

Discrete-Time Process Algebra

Discrete-Time Process Algebra

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
Rector Magnificus, prof.dr. M. Rem, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op donderdag 18 december 1997 om 14.00 uur

door

Jan Joris Vereijken

geboren te Vlaardingen



Dit proefschrift is goedgekeurd door de promotoren:

prof. dr. J. C. M. Baeten
prof. dr. J. A. Bergstra

Copromotor:

dr. S. Mauw

CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Vereijken, Jan Joris

Discrete-time process algebra / Jan Joris Vereijken. - Eindhoven :
Eindhoven University of Technology, 1997. - xiii, 328 p.

Proefschrift. - ISBN 90-386-0541-2

NUGI 852

Subject headings: process algebra / programming languages ;
real-time / software description ; program verification

CR Subject Classification (1991): F.3.2, F.1.2, D.2.4, D.1.3

IPA Dissertation Series 1997-06.

Copyright ©1997 by Jan Joris Vereijken, Amsterdam.

Typeset by \LaTeX 2 ϵ in Lucida® Bright.

Printed by PrintPartners Ipskamp, Enschede.



This thesis has been supported by the Netherlands Computer Science Research Foundation (SION) with financial support from the Netherlands Organization for Scientific Research (NWO). It has been carried out under the auspices of the Institute for Programming Research and Algorithmics (IPA).

dedicated to Tieleke

Preface

When in the spring of 1993 I was just about finished with my Master's degree studies at Leiden University, I found myself in need of a job. I looked for positions both inside and outside of academia, even applied to a few, but none of them felt right. Some were very theoretical in nature, and I feared taking them would cut me off from applied computer science forever—this is not a field where you can afford to get out of touch with recent developments. The ones that were of a practical nature, however, all precluded the possibility to do any research, and at the time I was very eager to write a dissertation.

Time was running out, unfortunately, as delaying my choice would invariably mean getting conscripted for military service in the Royal Dutch Army, which at the time was compulsory. That threat clearly presented the worst of both worlds: the assurance of both falling behind in practical matters *and* not being able to do any fundamental research. So, I reluctantly decided to accept one of the offers I had had *quickly*, even if that meant, heaven forbid, rolling a dice to make up my mind.

Right that same week, I heard Jos Baeten was looking for an *onderzoeker in opleiding* to do research on a project called *Real-Time and Real-Space Process Algebra*. Although Jos's name was known to me, I had not the slightest idea what this project might be about. The project description spoke about orbiting satellites and real-time algorithms, so I supposed it had something to do with real-time operating systems or the like. Cautiously, because I had also expressed my interest in working for him, I went to see Joost Engelfriet (whose advice I trust blindly) and asked him what process algebra was all about. Joost explained to me it was a new field, very much *en vogue*, that combined aspects of theoretical computer science with their practical application to real-life problems. "Furthermore," he said, "Jos Baeten is a great fund-raiser, and he attracts many good people to work for him." My mind was made up; I went to see Jos, and to my amazement within a week I got offered a position.

One of the first things Jos asked me to do, was to take a look at an algorithm called *Fischer's Protocol*. I studied it, tried to analyze it using *discrete-time process algebra* and *real-time process algebra*, and wrote a paper about it [191], parts of which appear as Chapter 7 of this thesis.

Apparently Jos liked it, because immediately afterward he suggested a far more ambitious project: I was to apply process algebra to *Hybrid Systems*. This was a mistake. Everything I wrote down seemed to run wild in all possible directions. Although I did finish an extended abstract [192], the paper itself was a mess, stayed a mess, and eventually I just gave up on it. None of that work appears in this thesis. Nevertheless, I think it is the most important paper I have (not) written, as it put me in shockingly clear perspective of my own limitations.

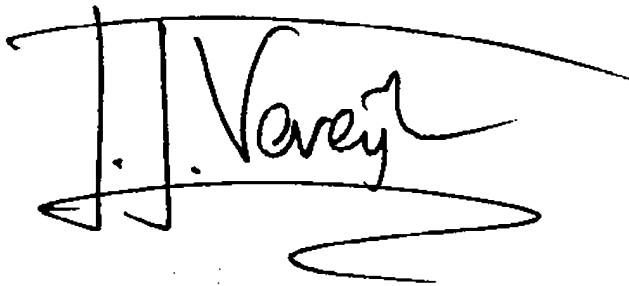
Next, I decided to do something completely different, preferably something less adventurous. Jos at that moment had just finished a paper on *discrete-time process algebra*,

and buried deep inside it, there was a theorem without a proof. He suggested Michel Reniers (then my roommate) and I see if we could supply that proof. It should take hardly three weeks, he estimated, and pose no surprises. As it turned out, the theorem was incorrect, and correcting all problems took over a year, resulting in a 100+ page proof. This was published as [180], and parts of it appear as Chapters 4 and 5 of this thesis.

Having spent so much time on *discrete-time process algebra* (which was completely out of the scope of the original project description!), I had grown to like it. Could I maybe write another paper on it? That could be arranged; Jos had been thinking about the incorporation of the *empty process* in *discrete-time process algebra*, and although he had very clear ideas on it, he had never found the time to write it all down. He explained to me the basic concepts, and I set out to turn them into solid process algebra. That work resulted in another paper [36], parts of which appear as Chapter 6 of this thesis.

Finally, it was Jan Bergstra who recognized that what I had been doing over the past four years could be clearly divided in two parts. First, there were the things on *discrete-time process algebra*, and they were quite nice. Secondly, there was the other stuff which was, frankly, trash. When I first heard it put that way (and of course Jan did not put it so bluntly, but used wonderful euphemisms instead), I was both shocked and outraged. I tried to defend myself, could not, and consequently did not sleep for a week. Ultimately, I admitted to myself that Jan was right.

My dissertation should be called *Discrete-Time Process Algebra*.

A handwritten signature in black ink, appearing to read 'J. J. Vereijken'. The signature is stylized with a large, sweeping horizontal stroke above the letters and a long, horizontal flourish below the name.

