereas P_p holds the translated immediate probabilistic transitions that originate from the probabilistic states of $S_{G,p}$. To obtain these matrices the transition matrix P is first split to $P = P'_t + P'_p$ according to the timed and probabilistic transitions, respectively. The matrices P'_t and P'_p have to be adapted to transition matrices by adding 1s on the diagonal of the zero rows, where the other type of transitions are missing. \square

We illustrate the situation by an example.

Example 8.2.5 The unfolding of the discrete-time probabilistic reward graph from Figure 8.1a is given by the discrete-time Markov reward chain depicted in Figure 8.1b. The unfolded timed delays originating from states 1 and 2 introduce the new states 6 and 7, respectively. Here the timed states are $\{1, 2, 3, 6, 7\}$ and the probabilistic states are $\{4, 5\}$. The transition matrix of the timed and probabilistic transitions are given by

$$P_t = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad P_p = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{2}{5} & 0 & 0 & 0 & \frac{3}{5} & 0 & 0 \\ 0 & \frac{2}{3} & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} .$$

\square

As hinted above, the discrete-time Markov reward chain obtained by the unfolding, in general, does not truthfully represent the semantics of the original discrete-time probabilistic reward graph, in the sense that probabilistic states are immediate in the discrete-time probabilistic reward graph, whereas

they last one unit of time in the discrete-time Markov reward chain. For example, in the discrete-time probabilistic reward graph from Figure 8.1a, state 5 can be reached from state 1 with probability $\frac{1}{2}$ after a delay of 2 time units (via $1 \xrightarrow{2} 4 \xrightarrow{1/2} 5$), whereas in the unfolded version this cannot be done in less than 3 time units (that are required for a sojourn in the states 1, 6, and 4).

The solution to this problem is to eliminate the immediate probabilistic states appropriately. The elimination is achieved by the reduction-based aggregation method of Section 7.2, suitably adapted for the discrete-time setting. Intuitively, in the new setting the method computes the accumulative probability of reaching one timed state from another and adjusts the delays. More specifically, the process of aggregation is as follows: In a unfolding $U = (\sigma, P, \rho)$ the transition probability matrix P is split to the transition matrices of the timed and probabilistic transitions P_t and P_p , respectively. Next, the Cesaro sum of the transition matrix induced by P_p , given by

$$\Pi = \lim_{n \rightarrow \infty} \sum_{i=0}^n \frac{P_p + P_p^2 + \dots + P_p^n}{n},$$

is computed and its canonical product decomposition (L, R) is found (cf. Definition 7.2.13). The Cesaro sum plays the role of the ergodic projection in Definition 8.2.6 for the discrete-time case [61]. It represents the ergodic projection at one of the transition matrix P_p and it satisfies $\Pi P = P \Pi = \Pi$ [61]. Finally, the *aggregated* process is given by $M = (\sigma R, L P_t R, L \rho)$ as in Definition 7.2.14. We specify the aggregation method by the following definition.

Definition 8.2.6 Let G be a discrete-time probabilistic reward graph and $U = (\sigma, P, \rho)$ be its unfolding where P induces P_t and P_p . The translation by unfolding of G is the discrete-time Markov reward chain $M = (\bar{\sigma}, \bar{P}, \bar{\rho})$, given by $\bar{\sigma} = \sigma R$, $\bar{P} = L P R$, and $\bar{\rho} = L \rho$, where (L, R) is a canonical product decomposition of the Cesaro sum of P_p . \square

The translation preserves the unfolding sets of the timed transitions of G and their starting states. Only the probabilistic states are eliminated and the transitions of the last states in the unfolding of the timed transitions in U are adjusted in M . We note that the translation by unfolding has more states than the original process in the order of the sum of the duration of all timed transitions. We illustrate the translation by an example.

Example 8.2.7 The discrete-time Markov reward chain in Figure 8.1c is the aggregated chain of the one in Figure 8.1b. The aggregation eliminates the probabilistic states 4 and 5 and splits the incoming timed transitions from the states 6 and 3. The splitting is according to the accumulative (trapping) probabilities of 4 and 5 to the timed states 1 and 2 (which actually represent ergodic classes in the terminology of Section 7.1). Thus, in the aggregated chain there are two outgoing transitions from the states 6 and 3 to 1 and 2 (instead of a single one in the unfolded chain). The aggregation method conforms to the Markovian semantics that after a delay of one time unit there is an immediate probabilistic choice, which in the unfolded discrete-time Markov reward chain is explicitly stated by the immediate probabilistic transitions. It is straightforwardly checked that the discrete-time Markov reward chain in Figure 8.1c models the same system as the discrete-time probabilistic reward graph in Figure 8.1a when the discrete-time probabilistic reward graph is observed in the states 1, 2, and 3. \square

Remark 8.2.8 An alternative and more evident, but possibly analytically and computationally intractable approach would be to translate and analyze discrete-time probabilistic reward graphs as semi-Markov reward chains [57]. It is not difficult to observe that discrete-time probabilistic reward graphs resemble a very simple class of semi-Markov reward chains with deterministic distributions. However, to obtain the form of a semi-Markov reward chain, the aggregation by reduction still has to be applied to eliminate subsequent probabilistic transitions and probabilistic transitions must be introduced between subsequent timed transitions. Recently, a recurrence-relation-based tailored analysis approach for discrete-time semi-Markov processes has been proposed in [92]. \square

The following lemma gives an important property of the long-run probability vector of the unfolding in terms of a relation between the states that belong to the same unfolding set. The result supports the assignment of the same reward to all states in an unfolding of a timed transition as in Definition 8.2.3. It also plays a role in the proof of an optimization technique described below.

Lemma 8.2.9 *Let π be the long-run probability vector of the translation of a discrete-time probabilistic reward graph G . Then for every state $k \in S_{G,t}$ and $i, j \in \text{US}(k)$ it holds that $\pi[i] = \pi[j]$.* \square

PROOF Let P be the transition probability matrix of the translation. As π is the long-run probability vector, it holds that $\pi P = \pi$. Now, assume that

$i, j \in \text{US}(k)$, for some timed state k of \mathbf{G} , are two subsequent states of the unfolding sequence of some timed transition, i.e., $i \xrightarrow{1} j$. Then, $P[i, j] = 1$ and $P[i', j] = 0$ for every $i' \neq i$, as there are no other incoming transitions to j . Now, it can be observed that $\pi[j] = \pi P^{(-:j)} = \pi[i]$, where $P^{(-:j)}$ denotes the j -th column of P . Hence $\pi[i] = \pi[j]$, for any two subsequent states in $\text{US}(k)$, which completes the proof. ■

Next, we investigate how to related the translation back to the original process.

With the transformation of a discrete-time probabilistic reward graph into a discrete-time Markov reward chain in place, we can use the standard theory and tools to compute all common performance measures. As before, our focus is on the expected reward rate after n time units or in the long-run.

If the resulting discrete-time Markov reward chain is ergodic, the expected reward at time step n is standardly computed as $R(n) = \sigma P(n) \rho$ and the long-run reward as $R^\infty = \pi \rho$, where (σ, P, ρ) is the translated discrete-time Markov reward chain, $P(n)$ is its transition probability matrix, and π is its long-run probability vector. In case the resulting process is not ergodic, we can always partition the original discrete-time probabilistic reward graph into subgraphs that produce ergodic processes and analyze them separately. So, we do not consider the ergodicity condition as restrictive to our analysis and from now on we assume that we work only with ergodic processes. After determining the performance metric of the discrete-time Markov reward chain we have to interpret the obtain result back in the discrete-time probabilistic reward graph setting.

To give the backward relation between the discrete-time probabilistic reward graph \mathbf{G} and its translation \mathbf{M} we use specially adapted distributor and collector matrices. The idea is to fold back the unfolded timed transitions and restore the effect of the probabilistic transitions in \mathbf{G} by multiplying the transition matrix of \mathbf{M} with these matrices. In that way, we can obtain the transition matrix and, consequently, the expected reward of \mathbf{G} .

First, we define the folding collector matrix of the unfolding \mathbf{U} of \mathbf{G} as the collector of the partition induced by the unfolding sets. Due to the reduction-based aggregation, all probabilistic states have been eliminated to obtain the translation \mathbf{M} . So, the folding distributor and collector of \mathbf{U} have too many states, as they also account for the already eliminated probabilistic transitions, and they have to be shrunk. Therefore, the rows and columns corresponding to the eliminated probabilistic transitions are omitted to obtain the folding distributor and collector of \mathbf{M} .

The multiplication of the transition matrix of \mathbf{M} with its folding collector

produces the accumulative probability of residing in each unfolded timed state of M per unfolding set. So, the probabilities of residing in a timed state in the original process G can be extracted as the folded probability of the starting timed state of every unfolded timed transition. This is carried out by multiplying the folded transition matrix with a folding distributor that extracts only the probabilities of the starting states of the unfolding of each transition. The folding distributor and collector matrices of the unfolding U and the translation M are defined as follows.

Definition 8.2.10 Let G be a discrete-time probabilistic reward graph, U its unfolding, and M its translation. The folding collector matrix V_U of U is given by $V_U[i, j] = 1$ iff $j \in \text{US}(i)$ and $V_U[i, j] = 0$ otherwise, for $i, j \in S_U$. The folding distributor U_U is given by $U_U[i, j] = 1$ iff $j = \text{us}(\text{US}(i))$ and $U_U[i, j] = 0$ otherwise. The folding distributor and collector matrix U_M and V_M of M are obtained by omitting the rows and columns of U_U and V_U , respectively, that correspond to the probabilistic states in S_U, p . \square

The folding collector V_M of the translation M has the following property given as a corollary of Lemma 8.2.9.

Corollary 8.2.11 Let G be a discrete-time probabilistic reward graph and M its translation. Let π be the long-run probability vector of M , V the folding collector of M , and U some distributor corresponding to V . Then, $\pi = \pi V U$. \square

PROOF Pick $i \in M$. Let k be the state such that $i \in \text{US}(k)$. Then,

$$(\pi V U)[i] = \sum_{j \in \text{US}(k)} \pi[j] U[j, i] = \pi[i] \sum_{j \in \text{US}(k)} U[j, i] = \pi[i] \cdot 1 = \pi[i],$$

which completes the proof. \blacksquare

Intuitively, the corollary states that folding the long-run probabilities of the unfolded timed states in the translation can be done using the folding collector and an arbitrary distributor. So, we can reconstruct the behavior of the timed states in the original process G . However, the folding distributor and collector matrices cannot restore the behavior of the probabilistic states. Recall that we used the canonical decomposition (L, R) of the Cesaro sum Π to obtain the translation M from the unfolding U . To properly eliminate the effect of the probabilistic transitions the folding distributor U_U has to be multiplied by R to the right, obtaining $R_M = U_U R$, whereas the folding collector V_U is multiplied by L to the left obtaining $L_M = L V_U$. The

matrices L_M and R_M no longer have the form of a distributor and collector matrix.

Now, we have all prerequisites to define $P_G(n)$, the transition matrix after n time steps of the discrete-time probabilistic reward graph G . It is given by $P_G(n) = R_M P_M(n) L_M$. It should not be difficult to see from Definition 8.2.3 and Definition 8.2.6 that we also have $\sigma_M = \sigma_G R_M$ and $\rho_M = L_M \rho_G$. Then,

$$R_M(n) = \sigma_M P_M(n) \rho_M = \sigma_G U_M P_M(n) V_M \rho_G = \sigma_G P_G(n) \rho_G = R_G(n).$$

Similarly, we put $\pi_G = \pi_M L_M$ for the long-run probabilities. Then,

$$R_M^\infty = \pi_M \rho_M = \pi_M L_M \rho_G = \pi_G \rho_G = R_G^\infty.$$

Remark 8.2.12 The translation of the discrete-time probabilistic reward graph G can also be given directly by means of discrete-time Markov reward chains with fast transitions, as the counterpart of the Markov reward chains with fast transitions given in Section 7.1. Actually, we have implicitly used such an interpretation as

$$P_G(n) = R_M P_M(n) L_M = U_U R L P'(n) R L V_U = U_U I I P_U(n) I I V_U.$$

By recalling Definition 7.1.5 of the limit of a Markov reward chain with fast transitions it is clear that we treat discrete-time probabilistic reward graphs as folded limits of discrete-time Markov reward chains with fast transitions, where the fast transitions model the immediate probabilistic choices. However, we believe that the transformation in two steps given in the current setting is natural and contributes to the clarity of the presentation.

We also note that the lumping condition does not hold for L_M and R_M (as hinted by their names), i.e., in general, $L_M R_M P_M(n) L_M \neq P_M(n) L_M$. As a consequence, the possibility (and means) of computing $P_G(n)$ using $P_G(n-1)$ is not immediately clear. Thus, in the current setting, for transient analysis of a discrete-time probabilistic reward graphs we resort to computing the bigger transition probability matrix $P_M(n)$ of its translation M and folding it back using the specially adapted matrices L_M and R_M as elaborated above. \square

We illustrate the situation by an example.

Example 8.2.13 The folding distributor and collector matrix of the unfolding U in Figure 8.1b of the discrete-time probabilistic reward graph G in

Figure 8.1a are given by

$$U_U = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad V_U = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}.$$

The canonical decomposition (L, R) of the Cesaro sum of the transition matrix of the immediate probabilistic transitions is given by

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad R = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{5}{6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

The folding distributor and collector matrices of the transition M depicted in Figure 8.1c are given by

$$U_M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad V_M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

The adapted versions R_M and L_M of the folding distributor and collector are given by:

$$R_M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{5}{6} & 0 & 0 & 0 \end{pmatrix} \quad L_M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix},$$

where the states 6 and 7 have been renumbered to 4 and 5 in the matrix representation.

The initial probability vector σ_M and the reward vector ρ_M are given by

$$\sigma_M = (0 \ 0 \ 0 \ 0 \ 1) \quad R_M = \left(\frac{1}{6} \ \frac{5}{6} \ 0 \ 0 \ 0\right) \quad \rho_M = L_M \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_1 \\ r_2 \end{pmatrix}.$$

The probability transition matrix of G after 1, 2, and 3 time units is given by

$$P_G(1) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{5}{6} & 0 & 0 & 0 \end{pmatrix} \quad P_G(2) = \begin{pmatrix} \frac{1}{6} & \frac{5}{6} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{12} & \frac{5}{12} & \frac{1}{2} & 0 & 0 \\ \frac{1}{36} & \frac{5}{36} & \frac{5}{6} & 0 & 0 \end{pmatrix} \quad P_G(3) = \begin{pmatrix} \frac{1}{6} & \frac{5}{6} & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{12} & \frac{5}{12} & \frac{1}{2} & 0 & 0 \\ \frac{1}{3} & \frac{2}{3} & 0 & 0 & 0 \\ \frac{4}{9} & \frac{5}{9} & 0 & 0 & 0 \end{pmatrix}.$$

We can directly check the correspondence with the execution of the discrete-time probabilistic reward graph depicted in Figure 8.1. Note that the process never resides in the probabilistic states 4 and 5.

The long-run expected reward rate of the discrete-time probabilistic reward graph depicted in Figure 8.1a is obtained from the long-run probability vector π_M of its translation of Figure 8.1c. This vector is

$$\pi_G = \pi_M L_M = \left(\frac{1}{11} \ \frac{3}{11} \ \frac{3}{11} \ \frac{1}{11} \ \frac{3}{11}\right) L_M = \left(\frac{2}{11} \ \frac{6}{11} \ \frac{3}{11} \ 0 \ 0\right).$$

Note that the long-run probability vector of G has 0s for the places of the probabilistic states. The long-run expected reward rate of G is

$$R_G^\infty = \pi_G \rho_G = \left(\frac{2}{11} \ \frac{6}{11} \ \frac{3}{11} \ 0 \ 0\right) \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix} = \frac{2}{11} r_1 + \frac{6}{11} r_2 + \frac{3}{11} r_3.$$

It is the same as the long-run probability vector of M , i.e.,

$$R_M^\infty = \pi_M \rho_M = \left(\frac{1}{11} \ \frac{3}{11} \ \frac{3}{11} \ \frac{1}{11} \ \frac{3}{11}\right) \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_1 \\ r_2 \end{pmatrix} = \frac{2}{11} r_1 + \frac{6}{11} r_2 + \frac{3}{11} r_3. \quad \square$$

We can visualize the full process of obtaining the performance measures of a discrete-time probabilistic reward graph by means of translation by unfolding in the left branch in Figure 8.2.

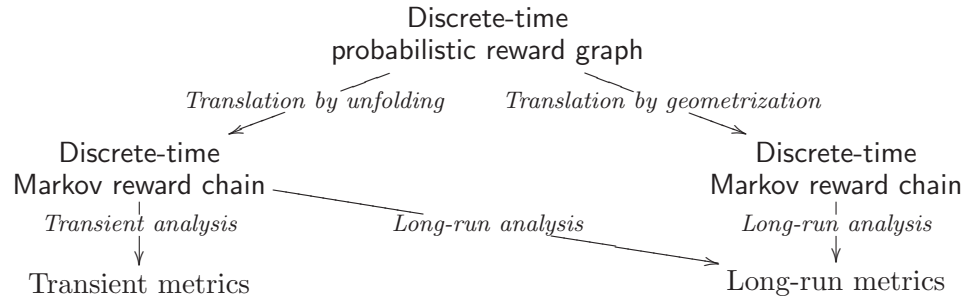


Figure 8.2: Performance measuring for discrete-time probabilistic reward graphs

The analysis of a discrete-time probabilistic reward graph by its translation to a discrete-time Markov reward chain using the approach described above introduces extra states that are required for the unfolding of the timed transitions. In the following section we give an optimized translation in case of long-run analysis.

Note that the unfolded discrete-time Markov reward chain can have, in general, substantially more states than the original discrete-time probabilistic reward graph, as every delay of duration n introduces $n - 1$ new states. This means that the translation by unfolding, although straightforward to serve as a definition, leads to computations on large state spaces. In the rest of this section, we optimize our method, using ‘geometrization’ of time delays to obtain a discrete-time Markov reward chain of, at most, the size of the original graph. This discrete-time Markov reward chain has the same long-run expected reward rate as the one translated by unfolding. The main idea is to replace discrete delays by geometrically distributed ones with the same mean instead of unfolding them. First, we define the geometrization of a discrete-time probabilistic reward graph.

Definition 8.2.14 Let $G = (\sigma, S, \dashrightarrow, \dashrightarrow, \rho)$ be a discrete-time probabilistic reward graph. Then, the *geometrization* of G is the discrete-time Markov reward chain $W = (\sigma, S, \longrightarrow, \rho)$, if

1. for each timed transition $s \dashrightarrow^n s'$ in G we have the two transitions $s \xrightarrow{1/n} s'$ and $s \xrightarrow{(n-1)/n} s$ in W ; and

2. for each probabilistic transition $s \xrightarrow{p} s'$ in G we have $s \xrightarrow{p} s'$ in M . \square

The geometrization of a timed transition in G replaces the transition by two transitions in W such that they induce a geometric sojourn time in the state with mean equal to the duration of the timed transition. As before, to obtain the final discrete-time Markov reward chain it is required to eliminate the probabilistic transitions by reduction-based aggregation. However, the translation by geometrization is not adequate for transient analysis as it does not truthfully depict the semantics of G . Still, we will show that the long-run expected reward of the discrete-time Markov reward chains obtained by translating the same discrete-time probabilistic reward graph by unfolding and geometrization is the same. First, we illustrate the translation by geometrization.