Jan Joris Vereijken,

Amsterdam, October 3, 1997.

Contents

Preface	vii
Contents	ix
1 Introduction	1
1.1 Prologue	1
1.2 Formal Methods	1
1.3 Process Algebra	2
1.4 Time	3
1.5 Organization of Chapters	4
2 Untimed Process Algebra	5
2.1 Introduction	5
2.2 Elementary Principles	6
2.3 Basic Process Algebras	6
2.3.1 BPA	7
2.3.2 BPA_δ	13
2.3.3 BPA_ϵ	16
2.3.4 $BPA_{\delta\epsilon}$	18
2.3.5 BPA_δ^\dagger	19
2.4 Process Algebras	21
2.4.1 PA	21
2.4.2 PA_δ	23
2.4.3 PA_ϵ	24
2.4.4 PA_δ^\dagger	26
2.5 Algebras of Communicating Processes	28
2.5.1 ACP	28
2.5.2 ACP_ϵ	32
2.5.3 ACP_δ^\dagger	34
2.6 Properties	35
2.7 Other Topics	39
3 Discrete-Time Process Algebra	41
3.1 Introduction	41
3.2 Basic Process Algebras	42
3.2.1 $BPA_{\text{drt}}^- - \delta$	42
3.2.2 $BPA_{\text{drt}}^- - \text{ID}$	47
3.2.3 BPA_{drt}^-	51

3.2.4	BPA_{drt}^+	54
3.3	Properties	65
4	Soundness and Completeness	67
4.1	Introduction	67
4.2	Techniques	67
4.2.1	Proving Elimination	67
4.2.2	Proving Soundness	69
4.2.3	Proving Completeness	70
4.3	Results	72
4.3.1	$BPA_{drt}^- - \delta$	72
4.3.2	$BPA_{drt}^- - ID$	79
4.3.3	BPA_{drt}^-	85
4.3.4	BPA_{drt}^+	91
4.3.5	BPA'_{drt}	102
4.4	Conclusions	103
5	Axioms for Concurrency	105
5.1	Introduction	105
5.2	Process Algebras with Undelayable Actions	105
5.2.1	$PA_{drt}^- - ID$	105
5.2.2	$PA_{drt}^- - ID'$	113
5.2.3	$ACP_{drt}^- - ID$	121
5.2.4	$ACP_{drt}^- - ID'$	134
5.3	Process Algebras with Delayable Actions	141
5.3.1	PA_{drt}^+	142
5.3.2	PA'_{drt}	151
5.3.3	ACP_{drt}^+	152
5.3.4	ACP'_{drt}	167
5.3.5	ACP''_{drt}	168
5.4	Properties	172
5.5	Conclusions	173
6	Adding the Empty Process	177
6.1	Introduction	177
6.2	Design Goals for the Discrete-Time Empty Process	178
6.3	Process Algebras with Undelayable Actions	178
6.3.1	$BPA_{drt,\epsilon}^- - ID$	178
6.3.2	$PA_{drt,\epsilon}^- - ID$	187
6.3.3	$ACP_{drt,\epsilon}^- - ID$	198
6.3.4	Immediate Deadlock versus the Empty Process	202
6.4	Process Algebras with Delayable Actions	204
6.4.1	$BPA_{drt,\epsilon}^- - ID$	204
6.4.2	$PA_{drt,\epsilon}^- - ID$	207
6.4.3	$ACP_{drt,\epsilon}^- - ID$	209
6.5	Embeddings	211
6.6	Conclusions	212

7	Fischer's Protocol	213
7.1	Introduction	213
7.2	The Protocol	213
7.2.1	History	213
7.2.2	Proof Requirements	214
7.2.3	First Informal Description	215
7.2.4	Second Informal Description	215
7.3	Adding Silent Actions	216
7.3.1	Abstraction	216
7.3.2	ACP _{drt,τ}	217
7.4	Specification	219
7.5	Verification	221
7.6	Conclusions	227
8	Related Work	229
8.1	Introduction	229
8.2	ACP-Style Timed Process Algebras	229
8.2.1	ACP with Real-Time Steps (Groote)	229
8.2.2	Two-Phase Discrete-Time Process Algebra (Baeten and Bergstra)	230
8.2.3	Timestamped Discrete-Time Process Algebra (Baeten and Bergstra)	231
8.2.4	Real-Time Process Algebra (Baeten and Bergstra)	232
8.2.5	Real-Time Process Algebra with Infinitesimals (Baeten and Bergstra)	232
8.2.6	Real-Space Process Algebra (Baeten and Bergstra)	233
8.2.7	Real-Time Process Algebra (Klusener)	233
8.2.8	Algebra of Timed Frames (Bergstra, Fokkink, and Middelburg)	234
8.2.9	Timed μ CRL (Groote)	234
8.3	CCS-Style Timed Process Algebras	235
8.3.1	Temporal CCS (Moller and Tofts)	235
8.3.2	Timed CCS (Wang)	235
8.3.3	Timed CCS (Chen)	236
8.3.4	Timed Probabilistic CCS (Hansson)	236
8.3.5	Linear Timed CCS (Jeffrey)	237
8.3.6	CCS with Interval Time (Daniels)	237
8.3.7	Constraint-Oriented Real-Time Process Calculus (Fidge)	237
8.4	CSP-Style Timed Process Algebras	238
8.4.1	Timed CSP (Reed and Roscoe)	238
8.4.2	Discrete Timed CSP (Jeffrey)	238
8.5	LOTOS-Style Timed Process Algebras	238
8.5.1	Timed Calculus (Quemada, de Frutos, and Azcorra)	239
8.5.2	U-LOTOS / T-LOTOS (Bolognesi and Lucidi)	239
8.5.3	TLOTOS (Leduc)	239
8.5.4	Timed LOTOS / ET-LOTOS (Leduc and Léonard)	240
8.5.5	TE-LOTOS (Davies, Bryans, and Schneider)	240
8.5.6	LOTOS-T (Miguel, Fernández, and Vidaller)	240
8.5.7	LOTOS/T (Nakata, Higashino, and Taniguchi)	241
8.6	Other Timed Process Algebras	241
8.6.1	Algebra of Timed Processes (Nicollin and Sifakis)	241

8.6.2	Temporal Process Language (Hennessy and Regan)	244
8.6.3	Interval Process Algebra (Murphy)	245
8.6.4	Calculus of Timed Refinement (Čerāns)	245
8.6.5	Algebra of Comm. Shared Resources (Brémond-Grégoire <i>et al.</i>)	245
8.6.6	Abstract-Time Process Algebra (Jeffrey)	246
8.6.7	Process Algebra with Multiple Clocks (Andersen and Mendler)	246
8.6.8	Calculus for Synchrony and Asynchrony (Cleaveland <i>et al.</i>)	246
8.6.9	Real Time Agents (Cardelli)	246
8.6.10	Discrete-Time TOOLBUS (Bergstra and Klint)	246
8.7	Further Reading	247
9	Conclusions	249
9.1	Discrete-Time Process Algebra	249
9.2	Soundness and Completeness	249
9.3	Axioms for Concurrency	250
9.4	The Empty Process	251
9.5	Fischer's Protocol	251
9.6	Future Research	252
A	Overview of Axioms	255
A.1	Nomenclature	255
A.2	List of Axioms	256
B	Overview of Process Algebras	261
B.1	Nomenclature	261
B.2	Signatures	262
B.3	Axioms	263
C	Overview of Theorems	267
C.1	Elimination, Soundness, and Completeness	267
D	Notational Issues	269
D.1	Mathematical Symbols	269
D.2	Alternative Process Algebraic Notations	270
	Summary	273
	Samenvatting (Summary in Dutch)	277
	Acknowledgments	281
	Curriculum Vitae	283
	List of Tables	285
	Bibliography	289
	Symbol Index	303
	Axiom Index	307

Person Index 311

Index 315

1

Introduction

1.1 Prologue

One of the things that separates humans from other animals is the fact that humans have the ability to construct *tools*. Without tools, a human is a helpless, vulnerable, naked nothing. Yet, with tools, humans can fly faster than eagles, dive deeper than whales, and build larger communities than termites.

This tool-building ability does not restrict itself to purely physical tools. Next to the wheel, fire, the car, and the computer, we also have conceptual tools, such as language, law, and mathematics. It needs no further argumentation that the human tool-building ability is the key to the evolutionary success of the human species. Tools are a natural ingredient of everyday life. Nobody will try to cross the ocean swimming, or wage war with bare hands.

Nevertheless, when it comes to one of the most difficult tasks imaginable, *programming computers*, it is established practice to perform this task with the bare mind. Most software developers approach computers in the way a Neanderthaler would approach a car. The Neanderthaler might figure out how to use the steering wheel, but probably the ignition would be beyond him. He would be pushing the car around with his bare hands, and be quite proud of the achievement at that.

We are no better. Content with our current ability at programming, we happily produce software of extremely poor quality, marvel at the pretty pictures, and call it state-of-the-art technology. Meanwhile, automated systems fail in all sorts of ways.

This state of affairs is intolerable. We need conceptual tools to aid the mind in producing software of high quality.

1.2 Formal Methods

We live in a society that is increasingly dependent on information technology, where minute software malfunctions can cause great loss of productivity, money, and even lives. In the very readable account NEUMANN [152], we find a large collection of computer related mishaps, and many of these are due to faulty software. We name a few:

- When the first Space Shuttle was launched in 1981, faulty software caused one of the clocks aboard the spacecraft to be off by a fraction of a second. This delayed the launch by two days.
- During the Persian Gulf war, the Patriot anti-missile defense system's clock was also off by a fraction of a second, reducing the effectiveness of the system in intercepting incoming Iraqi Scud missiles from 95% to 13%.
- In 1991, a Japanese automated railway-signaling system failed, and when operators reverted to manual procedures, they caused a head-on collision between two trains, costing the lives of 42 passengers.
- In 1992, a system collapse at the London ambulance-dispatch control center caused ambulance service to be unavailable for 11 hours. Around 20 people died because medical attention could not be provided to them in time.

This list could be extended almost indefinitely. We conclude that programming as it is traditionally done, i.e., with the bare mind, leads to software systems that are not reliable enough.

What can be done to remedy this situation? One of the problems (although by far not the only one) is the fact that software systems often do not conform to their specifications, or do not even have clear specifications. In the Patriot example above, the clock problems were due to the fact that the system had been in operation for several weeks, while the clock could only provide accurate timing information for a few days at most. Had this been clearly specified during the construction of the system, then the system could have given a warning that it needed a clock adjustment. This is an unforgivable mistake; when a programmer implements a clock, it should function as a clock. It should provide the time within a specified margin of error, and that margin should be guaranteed. Simply implementing a clock, and hope for the best, will *not* do.

So, we must have a method that in some way guarantees that systems conform to their specifications. To do this, we need two things. First, we must write down a specification in a clear and unambiguous way, and secondly, we must provide a rigorous proof that the system we have implemented indeed conforms to this specification (such a proof is called a *verification*). Over the past two decades there has been a lot of research on such methods, which due to their formality and rigor are sometimes called *formal methods*.

As noted by CLARKE AND WING [63], formal methods have a history of not fulfilling promises. In the past, they were simply too inadequate, too obscure, or too expensive to apply to large-scale software systems. However, they also argue that recently we have seen the rise of more mature formal methods, methods that prove to be valuable tools in developing reliable software systems.

In this thesis we will look at one class of modern formal methods: *process algebra*.

1.3 Process Algebra

Of all formal methods that have been developed over the past two decades, there is a small class of methods, collectively called *process algebra*, that will have our special attention. These methods have in common that they find their roots in algebra; they all provide methods for calculating algebraically with processes.