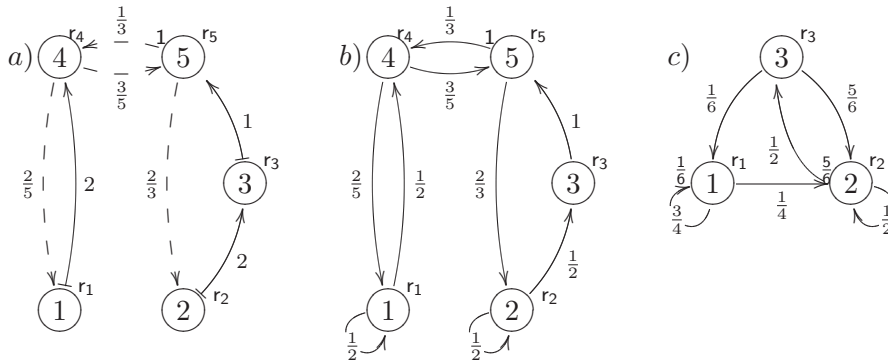


Figure 8.3: a) A discrete-time probabilistic reward graph, b) its geometrization, and c) aggregated geometrization

Example 8.2.15 Consider again the discrete-time probabilistic reward graph from Figure 8.1a, repeated in Figure 8.3a. Figure 8.3b depicts its geometrization. For the same reasons as discussed above, the discrete-time Markov reward chain obtained by geometrization still needs to be aggregated. The result of the complete translation is depicted in Figure 8.3c. \square

The translation by geometrization is depicted by the right branch in Figure 8.2. To show that the two translations indeed commute, i.e., they give rise to discrete-time Markov reward chains with the same long-run performance measure, we need to find the relation between the resulting processes.

Again, we turn to the matrix representation of a discrete-time Markov reward chain and we relate the two methods by using the folding collector and a special *uniform distributor*. The uniform distributor \bar{U} is defined as the distributor corresponding to the folding collector, in which the distribution coefficients corresponding to the states in the same partitioning class are equal. For example, the uniform distributor corresponding to the folding collector V_U in Example 8.2.13 of the unfolding U depicted in Figure 8.1b is given by

$$\bar{U} = \begin{pmatrix} \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

It can be directly checked that the discrete-time Markov reward chain depicted in Figure 8.3b is obtained by multiplying the transition matrix of the one depicted in Figure 8.1 with the uniform distributor \bar{U} and the folding collector V from Example 8.2.13.

One observes that this holds in general, because when a timed transition $s_i \xrightarrow{m} s_j$ is unfolded to $s_{i1} \xrightarrow{1} \dots \xrightarrow{1} s_{im} \xrightarrow{1} s_{j1}$, then the multiplication of the transition probability matrix P by the uniform distributor \bar{U} on the left, and by the folding collector V on the right, transforms the sequence into two transitions $US(s_i) \xrightarrow{1/m} US(s_j)$ and $US(s_i) \xrightarrow{(m-1)/m} US(s_i)$. This directly corresponds to geometrizing the delays of the original discrete-time probabilistic reward graph as given by Definition 8.2.14, after the renaming of $US(s)$ to s . Thus, by folding the unfolding of a discrete-time probabilistic reward graph using the uniform distributor, we obtain the geometrization of the discrete-time probabilistic reward graph.

The following theorem states that the translations produce discrete-time Markov reward chains with the same long-run expected reward race.

Theorem 8.2.16 *Let G be a discrete-time probabilistic reward graph, M_1 its translation by unfolding, and M_2 its translation by geometrization. Then $R_{M_1}^\infty = R_{M_2}^\infty$. \square*

PROOF Let the $U = (\sigma, P, \rho)$ be the unfolding of G . Let P_t and P_p be the transition matrices of the timed nad probabilistic transitions and (L_1, R_1) be the canonical product decomposition of the Cesaro sum of P_p as given by Definition 8.2.6. Then $M_1 = (\sigma R_1, L_1 P_t R_1, L_1 \rho)$ is the translation by unfolding. Let (L_2, R_2) be the canonical product decomposition of $\bar{U}_U P_p V_U$

required to give the translation by geometrization. Then

$$\mathbf{M}_2 = (\sigma V_{\mathbf{U}} R_2, L_2 \bar{U}_{\mathbf{U}} P_t V_{\mathbf{U}} R_2, L_2 \bar{U}_{\mathbf{U}} \rho)$$

is the translation by geometrization.

First, we will show that $\pi_{\mathbf{M}_1} V_{\mathbf{M}_1}$ is the long-run probability vector of \mathbf{M}_2 , where $V_{\mathbf{M}_1}$ is the folding collector of \mathbf{M}_1 , and, then, as a consequence it will follow that $R_{\mathbf{M}_1}^\infty = R_{\mathbf{M}_2}^\infty$. We note that this result is stronger than the one stated by the theorem, as it gives the relation between the long-run probability vectors of the translations by unfolding and geometrization.

Without loss of generality, we assume that \mathbf{G} has k timed transitions, ℓ closed loops of probabilistic transitions, and m open loops or sequences of probabilistic transitions. They correspond to $t_1 + \dots + t_k$ trivial ergodic classes of one element for the duration of the timed transitions t_1, \dots, t_k , ℓ ergodic classes with more than one element, and m transient states (cf. Section 7.1). To alleviate the computations, again without loss of generality, we assume a numbering of the states such that unfolding sets contain states with consecutive indices, after which we place the closed loops, and finally, the transient states.

For such a numbering, the matrices $\bar{U}_{\mathbf{M}_1}$ (the uniform distributor corresponding to $V_{\mathbf{M}_1}$), $V_{\mathbf{M}_1}$, L_1 , and R_1 , have the following form:

$$U_{\mathbf{M}_1} = \begin{pmatrix} u_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & u_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & 1 \end{pmatrix} \quad V_{\mathbf{M}_1} = \begin{pmatrix} \mathbf{1}^{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & \mathbf{1}^{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & 1 & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & 1 \end{pmatrix}$$

$$L_1 = \begin{pmatrix} I^{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & I^{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mu_1 & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mu_\ell & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}^m \end{pmatrix} \quad R_1 = \begin{pmatrix} I^{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & I^{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1}^{E_1} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{1}^{E_\ell} & \mathbf{0} \\ (\delta_{1\mathbf{0}}) & \dots & (\delta_{k\mathbf{0}}) & d_1 & \dots & d_\ell & \mathbf{0}^m \end{pmatrix}$$

where u_{t_1}, \dots, u_{t_k} are uniformly distributed positive stochastic row vectors, μ_1, \dots, μ_ℓ are the ergodic probability row vectors of the ergodic classes,

$\delta_1, \dots, \delta_k$ are the transient probability vectors of the first states in the unfolding sequences, and $(\delta_i \mathbf{0})$ is a square matrix where the first column is δ_i for $1 \leq i \leq k$, and d_1, \dots, d_l are the transient probability vectors of the ergodic classes.

The matrices \bar{U}_U, V_U, L_2 , and R_2 have the following form:

$$\bar{U}_U = \begin{pmatrix} u_{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & u_{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I^{E_1} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & I^{E_\ell} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & I^m \end{pmatrix} \quad V_U = \begin{pmatrix} \mathbf{1}^{t_1} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & \mathbf{1}^{t_k} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & I^{E_1} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & I^{E_\ell} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & I^m \end{pmatrix}$$

$$L_2 = \begin{pmatrix} 1 & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mu_1 & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mu_\ell & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}^m \end{pmatrix} \quad R_2 = \begin{pmatrix} 1 & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0} & \dots & 1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1}^{E_1} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{1}^{E_\ell} & \mathbf{0} \\ \delta_1 & \dots & \delta_k & d_1 & \dots & d_l & \mathbf{0}^m \end{pmatrix}.$$

Next, we show that $\pi_{M_1} V_{M_1}$ is the long-run probability vector of the transition matrix $\bar{U}_{M_1} L_1 P_t R_1 V_{M_1}$. By Corollary 8.2.11, applied for the uniform distributor \bar{U}_{M_1} , and the fact that π_{M_1} is the long-run probability vector of $L P_t R$ we have that

$$\pi_{M_1} V_{M_1} \bar{U}_{M_1} L_1 P_t R_1 V_{M_1} = \pi_{M_1} L_1 P_t R_1 V_{M_1} = \pi_{M_1} V_{M_1}.$$

Now, to prove that $\pi_{M_2} = \pi_{M_1} V_{M_1}$ it remains to show that $\bar{U}_{M_1} L_1 = L_2 \bar{U}_U$ and $R_1 V_{M_1} = V_U R_2$. This is obtained by direct multiplication of the matrices given above, which completes the proof that $\pi_{M_2} = \pi_{M_1} V_{M_1}$. Now, using this result we have that

$$R_{M_1}^\infty = \pi_{M_1} L_1 \rho = \pi_{M_1} V_{M_1} \bar{U}_{M_1} L_1 \rho = \pi_{M_1} V_{M_1} L_2 \bar{U}_U \rho = \pi_{M_2} L_2 \bar{U}_U \rho = R_{M_2}^\infty,$$

which completes the proof. ■

We illustrate the result by an example.

Example 8.2.17 The long-run probability vector π' of the translation by geometrization in Figure 8.3c is $\pi' = (\frac{2}{11} \frac{6}{11} \frac{3}{11})$. Its reward vector is $\rho' = (r_1 \ r_2 \ r_3)$, and so its long-run reward $R' = \frac{2}{11} r_1 + \frac{6}{11} r_2 + \frac{3}{11} r_3$ coincides with the R of the discrete-time Markov reward chain from Figure 8.1c. □

8.3 The Concurrent Alternating Bit Protocol

In this section we specify the concurrent alternating bit protocol both in the process theory TCP^{drst} and in the specification language χ . By restricting to deterministic timed delays, we show how to analytically obtain transient performance measures out of a χ -specification based on the proposal for long-run analysis in [96]. In the general case, we exploit discrete-event simulation in χ . For comparison, we perform Markovian analysis using an extension of the χ toolset by turning all delays into exponential ones with mean values equal to the duration of the timed delays.