Recalling the tool metaphor from Section 1.1, we can describe process algebra as a tool to help the mind reason about software:

- First, it can be used to *specify* exactly what a certain piece of software *should do*, i.e., help us express what exactly we do expect from that piece of software.
- Secondly, it can be used to formally grasp the semantics of an *implementation* of that piece of software, i.e., tell us what it actually *does do*.
- Thirdly, it can tell us about the *relation* between these two aspects: will our piece of software behave the way we expect it to? If not, in what aspect? And why?
- Fourthly, in the process of specifying, implementing, and verifying a piece of software, process algebra helps us to gain *insight*. Insight about the problem we are trying to solve, but also insight about the way we might solve it. Due to the formal and precise nature of process algebra, it leads our minds onto paths they would not have taken so easily otherwise.

The fact that these four aspects can all be pursued using the same formalism is one of the biggest strengths of process algebra: it provides a unifying language for the study of many different stages of software development.

In this thesis, we will look at one specific type of process algebra: the *Algebra of Communicating Systems* (ACP) as introduced by BERGSTRA AND KLOP [45]. **When in this thesis we speak of *process algebra*, we mean *ACP-style process algebra*, unless explicitly indicated otherwise.**

1.4 Time

Of the four software related mishaps we mentioned in Section 1.2, two had to do with *time*. This is no coincidence: many modern software systems are not only expected to produce correct results, they are also expected to do so *in time*.

Let us give some examples. Suppose you have a mobile telephone, and are making a call while driving in your car. Then, sooner or later, you will get out of reach of the base station your mobile phone is in contact with. When this happens, a certain piece of software in the telephone network will quickly calculate which other base station is closest to you at the moment, and switch your call over to there. All you notice is a short beep, and you can continue your call. It is obvious that this procedure should be completed quickly: if the telephone network takes minutes to find you a new base station, you might already be completely out of reach, and your call would have been terminated.

Then again, making a call from your car may not be a good idea anyway. Suppose you are distracted by the conversation, and bump into another car. At that moment, a piece of software within your airbag safety device will quickly decide whether this is a minor accident, which does not warrant airbag activation, or a big one, which requires the airbag to be inflated within the next few milliseconds. Again, this decision should be made quickly, as with a full-blown crash in progress, every millisecond can mean the difference between life and death.

These examples show that the timing aspects of software can often be an integral part of the overall correctness. Therefore, any method for analyzing such software should be able to take time into account.

Process algebra, initially, was not very well equipped to handle timing aspects. Over the years, however, many extensions to process algebra have been proposed that extend standard process algebra with mechanisms for reasoning about time.

One of these extensions, called *discrete-time process algebra*, will be the object of study of this thesis.

1.5 Organization of Chapters

Finally, we give an overview of the chapters of this thesis, and how they are organized:

- In Chapter 2 we give an introduction into process algebra, treating all the subjects that are relevant for this thesis.
- In Chapter 3 we extend the (untimed) process algebra of Chapter 2 to *discrete-time process algebra*, a variant of process algebra that allows us to specify and analyze timing aspects of protocols in an explicit way.
- In Chapter 4 we give a number of theoretical results about the process algebras we have treated in Chapter 3. Most importantly, we prove *soundness* and *completeness* results.
- In Chapter 5 we extend some process algebras treated in Chapter 4 with *concurrency* constructs; constructs that allow us to talk about processes that consist of multiple components that are simultaneously active.
- In Chapter 6 we examine the combination of *discrete-time process algebra* and the *empty process*. We discuss the difficulties we have encountered with this combination, motivate our solutions, and state a number of theoretical results.
- In Chapter 7 we apply *discrete-time process algebra* to the verification of a protocol, namely *Fischer's Protocol for mutual exclusion*, a protocol whose correctness heavily depends on timing aspects.
- In Chapter 8 we discuss work on related topics that has been published by others.
- In Chapter 9, finally, we give an overview of the results from this thesis, indicate the strong and the weak points of the methods we have used and developed, and discuss future research.


Because some of the chapters get very technical, we have provided some appendices and indices for reference purposes. Together with the summary, acknowledgments, curriculum vitae, list of tables, and bibliography they form the final part of this thesis.

2

Untimed Process Algebra

2.1 Introduction

In this chapter we give an overview of some basic aspects of process algebra, as introduced by BERGSTRA AND KLOP [45] in 1984. Since then, many of developments have taken place; for a comprehensive overview of more recent developments, see the textbook by BAETEN AND WEIJLAND [38], published in 1990, or the handbook article by BAETEN AND VERHOEF [37], published in 1995. We will limit ourselves to describing those aspects of process algebra that are relevant for the present thesis.

The purpose of this overview is twofold. First, we want to give the reader a solid intuitive understanding of all relevant concepts involved, in their pure (i.e. untimed) form. So, we give an intuitive motivation for every definition, theorem, etc., and give examples where needed. When in later chapters some issue seems unclear or confusing, it may be useful to look at the same problem in the smaller, untimed setting of this chapter, and read the relevant motivations once more. Note that the motivations are always preceded by a “helping hand” symbol, , to visually separate the formal from the informal text.

Secondly, we want to lay a basis upon which to extend untimed process algebra with discrete time. The definitions we give are therefore carefully chosen to avoid running into insignificant, but very irritating, technical problems later on when we add discrete-time extensions. Consequently, readers familiar with process algebra will find our definitions slightly different from those given in, e.g., BAETEN AND WEIJLAND [38]. In this way, we can make a clean separation between issues involving process algebra *per se* (not necessarily a subject of this thesis), and issues involving *discrete-time* process algebra specifically (very much a subject of this thesis).

In view of these goals, we have chosen to omit all proofs from this section; they are neither relevant nor interesting for our purpose here. In Chapters 4, 5, and 6 we will give plenty of proofs; and since most of these proofs concern process algebras that are clean extensions of the untimed ones treated in this chapter, they do also apply to the untimed case.

On first reading one should probably not bother too much with the subtle intricacies of our definitions, but rather concentrate on the examples and the motivations given. In later chapters, when this chapter comes to serve as a reference, the fine details of the

definitions will be much better appreciated. Note that some remarks in this chapter are numbered; those remarks are of a rather formal nature, falling somewhere in between the definitions and the motivations.

2.2 Elementary Principles

Process algebra is a means of *specifying, verifying, and in general, talking about* computer algorithms and protocols (formally called *processes*) in a clearly defined, formal manner. Compared with other formalisms developed for this purpose, process algebra's strong point is the fact that it is based on algebra. In this way, calculations with processes become algebraic calculations, allowing room for high-level abstractions to be made, potentially avoiding the complexity blow-up that is always just around the corner in this area of computer science.

The whole of methods and formalisms collectively known as *process algebra* can be subdivided into separate parts of which every single one, again, is called a *process algebra*. Confusion between these two uses is not necessary; usually the presence of an article ("the" or "a") indicates we are not talking about the whole.

At the heart of every process algebra lies a set of *axioms*. Every axiom consists of an *equality* between two *process terms* (that may contain free variables). Behind every axiom there is an intuitive motivation: an insight that explains why these two process terms should be considered equal. Together these axioms lead to an *algebra*: a mathematical structure that allows for the manipulation of terms.

Given the set of axioms of a certain process algebra, it is possible to construct a *model*: a mathematical "world" that obeys the equalities given by the axioms. Such a model is called a *semantics* for that algebra. Typically, several clearly distinct models can be given for any given process algebra. However, there is a tendency always to use the same kind of model, called a *bisimulation model*.

Operationally, a *process* is an entity that executes a number of *actions*, where the question whether or not a certain action can execute at a given moment depends very subtly on both visible and invisible aspects of the execution sequence so far. Such *actions* are the most elementary part of the execution. They take no time, and cannot be further subdivided; they are *atomic*.

In the following section, we will give concrete examples of the concepts mentioned above, and explain the intuitions that are behind them.

2.3 Basic Process Algebras

We start by treating the most simple process algebras of them all, historically called the class of *Basic Process Algebras (BPA's)*. These process algebras form the core around which all other process algebras are built. They contain just two operators: one that expresses *choice*, and another one expressing *sequential execution*.

2.3.1 BPA

In this section, we introduce the process algebra BPA. Before we can do this, we need some introductory definitions.

First we need an *alphabet* whose symbols will stand for the *actions* that our processes can execute.

Definition 2.3.1.1 (Alphabet)

For this section, and all sections to come, we presume the existence of a fixed, finite alphabet A , that can be considered a parameter of the respective process algebras. Furthermore, we define A_δ as $A \cup \{\delta\}$, $A_{\delta\varepsilon}$ as $A \cup \{\delta, \varepsilon\}$, A_τ as $A \cup \{\tau\}$, $A_{\delta\tau}$ as $A \cup \{\delta, \tau\}$, and A_σ as $A \cup \{\sigma\}$, where δ , ε , τ , and σ are (yet to be treated) symbols that are not contained in A .

Example 2.3.1.2 (Alphabet)

Suppose we want to use process algebra to study a *cola machine*; the kind where you insert money, push a button, and get a can of cola served. If we abstract our description to a level where those three actions suffice to describe everything, we might choose our alphabet as follows:

$$A = \{\text{insert-money, push-button, serve-cola}\}$$

or, if we would want to be less verbose:

$$A = \{i, p, s\}$$

☞ In practice, we will mostly choose lowercase letters from the beginning of the Roman alphabet as our symbols: a , b , c , etc.

We will use cola machines as our running example throughout this section. But before we can start building one, we first need some *operators*.

The first operator we introduce is the *choice operator*, also called the *alternative composition operator*. This operator expresses the choice between two processes; given processes x and y , the process that executes either x or y is denoted as $x + y$. The second operator is the *sequential composition operator*. This operator executes two processes sequentially; given again x and y , the process that first executes x , and after x has finished, continues with y is denoted as $x \cdot y$.

Now we are ready to start defining BPA. First, we define the *signature* of BPA, that is, the constants and operators from which the process terms of BPA are built.

Definition 2.3.1.3 (Signature of BPA)

The signature of BPA consists of the *actions* $\{a \mid a \in A\}$, the *alternative composition operator* $+$, and the *sequential composition operator* \cdot .

Remark 2.3.1.4 (Symbol versus Action)

Note that in Definition 2.3.1.3, in the expression $\{a \mid a \in A\}$, the second a refers to the *symbol* a , while the first one refers to the *action* a . This distinction should be clearly made, and it can be considered a tragic historical incident that these different notions have received the same notation.

Definition 2.3.1.5 (Closed Terms, Open Terms, Process Terms)

Expressions over the signature of a given process algebra are called *closed terms*. When we allow free variables to appear in such an expression, the result is called an *open term*. Open terms are sometimes also called *process terms*.

The set of all closed terms of a given process algebra P is denoted by $C(P)$.

☞ Using these definitions, we now introduce the set of *axioms* associated with BPA. Every axiom consists of an equality between two process terms. There are five of them, named A1 through A5. Three of them are concerned with the $+$ operator, one with the \cdot operator, and one with the interaction between the two.

Definition 2.3.1.6 (Axioms of BPA)

The process algebra BPA is axiomatized by Axioms A1–A5 shown in Table 2.1: BPA = A1–A5.

$x + y = y + x$	A1
$(x + y) + z = x + (y + z)$	A2
$x + x = x$	A3
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5

Table 2.1: Axioms of BPA.

☞ The axioms of BPA capture the very essence of process algebra; they, and only they, are included in *every* process algebra we will describe. None of these axioms is very surprising; we discuss them shortly.