The concurrent alternating bit protocol is used for communicating data along an unreliable channel with a guarantee that no information is lost. The protocol relies on retransmission of data. An overview of the concurrent alternating bit protocol is depicted in Figure 8.4.

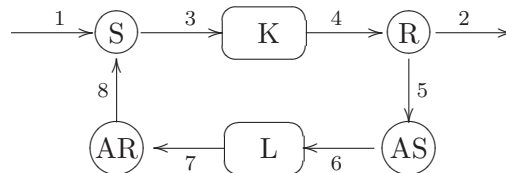


Figure 8.4: Scheme of the concurrent alternating bit protocol

The arrival process sends the data at port 1 to the sender process S . The sender adds an alternating bit to the data and sends the package to receiver R via the channel K using port 3. It keeps re-sending the same package with a fixed time-out, waiting for the correct acknowledgement that the data has been correctly received. The channel K has some probability of failure and it transfers the data with a generally distributed delay to the port 4. If the data is successfully received by R , then it is unpacked and the data is sent to the exit process via port 2. The alternating bit is sent as an acknowledgement back to the sender using the acknowledgement sender AS . The receiver R communicates with AS using port 5. The acknowledgement is sent via the unreliable channel L using port 6. Similarly to S the acknowledgement process re-sends data after a fixed time-out. The acknowledgement is communicated to the acknowledgement receiver process AR . If the received acknowledgement is the one expected, then AR informs the sender S that it can start with the transmission of the next data package.

We can specify the concurrent alternating bit protocol as below for a data set D . We note that the process theory does not contain an explicit probabilistic choice operator. To specify probabilistic behavior of the channel, we introduce time-outs to the channels K and L with duration t_k and t_ℓ , respectively, along the lines of Example 5.2.3. Thus, the messages are sent via the channels K and L before the time-out expires with a delay distributed according to the conditional random variables $\langle X \mid X < t_k \rangle$ and $\langle X \mid X < t_k \rangle$, respectively, or they get lost with probability $1 - F_X(t_k)$, and $1 - F_Y(t_\ell)$, respectively. We note that to eliminate the nondeterministic choice between s_4 and r_3 it must be that $P(X = t_k) = 0$ and $P(Y = t_\ell) = 0$. The concurrent alternating bit protocol is specified as $\theta_I(\partial_H(S \parallel K \parallel R \parallel AS \parallel L \parallel AR))$ with

$$\begin{aligned}
S &= S_0 \\
S_b &= \sum_{d \in D} \bar{r}_1(d). \sigma^{t_p}. \bar{s}_3(d, b). T_{d,b} \\
T_{d,b} &= \sigma^{t_s}. \bar{s}_3(d, b). T_{d,b} + \bar{r}_8(ack). S_{1-b} \\
K &= \sum_{e \in D \times \{0,1\}} \bar{r}_3(e). \theta_i([X]. \dot{i}. \bar{s}_4(e). K + \sigma^{t_k}. \dot{i}. K) \\
R &= R_0 \\
R_b &= \sum_{d \in D} \bar{r}_4(d, b). \sigma^{t_r}. \bar{s}_5(ack). \bar{s}_2(d). R_{1-b} + \sum_{d \in D} \bar{r}_4(d, 1-b). R_b \\
AS &= AS_1 \\
AS_b &= \bar{r}_5(ack). \bar{s}_6(1-b). AS_{1-b} + \sigma^{t_a}. \bar{s}_6(b). AS_b \\
L &= \sum_{b \in \{0,1\}} \bar{r}_5(b). \theta_i([Y]. \dot{i}. \bar{s}_6(b). L + \sigma^{t_\ell}. \dot{i}. L) \\
AR &= AR_0 \\
AR_b &= \bar{r}_7(b). \bar{s}_8(ack). AR_{1-b} + \bar{r}_7(1-b). AR_b,
\end{aligned}$$

where the recursion variables are parameterized by $d \in D$ and $b \in \{0, 1\}$,

$$\begin{aligned}
I &= \{r_1(d), r_2(d) \mid d \in D\} \cup \{c_3(d, b), c_4(d, b) \mid b \in \{0, 1\}, d \in D\} \cup \\
&\quad \{c_6(b), c_7(b) \mid b \in \{0, 1\}\} \cup \{c_5(ack), c_8(ack)\}, \text{ and} \\
H &= \{s_3(d, b), s_4(d, b), r_3(d, b), r_4(d, b) \mid b \in \{0, 1\}, d \in D\} \cup \\
&\quad \{r_6(b), r_7(b), s_6(b), s_7(b) \mid b \in \{0, 1\}\} \cup \\
&\quad \{r_5(ack), r_8(ack), s_5(ack), s_8(ack)\}.
\end{aligned}$$

The deterministic timed delays with duration t_p , t_s , t_k , t_r , t_a , and t_ℓ represent the processing time of the sender, the time-out of the sender, the time-out of the data channel, the processing time of the receiver, the time-out

of the acknowledgement sender, and the time-out of the acknowledgement channel. The internal action i enables the probabilistic choices induced by the time-outs as discussed in Example 5.2.3.

8.4 Specification and Analysis in χ

We illustrate some features of the language χ by presenting the χ specification of the sender process in Figure 8.5. Our example is based on the timed χ version as defined in [25].

```

sender ( c1, c3, c8: chan ) =
  | [ altbit: bool = false, data: nat, ack: bool
    , tp: nat = 1, ts: nat = 10
    | c1?data; delay tp; c3!<data,altbit>
    ; ( delay ts; c3!<data,altbit>
      | c8?ack; altbit := not altbit
      ; c1?data; delay tp; c3!<data,altbit>
      )*; deadlock
    ] |
  ] |

```

Figure 8.5: The sender process in χ

The process `sender` communicates with the other processes via three channels: `c1,c3,c8` (see Figure 8.4). The alternating bit is defined as a boolean variable and the data set is assumed to be the set of natural numbers. The sender waits for an arrival of a new data element, which it packs in `tp` time units. Afterwards, a frame with the data and the alternating bit is send via channel `c3`. Here, the process enters the iterative construct represented by `*(...)` and it either resubmits the data every `ts` time units or it waits for an acknowledgement at channel `c8` from the acknowledgement receiver process. If the acknowledgement is received before the time-out expires, the process flips the alternating bit, packs the new data in `tp` time units, and sends it again via channel `c3`. Note that in the example, the processing time `tp = 1` and the time-out `ts = 10` time units.

The standard semantics of (discrete-event) χ is in terms of timed transition systems [16, 11]. The main idea underlying the construction of a discrete-time probabilistic reward graph from a timed transition system, as proposed here, is to hide all actions, i.e., to rename them to the special internal action τ , and then use the concept of timed branching bisimulation [10, 94] to reduce the system while abstracting from its internal tran-

sitions. If there is no real nondeterminism in the model, a timed transition system without any action labeled transition is obtained, i.e., a discrete-time probabilistic reward graph without probabilistic transitions. If there is one or more nondeterministic transition left, then the system is underspecified. In that case, the resolution of the remaining nondeterministic choices depends on the environment, so its performance cannot be measured in the standard way.

Since χ has no features to model probabilistic choice, the random behavior of the data and acknowledgement channel is modeled in χ by a nondeterministic choice. When the corresponding discrete-time probabilistic reward graph is generated from the χ model these nondeterministic choices must be appropriately replaced by probabilistic ones. For this we slightly adjust the method described in the previous paragraph. Instead of hiding all actions, the special actions used to indicate probabilistic branching remain visible. After the minimization, the probabilities that were intentionally left out are put as labels on the nondeterministic transitions. Again, if there is still nondeterminism remaining in the model, we cannot proceed with performance analysis. Note that although the method is not always sound (in case of multiple probabilistic transitions from the same state) as it requests manipulation on the resulting graph, it serves its purpose for this and similar examples. Of course, another approach is to extend χ with an explicit probabilistic choice operator (e.g. the one in [50]). However, this requires drastic changes of the language and tools, and as such goes beyond the scope of this thesis. Notably, the framework makes use of probabilistic choices, but only for simulation purposes.

The χ language does not directly support reward specification either. We take a similar approach as for the absence of a probabilistic choice, and add rewards by manipulating the χ specification (again side-stepping changes in χ). We add, for each reward criterion, an ever repeating parallel component to the specification. The result is that in the timed transition system yielded, every state has a self-loop labeled by a special action denoting the reward rate of the state. These actions will not be hidden by branching bisimulation reduction and, therefore, persist in the resulting discrete-time probabilistic reward graph. As in the case for the probabilistic choice, a systematic technique rendering the above can in principle be incorporated into the χ environment.

The complete pipeline of generating discrete-time probabilistic reward graphs from χ specifications is illustrated in Figure 8.6. Currently, we employ scripts tweaked into the χ environment that insert probabilities and rewards, in order to automatically produce the desired discrete-time proba-

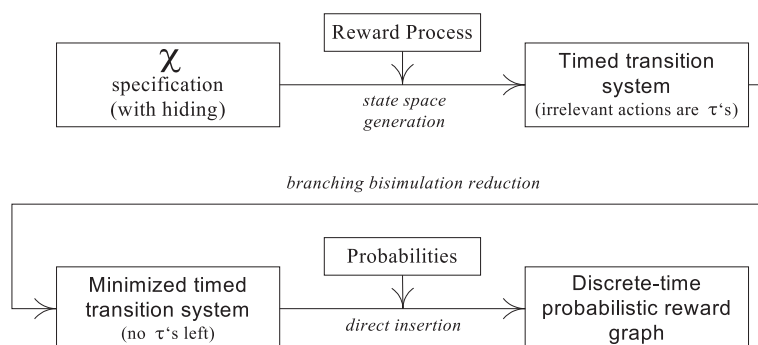


Figure 8.6: Generation of a discrete-time probabilistic reward graph from a χ specification

bilistic reward graph from a given χ specification.

If we assume that the distributions of the channels in the concurrent alternating bit protocol are deterministic, then we can obtain its discrete-time probabilistic reward graph representation and subsequently calculate its performance measures. First, we give in Fig. 8.7, the long-run utilization of the data channel K . We assume that $t_p = t_r = 1$, $t_s = t_a = 10$, $t_k = 6$, $t_\ell = 2$, that the distribution of the delay of the channel L is deterministic at 6, i.e., $P(X=6) = 1$, and that the distribution of the delay of the channel K is deterministic at 2, i.e., $P(Y=2) = 1$. To obtain the utilization of the data channel, we place reward 1 for every state in the unfolding of the timed delays with duration 6, which is the delay of the data channel K .

We note that, although the surface is smooth in the long-run analysis, if we observe the utilization at time step 200, we see that transient measure is not at all stable as depicted in Figure 8.8.

When the channels are generally distributed we resort to discrete-event simulation in χ for performance analysis. Figure 8.9 gives the utilization of the data channel K , when the distribution of the delay of the data channel is uniform between 2 and 10 and the distribution of the delay of the acknowledgement channel is uniform between 1 and 4. Thus, the uniform distributions of the data and the acknowledgement channels have the mean values of delay 6 and 2, respectively, as in the deterministic case.

For comparison, we also performed Markovian analysis, again by using discrete event simulation, and the result is depicted in Figure 8.10. The exponential delays were chosen of the same mean values as the corresponding delays in the deterministic case.

Finally, to give a flavor of the results, we show the dependence of the

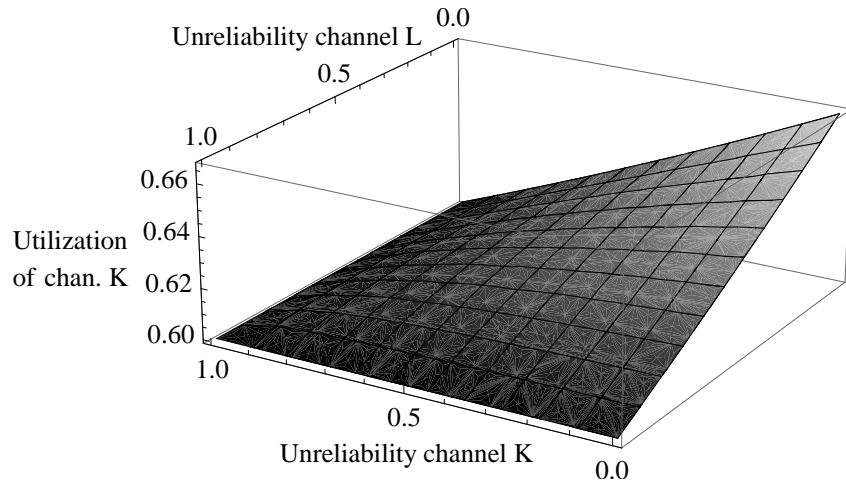


Figure 8.7: Long-run utilization of the data channel K

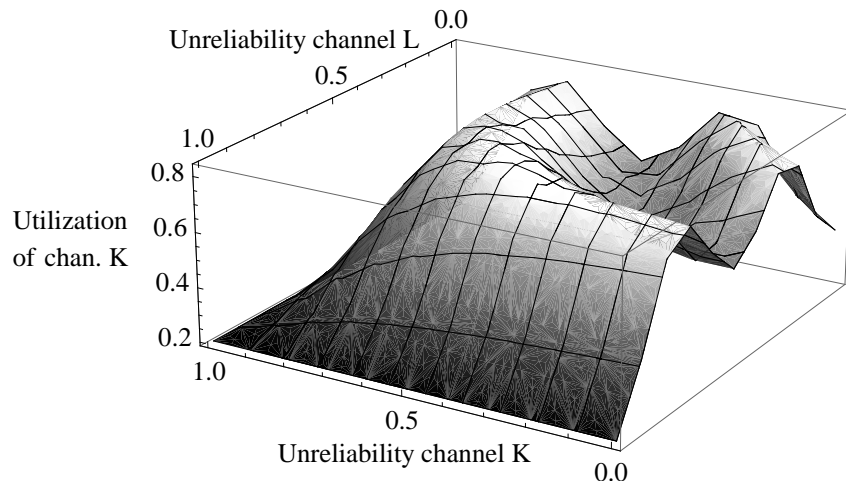


Figure 8.8: Utilization at time step 200 of the data channel K

utilization of the channel K on the unreliability of the channel K at time step 200 in Figure 8.11 for each approach. Here, the unreliability of the acknowledgement channel L is fixed to 0.5. One sees that the long-run analysis using discrete-time probabilistic reward graphs is close to the simulation re-

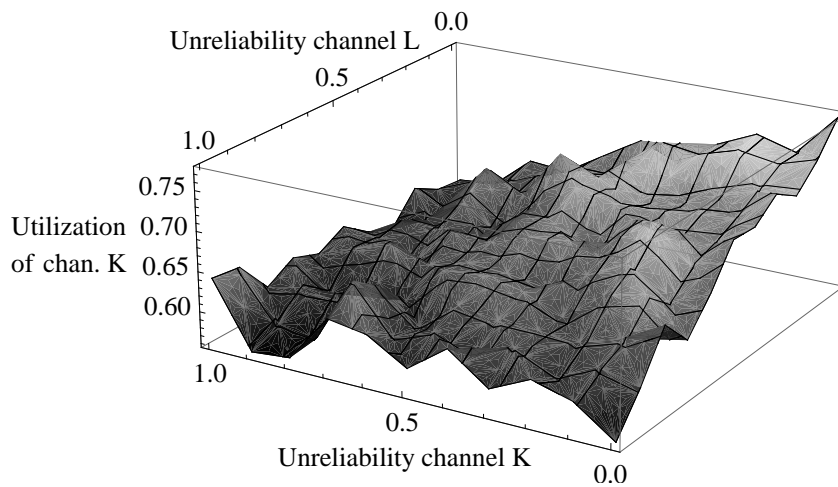


Figure 8.9: Utilization at 200 of channel K with uniformly distributed delays

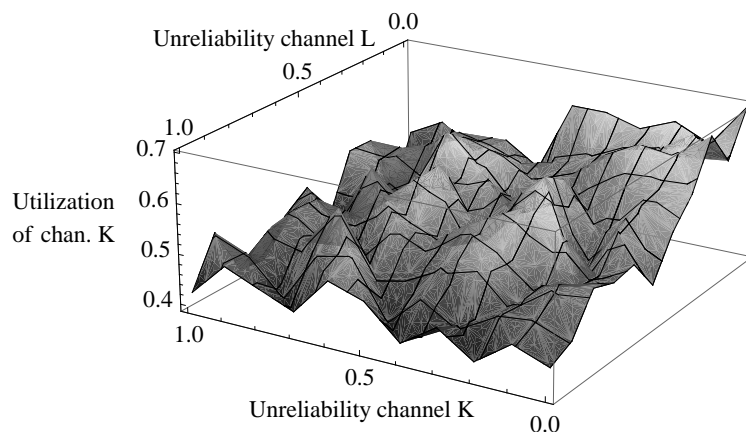


Figure 8.10: Utilization at 200 of channel K with exponentially distributed delays

sults for the uniformly distributed channels. This is expected because they have the same mean value. As noted in [96], the Markovian analysis always underestimates the performance because the expected value of the maximum

of two exponential delays is greater than maximum of the expected values of both delays, which increases the average cycle length of the system.

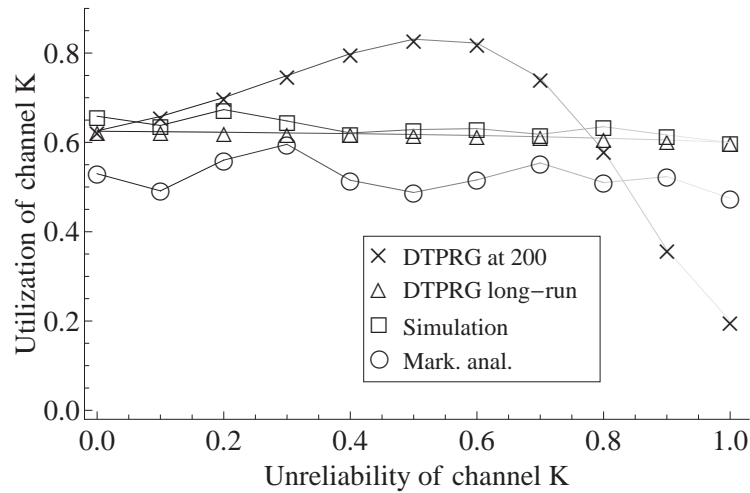


Figure 8.11: Utilization of the channel K at time 200 for unreliability 0.5 of the channel L

8.5 Summary

We introduce a mathematical model, called discrete-time probabilistic reward graphs, for the performance evaluation of systems featuring deterministic delay and probabilistic choice. We extend the χ -environment to a prototype that supports the new model, enabling an effective qualitative and quantitative analysis of probabilistic timed systems within the same framework. Then, we model the concurrent alternating bit protocol with lossy channels that were deterministically, exponentially, and uniformly distributed.

For long-run analysis, the results are close, although there are still some differences due to the low approximation of a deterministic delay with a single exponential distribution. However, the transient behavior of the protocol shows a substantially different behavior for differently distributed channels.

Chapter 9

Conclusions and Future Work

In the summary we answer the research questions posed in the introduction.

- What is the relationship between discrete real and generally-distributed stochastic time in process theories?

We attacked this problem from two angles. First, we developed a process algebra that comprises timed delays in a racing context, which are capable of breaking down the execution of stochastic delays in race condition semantics to unit timed delays. These delays provide explicit information about the expiration of the winning and the losing delays of the race. Then, we used them to derive discrete stochastic delays by means of recursive equations. In this way we can analyze the interaction between real and stochastic time per one time unit.

Afterwards, we developed a stochastic process algebra from scratch that follows the guidelines set by the timed setting. Here, we had to adjust standard timed delays to the new setting as the race condition does not comply with time additivity. We introduced context-sensitive interpolation, a new restricted notion of time additivity as its interpretation in the presence of the race condition.