Axiom A1 expresses the *commutativity* of the alternative composition: making a choice between x and y is the same as making a choice between y and x . Axiom A2 expresses the *associativity* of the alternative composition: when choosing among three alternatives, the grouping of the alternatives is irrelevant. Finally, Axiom A3 expresses the *idempotency* of the alternative composition: choosing between x and x is the same as doing x right away. Then we have Axiom A4, which expresses the *right-distributivity* of the sequential composition over the alternative composition: choosing between x and y , and after that doing z is the same as choosing between x followed by z and y followed by z . Last, Axiom A5 expresses the *associativity* of the sequential composition: when doing x , followed by y , followed by z , the grouping of the three processes is irrelevant.

The most surprising axiom of BPA is probably the one that *is not present*: namely the *left-distributivity* of the \cdot over the $+$. This axiom, Axiom LD, is shown in Table 2.2 on the facing page. An argument against this axiom could be that it unifies too many processes: it may very well be the case that the execution of x yields information that influences the following choice between y and z . Hence, it would be incorrect to let $x \cdot (y + z)$ be equal to $x \cdot y + x \cdot z$. This, however, is a matter of taste, and making the unification or not leads to entirely different theories. If we would include Axiom LD, we would get a so-called *trace semantics*, also called *linear-time semantics*. Choosing not to include it, as we do,

leads to a so-called *bisimulation semantics*, also called *branching-time semantics*. For a full discussion of these two opposites, and the whole spectrum in between them, see the dissertation of VAN GLABBEK [83].

$$x \cdot (y + z) = x \cdot y + x \cdot z \quad \text{LD}$$

Table 2.2: Left-distributivity.

Definition 2.3.1.7 (Operator Precedence)

Throughout this thesis we adhere to the following operator precedence scheme, which consists of four categories of operators. The four categories, from strongly binding to weakly binding, are:

- (i). all unary operators,
- (ii). the sequential composition operator \cdot ,
- (iii). all binary operators, except $+$ and \cdot ,
- (iv). the alternative composition operator $+$.

Within one category, all operators bind equally strong.

☞ This precedence scheme allows us to get rid of as many parentheses as possible; for example, we need not write anymore $(a \cdot b) + c$, as we can use $a \cdot b + c$ instead. Given the associativity of the $+$ and the \cdot (as expressed by Axioms A2 and A5), we also need not write $(a + b) + c$ and $(a \cdot b) \cdot c$; we can use $a + b + c$ and $a \cdot b \cdot c$ instead. Finally, note that the operators in categories (i) and (iii) also include all operators that have yet to be defined (there will be no operators of arity three or higher).

Definition 2.3.1.8 (Derivability Relation)

Process terms x and y in a certain process algebra P that can be rewritten into each other, using equational logic, are called *derivably equal*, denoted by the *derivability relation* symbol \vdash and the *algebraic equality* symbol $=$:

$$P \vdash x = y$$

☞ For example, in BPA we can rewrite $c + b + b + a$ into $a + b + c$ by repeatedly applying Axioms A1-A3. Formally we denote this by $\text{BPA} \vdash c + b + b + a = a + b + c$.

Example 2.3.1.9 (Cola Machine in BPA)

Suppose we have actions i , p , and s standing for “insert coin”, “push button”, and “serve cola”. We could then model our cola machine as:

$$\text{cola-machine}_1 = i \cdot p \cdot s$$

where the equality symbol can be interpreted as “we define the process cola-machine_1 as follows ...” (alternatively, the equation can be viewed as an axiom for the new constant cola-machine_1).

Let us refine the “insert coin” action. Suppose cola costs 15 Euro cents (wishful thinking, but so is the Euro), and we may insert 5 and 10 Euro-cent coins, denoted by i_5 and i_{10} respectively. Then we could write:

$$\text{cola-machine}_2 = (i_{10} \cdot i_5 + i_5 \cdot i_{10} + i_5 \cdot i_5 \cdot i_5) \cdot p \cdot s$$

Alas, this is slightly different from what we want; if we now insert a 5 cent coin, we may either end up in the second or third summand (the choice is made non-deterministically, and cannot be observed from the outside). Were we to end up in the third, while we only have a 10 cent coin left, we lose: we cannot continue, as cola-machine_2 expects us to insert a 5 cent coin.

We will repair this by using a more carefully chosen branching structure:

$$\text{cola-machine}_3 = (i_{10} \cdot i_5 + i_5 \cdot (i_{10} + i_5 \cdot i_5)) \cdot p \cdot s$$

This is better: cola-machine_3 accepts 15 cents in any combination of 5 and 10 cent coins.

Definition 2.3.1.10 (Notation regarding Semantics, Part I)

In order to define a semantics, we will use term-deduction system semantics, also called “Structured Operational Semantics” or “Plotkin-style semantics”. (See PLOTKIN [166], where “Structured Operational Semantics” is called “*Structural* Operational Semantics”. For arguments why “structured” is more appropriate than “structural” here, see Section 1.2.1 of GROOTE [88].) We use the notation $x \xrightarrow{a} x'$ to denote that x can do an a -step (also called a -transition or action step) to x' , and $x \xrightarrow{a} \surd$ to denote that x can do an a -step and then terminate. Apart from the obvious negations of these, we also use $x \not\xrightarrow{a}$ to indicate that x cannot do an a -step, and $x \not\rightarrow$ to indicate that x cannot do any step whatsoever. Finally, we write $x \xrightarrow{a_1, \dots, a_n} x'$ as a shorthand for $x \xrightarrow{a_1} x', \dots, x \xrightarrow{a_n} x'$.

For each process algebra P we define, we will give a term-deduction system, denoted by $T(P)$. By using the concept of bisimulation (to be defined in Definition 2.3.1.13 on the next page), we then turn the term-deduction system into a model of the given axioms.

Definition 2.3.1.11 (Semantics of BPA)

The semantics of BPA are given by the term-deduction system $T(\text{BPA})$ induced by the deduction rules shown in Table 2.3 and Table 2.4 on the next page. In these deduction rules, a is a variable that ranges over the alphabet A .

$$a \xrightarrow{a} \surd$$

Table 2.3: Deduction rule for untimed actions.