The former approach to modeling stochastic time is more convenient from a theoretical point of view. However, the semantics of the stochastic delays requires infinite racing timed transition schemes. The latter approach manipulates finite objects, but every new feature, e.g., delayable action prefix or passage of time, has to be introduced as a separate construct. In return, the theory gets quite involved with introduction of new concepts.

- Is it possible, and if so, how, to (conservatively) extend timed process theories with stochastic time?

Again, we give two diametrical views on the matter. One way to conservatively extend timed process theories is to enrich the timed delays with probabilistic features in such a way that stochastic delays can be derived. Then, stochastic time is introduced in the theory as a derived concept, whereas the restrictions of the ‘probabilistic’ timed delays model the purely timed behavior. As discussed above, such an approach handles the execution of the stochastic delay per unit of time, leading to infinite transition systems.

When considering timed delays in a stochastic setting we have to investigate whether the fundamental properties of time can be preserved. This is a prerequisite for a conservative extension as the equations valid in the timed setting must also be valid in the stochastic counterpart. Here, we could not support time additivity and we had to resort to a more restricted notion of context-sensitive interpolation. This new concept allows interpolation of a timed delay only in the context of a compositional operator, like the alternative or parallel composition. When looking at strong bisimulation relations, we can come up with normal forms on which the two notions will coincide.

- Reversely, is it possible, and if so, how, to embed (discrete) real time in generally-distributed process theories?

The central notion in the stochastic setting was the race condition. We have introduced two notions dependent on the name of the delay. We embedded timed delays as independent Dirac stochastic delays. The dependence played an important role as the embedded timed delays must not be dependent on a stochastic delay, in which case they become generally-distributed as well. Again, there is no support for time additivity and we had to settle for context-sensitive interpolation.

- What is the effect of replacing timed delays by stochastic ones and what are the consequences of such a generalization?

A major consequence of replacing timed delays by stochastic ones is that the total order of the expiration of the delays is lost. This is because the standard race condition semantics models processes that race (compete) for a resource and, in general, every outcome of the race is possible.

To support this generalization we introduced the concept of partial races, i.e., the situation where the order of execution of the stochastic delays is imposed by the racing context. In combination with context-sensitive interpolation the modeler has all the support to safely extend timed specifications with stochastic time. Still, the process is not automatic as the decision must be made by the designer.

- Is it possible to show that the abstraction using the weak behavioral equivalence in Markovian process theories (and other modeling formalisms) is performance preserving and is such an approach compositional?

To treat intermediate performance models comprising exponential distributions, probabilistic choices, and nondeterministic (silent) transitions as stochastic processes we provide three extensions of Markov reward chains. For elimination of the fast and silent transitions we provide two aggregation methods that are used in different settings. Remarkably, if all fast and silent transitions are eliminated, both aggregations produce equivalent processes. We also show compositionality with respect to the parallel composition and the preorders induced by the aggregation methods.

- Can we do performance analysis using discrete-time delays and probabilistic choices?

Such models naturally translate to discrete-time Markov reward chains, which can then be analyzed using standard techniques. Here, we provide means to also reflect the performance measures back to the original process. For long-run analysis we develop a translation that does not increase the number of states, as the original translation transforms a timed delay into a series of transitions with probability one on the side of the discrete-time Markov reward chains.

As future work, it is interesting to introduce the hiding operator that produces internal transitions and to develop a notion of branching or weak bisimulation in that setting. This should pave the way for bigger case studies in Internet protocol verification and analysis as detailed performance specification is viable by using both generally distributed stochastic delays and standard timeouts. We can also exploit existing real-time specification as the theory is sufficiently flexible to allow extension of real-time with stochastic time, while retaining any imposed ordering of the original delays.

In the Markovian domain, further work can focus on the analysis of models that combine stochastic transitions and (non-internal) action labeled transitions, so that in addition to interleaving, synchronization can also be expressed.

Finally, a full-blown extension of the χ language to fully support the developed theory is viable, relieving the script-based short-cuts taken presently to intervene in the tool environment. We foresee that this can be achieved by introducing a probabilistic choice operator, and by facilitating the assignment of rewards in the toolset.

Bibliography

- [1] L. Aceto. Some of my favourite results in classic process algebra. *Bulletin of the EATCS*, 81:90–108, 2003.
- [2] R. P. Agaev and P. Y. Chebotarev. On determining the eigenprojection and components of a matrix. *Automated Remote Control*, 63:1537–1545, 2002.
- [3] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. Wiley, 1995.
- [4] M. Ajmone Marsan, A. Bianco, L. Ciminiera, R. Sisto, and A. Valenzano. A LOTOS extension for the performance analysis of distributed systems. *IEEE/ACM Transactions on Networking*, 2(2):151–165, 1994.
- [5] H. H. Ammar, Y. F. Huang, and R. W. Liu. Hierarchical models for systems reliability, maintainability, and availability. *IEEE Transactions on Circuits and Systems*, 34(6):629–638, 1987.
- [6] N. W. A. Arends. *A systems engineering specification formalism*. PhD thesis, Eindhoven University of Technology, 1996.
- [7] J. C. M. Baeten. A brief history of process algebra. *Theoretical Computer Science*, 335:131 – 146, 2005.
- [8] J. C. M. Baeten, T. Basten, and M. A. Reniers. *Process Algebra: Equational Theories of Communicating Processes*. Cambridge University Press, 2009. To appear.
- [9] J. C. M. Baeten, J. A. Bergstra, and J. W. Klop. On the consistency of Koomen’s fair abstraction rule. *Theoretical Computer Science*, 51(1):129–176, 1987.
- [10] J. C. M. Baeten, J. A. Bergstra, and M. A. Reniers. Discrete time process algebra with silent step. In *Proof, language, and interaction*:

- essays in honour of Robin Milner*, pages 535–569. MIT Press, Cambridge, MA, USA, 2000.
- [11] J. C. M. Baeten and C. A. Middelburg. *Process Algebra with Timing*. Monographs in Theoretical Computer Science. Springer, 2002.
 - [12] J. C. M. Baeten and M. A. Reniers. Timed process algebra (with a focus on explicit termination and relative timing). In *Proceedings of SFM 2004*, volume 3185 of *Lecture Notes of Computer Science*, pages 59–97. Springer, 2004.
 - [13] J. C. M. Baeten and W.P. Weijland. *Process Algebra*. Number 18 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.
 - [14] C. Baier, B. R. Haverkort, H. Hermanns, and J.-P. Katoen. Model checking meets performance evaluation. *SIGMETRICS Performance Evaluation Review*, 32(4):10–15, 2005.
 - [15] J. Banks, J. S. Carson II, B. L. Nelson, and D. M. Nicol. *Discrete-event system simulation*. Prentice Hall, 2000.
 - [16] D. A. van Beek, K. L. Man, M. A. Reniers, J. E. Rooda, and R. R. H. Schiffelers. Syntax and consistent equation semantics of hybrid Chi. *Journal of Logic and Algebraic Programming*, 68:129–210, 2006.
 - [17] D. A. van Beek, A. van der Ham, and J. E. Rooda. Modelling and control of process industry batch production systems. In *15th Triennial World Congress of the International Federation of Automatic Control*, Barcelona, Spain, 2002.
 - [18] J. A. Bergstra. On the design rationale of ACP style process algebras. *Electronic Notes in Theoretical Computer Science*, 162:79–85, 2006.
 - [19] J. A. Bergstra, A. Ponse, and Scott A. Smolka, editors. *Handbook of Process Algebra*. Elsevier, 2001.
 - [20] M. Bernardo and R. Gorrieri. A tutorial on EMPA: A theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theoretical Computer Science*, 202(1–2):1–54, 1998.
 - [21] S. Blom, W. Fokkink, J. F. Groote, I. van Langevelde, B. Lissner, and J. C. van de Pol. μ CRL: A toolset for analysing algebraic specifications. In *Proceedings of CAV 2001*, volume 2102 of *Lecture Notes in Computer Science*, pages 250–254, 2001.

- [22] H. C. Bohnenkamp, P. R. D'Argenio, H. Hermanns, and J.-P. Katoen. MODEST: A compositional modeling formalism for hard and softly timed systems. *IEEE Transactions on Software Engineering*, 32:812–830, 2006.
- [23] E. Bortnik, N. Trčka, A. J. Wijs, S. P. Luttik, J. M. van de Mortel-Fronczak, J. C. M. Baeten, W. J. Fokkink, and J. E. Rooda. Analyzing a χ model of a turntable system using Spin, CADP and UPPAAL. *Journal of Logic and Algebraic Programming*, 65:51–104, 2005.
- [24] E. M. Bortnik, D. A. van Beek, J. M. van de Mortel-Fronczak, and J. E. Rooda. Verification of timed Chi models using UPPAAL. In *Proceedings of ICINCO'05*, pages 486–492, Barcelona, 2005.
- [25] V. Bos and J. J. T. Kleijn. *Formal Specification and Analysis of Industrial Systems*. PhD thesis, Eindhoven University of Technology, 2002.
- [26] M. Bravetti. *Specification and Analysis of Stochastic Real-time Systems*. PhD thesis, Università di Bologna, 2002.
- [27] M. Bravetti, M. Bernardo, and R. Gorrieri. From EMPA to GSMPA: Allowing for general distributions. In *Proceedings of PAPM'97*, pages 17–33, Enschede, 1997.
- [28] M. Bravetti and P. R. D'Argenio. Tutte le algebre insieme – concepts, discussions and relations of stochastic process algebras with general distributions. In *Validation of Stochastic Systems*, pages 44–88. Lecture Notes of Computer Science 2925, 2004.
- [29] J. Bryans, H. Bowman, and J. Derrick. Model checking stochastic automata. *ACM Transactions on Computational Logic*, 4(4):452–492, 2003.
- [30] P. Buchholz. Exact and ordinary lumpability in finite Markov chains. *Journal of Applied Probability*, 31:59–75, 1994.
- [31] P. Buchholz. Markovian process algebra: composition and equivalence. In *Proceedings of PAPM 94*, pages 11–30, Erlangen, Germany, 1994. Universität Erlangen-Nürnberg.
- [32] P. Buchholz. Structured analysis techniques for large Markov chains. In *Proceedings of SMCTools 2006*, volume 201 of *ACM International Conference Proceedings Series*, pages 2–10, Pisa, Italy, 2006.