☞ These deduction rules capture the *operational behavior* of BPA. For example, the one in Table 2.3 expresses that the process a can do an a -step, and then terminate, and the first one of Table 2.4 on the facing page expresses the fact that when x can execute an a , then also can the alternative composition of x and y . Note that it does not say that $x + y$ *must* execute an a , only that it *may*.

$$\begin{array}{ccc}
\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} & \frac{x \xrightarrow{a} \surd}{x + y \xrightarrow{a} \surd} & \frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y} \\
\\
\frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'} & \frac{y \xrightarrow{a} \surd}{x + y \xrightarrow{a} \surd} & \frac{x \xrightarrow{a} \surd}{x \cdot y \xrightarrow{a} y}
\end{array}$$

Table 2.4: Deduction rules for alternative and sequential composition.

We use the predicate $\xrightarrow{a} \surd$ to denote that a process may execute an a , and then terminate. So, the last deduction rule expresses the fact that when x may do an a and then terminate, then x followed by y may execute an a , and then further continue as y . Note that \surd is not a process, but just part of the notation of the predicate $\xrightarrow{a} \surd$.

Definition 2.3.1.12 (Modeling Relation)

If for a term-deduction system $T(P)$ we want to express that a certain (transition) relation or predicate φ is *modeled* by $T(P)$, we denote this using the *modeling relation* \models :

$$T(P) \models \varphi$$

When there can be no confusion as to which term-deduction system is intended, we just write φ , as we have done above.

☞ For example, using the deduction rules shown in Table 2.4, we could write $T(\text{BPA}) \models (a + b) \cdot c \xrightarrow{a} c$.

Definition 2.3.1.13 (Bisimulation for BPA)

Bisimulation for BPA is defined as follows; a binary relation R on closed terms is a bisimulation iff the following transfer conditions hold for all closed terms p and q :

- (i). If $R^S(p, q)$ and $T(\text{BPA}) \models p \xrightarrow{a} p'$, where $a \in A$, then there exists a process term q' such that $T(\text{BPA}) \models q \xrightarrow{a} q'$ and $R^S(p', q')$,
- (ii). if $R^S(p, q)$ and $T(\text{BPA}) \models p \xrightarrow{a} \surd$, where $a \in A$, then $T(\text{BPA}) \models q \xrightarrow{a} \surd$.

Two closed BPA terms p and q are bisimilar, notation $p \sim_{\text{BPA}} q$, if there exists a bisimulation relation R such that $R(p, q)$. Where there can be no confusion, we abbreviate this by $p \sim q$.

☞ Note that by this definition the empty relation is also a valid bisimulation relation, although it does not lead to bisimilarity of any two BPA terms.

Remark 2.3.1.14 (Bisimulation)

The concept of bisimulation was first introduced by PARK [160].

Example 2.3.1.15 (Bisimulation)

If we take the two BPA processes p and q defined by $p \equiv a \cdot c + b \cdot c$ and $q \equiv (a + b) \cdot c$, then we have $p \sim_{\text{BPA}} q$ as we can construct the following bisimulation:

$$R = \{(a \cdot c + b \cdot c, (a + b) \cdot c), (c, c)\}$$

Considering that the only transitions of p and q are:

$$\begin{aligned} p &\equiv a \cdot c + b \cdot c \xrightarrow{a,b} c \xrightarrow{c} \surd \\ q &\equiv (a + b) \cdot c \xrightarrow{a,b} c \xrightarrow{c} \surd \end{aligned}$$

it is easy to check that R is indeed a bisimulation that relates p and q .

On the other hand, if we take $p \equiv a \cdot b + a \cdot c$ and $q \equiv a \cdot (b + c)$, then it is not possible to find a bisimulation R that relates p and q . This can be easily understood: first note that the only transition of q is the following: $q \equiv a \cdot (b + c) \xrightarrow{a} b + c$. As $p \equiv a \cdot b + a \cdot c \xrightarrow{a} b$, we must necessarily have that $R^S(b, (b + c))$. However, this is not possible, as $b + c \xrightarrow{c} \surd$, whereas $b \not\xrightarrow{c} \surd$, which is a violation of the second transfer condition of Definition 2.3.1.13. So, we have a contradiction, and no bisimulation exists between p and q .

☞ Bisimulation captures, for closed terms, precisely the equalities induced by the axioms and equational logic. If we look at the example above, we first see two terms that are equal by Axiom A4 and that are also bisimilar. Then, there are two terms that are not equal on the basis of our axioms, and indeed, they are not bisimilar.

We will formalize and prove this claim in Chapter 4, which is completely dedicated to this and related issues.

Definition 2.3.1.16 (Bisimulation Model for BPA)

Using bisimulation, we can now construct a model of the axioms of BPA. In order to do this, we first need to know that bisimulation is a *congruence* (see Property 2.6.1.1 on page 35) with respect to all operators. VERHOEF [194] proves that a sufficient condition for this is that:

- (i). The deduction rules are in the so-called *panth* format,
- (ii). the deduction rules are *well-founded*,
- (iii). a *stratification* can be given for the deduction rules.

It is easy to check that these three conditions are indeed satisfied. We will not give definitions of the concepts mentioned in these conditions, as that would go beyond the scope of this thesis.

We now construct the bisimulation model for BPA by taking the equivalence classes of the set of all closed BPA terms, with respect to bisimulation equivalence. As bisimulation is a congruence, the operators can be trivially defined on the equivalence classes. For example for the $+$ operator:

$$[x]_{\sim} + [y]_{\sim} = [x + y]_{\sim}$$

Here $[x]_{\sim}$ denotes the equivalence class of x with respect to the bisimulation equivalence relation \sim . The other operators are defined in the same way.

☞ The bisimulation model shown above, is by no means the only model of BPA. A very simple one is for example the one-point model, where every process term is associated with the one point of the model. But there are many more, such as the projective-limit model, or the term model. We will only concern ourselves with bisimulation models of the kind defined above. For an overview of other models, see BAETEN AND WEIJLAND [38] or the dissertation of BLANCO [48].