- [33] P. Buchholz and P. Kemper. Kronecker based matrix representations for large Markov chains. In *Validation of Stochastic Systems*, volume 2925 of *Lecture Notes in Computer Science*, pages 256–295, 2004.
- [34] E. J. J. van Campen. *Design of a Multi-Process Multi-Product Wafer Fab*. PhD thesis, Eindhoven University of Technology, 2000.
- [35] S. Cattani, R. Segala, M. Kwiatkowska, and G. Norman. Stochastic transition systems for continuous state spaces and non-determinism. In *Proceedings of FoSSaCS'05*, volume 3441, pages 125–139. Lecture Notes of Computer Science, 2005.
- [36] L. Cheung, N. Lynch, R. Segala, and F. Vaandrager. Switched PIOA: Parallel composition via distributed scheduling. *Theoretical Computer Science*, 365(1-2):83–108, 2006.
- [37] K. L. Chung. *Markov Chains with Stationary Probabilities*. Springer, 1967.
- [38] G. Ciardo, J. Muppala, and K. S. Trivedi. On the solution of GSPN reward models. *Performance Evaluation*, 12:237–253, 1991.
- [39] M. Coderch, A. S. Willsky, S. S. Sastry, and D. A. Castanon. Hierarchical aggregation of singularly perturbed finite state Markov processes. *Stochastics*, 8:259–289, 1983.
- [40] P. R. D’Argenio. From stochastic automata to timed automata: Abstracting probability in a compositional manner. In *Proceedings of WAIT 2003*, Buenos Aires, 2003.
- [41] P. R. D’Argenio and J.-P. Katoen. A theory of stochastic systems, part I: Stochastic automata. *Information and Computation*, 203(1):1–38, 2005.
- [42] P. R. D’Argenio and J.-P. Katoen. A theory of stochastic systems, part II: Process algebra. *Information and Computation*, 203(1):39–74, 2005.
- [43] F. Delebecque. A reduction process for perturbed Markov chains. *SIAM Journal of Applied Mathematics*, 2:325–330, 1983.
- [44] F. Delebecque and J. P. Quadrat. Optimal control of Markov chains admitting strong and weak interactions. *Automatica*, 17:281–296, 1981.

- [45] W. Doeblin. Sur l'équation matricielle $A(t + s) = A(t) \cdot A(s)$ et ses applications aux probabilités en chaîne. *Bulletin des Sciences Mathématiques*, 62:21–32, 1938.
- [46] J. L. Doob. *Stochastic Processes*. Wiley, 1953.
- [47] J. J. H. Fey. *Design of a Fruit Juice Blending and Packaging Plant*. PhD thesis, Eindhoven University of Technology, 2000.
- [48] P. W. Glynn. A GSMP formalism for discrete event systems. *Proceedings of the IEEE*, 77(1):14–23, 1989.
- [49] A. Graham. *Kronecker Products and Matrix Calculus With Applications*. Ellis Horwood, 1981.
- [50] H. A. Hansson. *Time and Probability in Formal Design of Distributed Systems*. Elsevier, 1994.
- [51] H. Hermanns. *Interactive Markov Chains: The Quest for Quantified Quality*, volume 2428 of *Lecture Notes in Computer Science*. Springer, 2002.
- [52] H. Hermanns, V. Mertsiotakis, and M. Rettelbach. Performance analysis of distributed systems using TIPP. In *Proceedings of UKPEW'94*, pages 131–144. University of Edinburgh, 1994.
- [53] E. Hille and R. S. Phillips. *Functional Analysis and Semi-Groups*. AMS, 1957.
- [54] J. Hillston. The nature of synchronisation. In *Proceedings of PAPM '94*, pages 51–70, Erlangen, Germany, 1994.
- [55] J. Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.
- [56] G. J. Holzmann. The model checker SPIN. *IEEE Transactions on Software Engineering*, 23(5):279–295, 1997. Special issue on Formal Methods in Software Practice.
- [57] R. A. Howard. *Dynamic Probabilistic Systems*. Wiley, 1971.
- [58] A. Jensen. Markoff chains as an aid in the study of Markoff processes. *Skandinavisk Aktuarietidskrift*, 36:87–91, 1953.

- [59] B. Jonsson, Yi Wang, and K.G. Larsen. Probabilistic extensions of process algebras. In [19], pages 685–710.
- [60] J. P. Katoen and P. R. D’Argenio. General distributions in process algebra. In *Lectures on formal methods and performance analysis*, Lecture Notes in Computer Science, pages 375–429. 2001.
- [61] J. G. Kemeny and J. L. Snell. *Finite Markov chains*. Springer, 1976.
- [62] M. Kwiatkowska, G. Norman, and D. Parker. PRISM: Probabilistic symbolic model checker. In *Proceedings of TOOLS 2002*, pages 200–204. Springer, 2002.
- [63] K. G. Larsen, P. Pettersson, and W. Yi. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer*, 1:134–152, 1997.
- [64] N. López and M. Núñez. NMSPA: A non-Markovian model for stochastic processes. In *Proceedings of ICDS 2000*, pages 33–40. IEEE, 2000.
- [65] N. López and M. Núñez. Weak stochastic bisimulation for non-Markovian processes. In *Proceedings of ICTAC’05*, volume 3722 of *Lecture Notes of Computer Science*, pages 454–468. Springer, 2005.
- [66] J. Markovski and E. P. de Vink. Embedding real-time in stochastic process algebras. In *Proceedings of EPEW 2006*, volume 4054, pages 47–62. Lecture Notes of Computer Science, 2006.
- [67] J. Markovski and E. P. de Vink. Embedding real time in stochastic process algebras. Technical Report CS 06/15, Eindhoven University of Technology, 2006.
- [68] J. Markovski and E. P. de Vink. Real-time in stochastic process algebra: Keeping track of winners and losers. Technical Report CS 07/13, Eindhoven University of Technology, 2007.
- [69] J. Markovski and E. P. de Vink. Real-time process algebra with stochastic delays. In *Proceedings of ACSD 2007*, pages 177–186. IEEE, 2007.
- [70] J. Markovski and E. P. de Vink. Discrete real-time and stochastic-time process algebra for performance analysis of distributed systems. Technical Report CS 08/10, Eindhoven University of Technology, 2008.

- [71] J. Markovski and E.P. de Vink. Discrete real-time and stochastic-time process algebra for performance analysis of distributed systems. In *Proceedings of ACSD'08*. IEEE, 2008. To appear.
- [72] J. Markovski and E.P. de Vink. Extending timed process algebra with discrete stochastic time. In *Proceedings of AMAST'08*. Lecture Notes of Computer Science, 2008. To appear.
- [73] J. Markovski, A. Sokolova, N. Trčka, and E. P. de Vink. Compositionality for Markov chains with fast transitions. Technical Report CS 07/17, Eindhoven University of Technology, 2007.
- [74] J. Markovski, A. Sokolova, N. Trčka, and E. P. de Vink. Compositionality for Markov reward chains with fast transitions. In *Proceedings of EPEW'07*, volume 4748 of *Lecture Notes of Computer Science*, pages 18–32, 2007.
- [75] J. Markovski and N. Trčka. Lumping Markov chains with silent steps. In *Proceedings of QEST'06*, pages 221–230, Riverside, CA, USA, 2006. IEEE Computer Society.
- [76] J. Markovski and N. Trčka. Lumping Markov chains with silent steps. Technical Report CS 06/13, Eindhoven University of Technology, 2006.
- [77] J. Markovski and N. Trčka. Aggregation methods for Markov reward chains with fast and silent transitions. Technical Report CS 07/08, Eindhoven University of Technology, 2007.
- [78] J. Markovski and N. Trčka. Aggregation methods for Markov reward chains with fast and silent transitions. In *Proceedings of MMB2008: Measurement, Modeling and Evaluation of Computer and Communication Systems*, pages 93–108. VDE Verlag, 2008.
- [79] D. Miller and A. Tiu. A proof theory for generic judgments. *ACM Transaction on Computational Logic*, 6(4):749–783, 2005.
- [80] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980.
- [81] U. Montanari and M. Pistore. *Formal Methods for Mobile Computing*, volume 3465 of *Lecture Notes in Computer Science*, chapter History-Dependent Automata: An Introduction, pages 1–28. Springer Berlin / Heidelberg, 2005.

- [82] M. F. Neuts. *Matrix-geometric solutions in stochastic models, an algorithmic approach*. John Hopkins University Press, 1981.
- [83] V. Nicola. Lumping in Markov reward processes. IBM Research Report RC 14719, IBM, 1989.
- [84] X. Nicollin and J. Sifakis. An overview and synthesis of timed process algebras. In *Real-Time: Theory in Practice*, volume 600 of *Lecture Notes of Computer Science*, pages 526–548. Springer, 1992.
- [85] W. van Niftrik. Context-sensitive interpolation. Master thesis, Eindhoven University of Technology, 2008.
- [86] B. Plateau and K. Atif. Stochastic automata network of modeling parallel systems. *IEEE Transactions on Software Engineering*, 17(10):1093–1108, 1991.
- [87] C. Priami. Stochastic π -calculus with general distributions. In *Proceedings of PAPM'96*, pages 41–57, Torino, 1996.
- [88] R. R. H. Schiffelers and K. L. Man. *Formal Specification and Analysis of Hybrid Systems*. PhD thesis, Eindhoven University of Technology, 2006.
- [89] A. Sokolova and E. P. de Vink. On relational properties of lumpability. In *Proceedings of 4th PROGRESS symposium on Embedded Systems*, Utrecht, The Netherlands, 2003.
- [90] J. Sproston. *Validation of Stochastic Systems*, volume 2925 of *Lecture Notes of Computer Science*, chapter Model Checking for Probabilistic Timed Systems, pages 189–229. Springer, 2004.
- [91] W. J. Stewart. *Introduction to the numerical solution of Markov chains*. Princeton University Press, New Jersey, USA, 1994.
- [92] A. T. Tai, K. S. Tso, and W. H. Sanders. A recurrence-relation-based reward model for performability evaluation of embedded systems. In *Proceedings of DSN'08*. IEEE, 2008.
- [93] N. Trčka. Verifying χ models of industrial systems in Spin. In *Proceedings of ICFEM 2006*, volume 4260 of *Lecture Notes in Computer Science*, pages 132–148. Springer, 2006.
- [94] N. Trčka. *Silent Steps in Transition Systems and Markov Chains*. PhD thesis, Eindhoven University of Technology, 2007.

- [95] N. Trčka, S. Georgievska, J. Markovski, S. Andova, and E. P. de Vink. Performance analysis of χ models using discrete-time probabilistic reward graphs. In *Proceedings of WODES'08*. IEEE, 2008. To appear.
- [96] N. Trčka, S. Georgievska, J. Markovski, S. Andova, and E. P. de Vink. Performance analysis of χ models using discrete-time probabilistic reward graphs. Technical Report CS 08/02, Eindhoven University of Technology, 2008.
- [97] A. Wijs. From χ_t to μ CRL: Combining performance and functional analysis. In *Proceedings of ICECCS'05*, pages 184–193, Washington, DC, USA, 2005. IEEE Computer Society.
- [98] S.-H. Wu, S. A. Smolka, and E. Stark. Composition and behaviors of probabilistic I/O automata. *Theoretical Computer Science*, 176(1–2):1–38, 1997.
- [99] W. Yi. CCS + time = an interleaving model for real-time systems. In *Proceedings of ICALP'91*, volume 510 of *Lecture Notes of Computer Science*, pages 217–228. Springer, 1991.

Curriculum Vitae

Jasen Markovski was born on the 18th of May 1978 in Skopje, Macedonia (former Yugoslavia). He studied computer science at the Institute of Informatics, Faculty of Natural Sciences and Mathematics, University of Skopje, Macedonia, and obtained the degree of Graduated Engineer in Informatics in October 2001. In May 2004 he obtained a M.Sc. degree in Informatics from the same institution. In August 2004 he became a Ph.D. student at the Formal Methods Group, Department of Mathematics and Computer Science, Eindhoven University of Technology, The Netherlands.

Titles in the IPA Dissertation Series since 2002

M.C. van Wezel. *Neural Networks for Intelligent Data Analysis: theoretical and experimental aspects.* Faculty of Mathematics and Natural Sciences, UL. 2002-01

V. Bos and J.J.T. Kleijn. *Formal Specification and Analysis of Industrial Systems.* Faculty of Mathematics and Computer Science and Faculty of Mechanical Engineering, TU/e. 2002-02

T. Kuipers. *Techniques for Understanding Legacy Software Systems.* Faculty of Natural Sciences, Mathematics and Computer Science, UvA. 2002-03

S.P. Luttik. *Choice Quantification in Process Algebra.* Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2002-04

R.J. Willemen. *School Timetable Construction: Algorithms and Complexity.* Faculty of Mathematics and Computer Science, TU/e. 2002-05

M.I.A. Stoelinga. *Alea Jacta Est: Verification of Probabilistic, Real-time and Parametric Systems.* Faculty of Science, Mathematics and Computer Science, KUN. 2002-06

N. van Vugt. *Models of Molecular Computing.* Faculty of Mathematics and Natural Sciences, UL. 2002-07

A. Fehnker. *Citius, Vilius, Melius: Guiding and Cost-Optimality in Model Checking of Timed and Hybrid Systems.* Faculty of Science, Mathematics and Computer Science, KUN. 2002-08

R. van Stee. *On-line Scheduling and Bin Packing.* Faculty of Mathematics and Natural Sciences, UL. 2002-09

D. Tauritz. *Adaptive Information Filtering: Concepts and Algorithms.* Faculty of Mathematics and Natural Sciences, UL. 2002-10

M.B. van der Zwaag. *Models and Logics for Process Algebra.* Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2002-11

J.I. den Hartog. *Probabilistic Extensions of Semantical Models.* Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2002-12

L. Moonen. *Exploring Software Systems.* Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2002-13

J.I. van Hemert. *Applying Evolutionary Computation to Constraint Satisfaction and Data Mining.* Faculty of Mathematics and Natural Sciences, UL. 2002-14

S. Andova. *Probabilistic Process Algebra.* Faculty of Mathematics and Computer Science, TU/e. 2002-15

Y.S. Usenko. *Linearization in μ CRL.* Faculty of Mathematics and Computer Science, TU/e. 2002-16

J.J.D. Aerts. *Random Redundant Storage for Video on Demand.* Faculty of Mathematics and Computer Science, TU/e. 2003-01

M. de Jonge. *To Reuse or To Be Reused: Techniques for component composition and construction.* Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2003-02

J.M.W. Visser. *Generic Traversal over Typed Source Code Representations.* Faculty of Natural Sciences, Mathematics, and Computer Science, UvA. 2003-03

S.M. Bohte. *Spiking Neural Networks.* Faculty of Mathematics and Natural Sciences, UL. 2003-04

T.A.C. Willemse. *Semantics and Verification in Process Algebras with Data and Timing.* Faculty of Mathematics and Computer Science, TU/e. 2003-05

- S.V. Nedeia.** *Analysis and Simulations of Catalytic Reactions.* Faculty of Mathematics and Computer Science, TU/e. 2003-06
- M.E.M. Lijding.** *Real-time Scheduling of Tertiary Storage.* Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2003-07
- H.P. Benz.** *Casual Multimedia Process Annotation – CoMPAs.* Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2003-08
- D. Distefano.** *On Modelchecking the Dynamics of Object-based Software: a Foundational Approach.* Faculty of Electrical Engineering, Mathematics & Computer Science, UT. 2003-09
- M.H. ter Beek.** *Team Automata – A Formal Approach to the Modeling of Collaboration Between System Components.* Faculty of Mathematics and Natural Sciences, UL. 2003-10
- D.J.P. Leijen.** *The λ Abroad – A Functional Approach to Software Components.* Faculty of Mathematics and Computer Science, UU. 2003-11
- W.P.A.J. Michiels.** *Performance Ratios for the Differencing Method.* Faculty of Mathematics and Computer Science, TU/e. 2004-01
- G.I. Jojgov.** *Incomplete Proofs and Terms and Their Use in Interactive Theorem Proving.* Faculty of Mathematics and Computer Science, TU/e. 2004-02
- P. Frisco.** *Theory of Molecular Computing – Splicing and Membrane systems.* Faculty of Mathematics and Natural Sciences, UL. 2004-03
- S. Maneth.** *Models of Tree Translation.* Faculty of Mathematics and Natural Sciences, UL. 2004-04
- Y. Qian.** *Data Synchronization and Browsing for Home Environments.* Faculty of Mathematics and Computer Science and Faculty of Industrial Design, TU/e. 2004-05
- F. Bartels.** *On Generalised Coinduction and Probabilistic Specification Formats.* Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2004-06
- L. Cruz-Filipe.** *Constructive Real Analysis: a Type-Theoretical Formalization and Applications.* Faculty of Science, Mathematics and Computer Science, KUN. 2004-07
- E.H. Gerding.** *Autonomous Agents in Bargaining Games: An Evolutionary Investigation of Fundamentals, Strategies, and Business Applications.* Faculty of Technology Management, TU/e. 2004-08
- N. Goga.** *Control and Selection Techniques for the Automated Testing of Reactive Systems.* Faculty of Mathematics and Computer Science, TU/e. 2004-09
- M. Niqui.** *Formalising Exact Arithmetic: Representations, Algorithms and Proofs.* Faculty of Science, Mathematics and Computer Science, RU. 2004-10
- A. Löb.** *Exploring Generic Haskell.* Faculty of Mathematics and Computer Science, UU. 2004-11
- I.C.M. Flinsenberg.** *Route Planning Algorithms for Car Navigation.* Faculty of Mathematics and Computer Science, TU/e. 2004-12
- R.J. Bril.** *Real-time Scheduling for Media Processing Using Conditionally Guaranteed Budgets.* Faculty of Mathematics and Computer Science, TU/e. 2004-13
- J. Pang.** *Formal Verification of Distributed Systems.* Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2004-14
- F. Alkemade.** *Evolutionary Agent-Based Economics.* Faculty of Technology Management, TU/e. 2004-15
- E.O. Dijk.** *Indoor Ultrasonic Position Estimation Using a Single Base Station.*

Faculty of Mathematics and Computer Science, TU/e. 2004-16

S.M. Orzan. *On Distributed Verification and Verified Distribution.* Faculty of Sciences, Division of Mathematics and Computer Science, VUA. 2004-17

M.M. Schrage. *Proxima - A Presentation-oriented Editor for Structured Documents.* Faculty of Mathematics and Computer Science, UU. 2004-18

E. Eskenazi and A. Fyukov. *Quantitative Prediction of Quality Attributes for Component-Based Software Architectures.* Faculty of Mathematics and Computer Science, TU/e. 2004-19

P.J.L. Cuijpers. *Hybrid Process Algebra.* Faculty of Mathematics and Computer Science, TU/e. 2004-20

N.J.M. van den Nieuwelaar. *Supervisory Machine Control by Predictive-Reactive Scheduling.* Faculty of Mechanical Engineering, TU/e. 2004-21

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 π -Calculus Processes with*

- Replication*. Faculty of Mathematics and Natural Sciences, UL. 2005-17
- P. Zoetewij**. *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 Multidisciplinary 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. Calam.** *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