Definition 2.3.1.17 (Basic Terms of BPA)

We define *deadlock-free basic terms* inductively as follows:

- (i). For every $a \in A$, a is a deadlock-free basic term,
- (ii). if $a \in A$ and t is a deadlock-free basic term, then $a \cdot t$ is a deadlock-free basic term,
- (iii). if s and t are deadlock-free basic terms, then $s + t$ is a deadlock-free basic term.

From now on, when we speak of basic terms in the context of BPA, we mean deadlock-free basic terms.

Example 2.3.1.18 (Basic Terms of BPA)

These are all basic terms of BPA:

$$a, a \cdot b, a + b, a \cdot (b + c), a \cdot (b \cdot c), a + (b + c), (a + b) + c$$

These, however, are not:

$$(a + b) \cdot c, (a \cdot b) \cdot c$$

☞ Later on we will show that for every closed term, there exists a basic term such that both are derivably equal by the axioms. Therefore, basic terms are very useful with regard to proving general properties: often it is enough to use induction on the structure of basic terms to prove a certain property for all closed terms.

Still, our definition is by no means the only definition that has this nice property. Hence, the term *basic* is quite misleading. There is nothing basic about them; we could just as well have named them “yet another normal form”. In fact, in Chapter 6 we will encounter a situation where we need *two* very distinct kinds of “basic” terms within *one* process algebra! Despite this, we will conform to the usage of the phrase “basic term” as in the literature, because we do not want to create more confusion than we are trying to remedy.

2.3.2 BPA_δ

In this section, we extend the process algebra BPA with the so-called *deadlock process*. We introduce a new constant, denoted δ , that will stand for the process which can do nothing; neither execute an action, nor terminate successfully. The extended process algebra that results is called BPA_δ.

Definition 2.3.2.1 (Signature of BPA_δ)

The signature of BPA_δ consists of the *actions* $\{a \mid a \in A\}$, the *deadlock constant* δ , the *alternative composition operator* $+$, and the *sequential composition operator* \cdot .

Definition 2.3.2.2 (Axioms of BPA_δ)

The process algebra BPA_δ is axiomatized by the axioms of BPA given in Definition 2.3.1.6 on page 8 and Axioms A6–A7 shown in Table 2.5: $BPA_\delta = A1$ –A7.

$$\begin{array}{ll} x + \delta = x & \text{A6} \\ \delta \cdot x = \delta & \text{A7} \end{array}$$

Table 2.5: Additional axioms for BPA_δ .

☞ As shown above, BPA_δ is very much like BPA. We have one new constant, δ , and two new axioms.

The new constant δ denotes the process that cannot do anything, is irreparably “stuck”, or, as it is sometimes said, denotes unsuccessful termination.

The first new axiom, Axiom A6, expresses that δ is a proper zero element with respect to the alternative composition: when making a choice between doing x or getting stuck, we never choose the latter. The second, Axiom A7, expresses that δ is a left-zero element with respect to the sequential composition: being stuck followed by doing x afterwards, is the same as just being stuck. Note that a is not equal to $a \cdot \delta$: the former can do an a , and terminate successfully, while the latter will end in deadlock after it has done an a .

From these axioms, it becomes clear that the alternative composition in process algebra is neither completely deterministic, nor completely non-deterministic. When a process contains a deadlock summand, the alternative composition tries to avoid it whenever possible, thus behaving in a deterministic way. However, when we look at a process such as $a \cdot b + a \cdot c$, then the choice between the first and the second summand is completely open, and the alternative composition behaves non-deterministically.

Remark 2.3.2.3 (Axiom A6 versus Axiom A6A)

Notice that in the presence of the other axioms of BPA_δ , Axiom A6 shown in Table 2.5 is equivalent, for *closed* BPA_δ terms, with Axiom A6A shown in Table 2.6. Therefore, we could replace A6 in BPA_δ by A6A without affecting the soundness or completeness (to be treated, see Section 2.6) of the resulting theory.

One such reason to do so, could be the fact that A6A remains valid in all discrete-time process algebras we will describe, whereas A6 does not. Still, for historical reasons, we prefer A6 to be used in the definition of BPA_δ .

$$a + \delta = a \quad \text{A6A}$$

Table 2.6: Alternative for axiom for deadlock.

Example 2.3.2.4 (Cola Machine in BPA_δ)

Let us use our new deadlock process to refine cola-machine_3 of Example 2.3.1.9 a little further. Suppose we have a cola machine that is willing to accept too much money, but reacts to this not by giving cola, but by locking up:

$$\text{cola-machine}_4 = (i_{10} \cdot (i_5 + i_{10} \cdot \delta) + i_5 \cdot (i_{10} + i_5 \cdot (i_5 + i_{10} \cdot \delta))) \cdot p \cdot s$$

This is just like cola-machine_3 , except that it goes into a deadlock state when it gets a chance to swallow 20 cents.

Definition 2.3.2.5 (Range of a)

When we write a (or b , or c , etc.) in the context of an equality or an ordering, we mean this a to range over A_δ (provided, of course, deadlock is part of the relevant signature). When we write it in the context of a deduction rule, we mean it to range over A . In all other cases, or when we deviate from the above rule, we explicitly state whether it ranges over A or A_δ .

Definition 2.3.2.6 (Semantics of BPA_δ)

The semantics of BPA_δ are given by the term-deduction system $T(BPA_\delta)$ induced by the deduction rules shown in Table 2.3 on page 10 and Table 2.4 on page 11.

☞ Note that the term-deduction system $T(BPA_\delta)$ is almost identical to the term deduction system $T(BPA)$ given in Definition 2.3.1.11 on page 10, as there are no deduction rules for δ . However, $T(BPA_\delta)$ does differ from $T(BPA)$ in the fact that it contains the symbol δ in its signature.

Definition 2.3.2.7 (Bisimulation and Bisimulation Model for BPA_δ)

Bisimulation for BPA_δ and the corresponding bisimulation model for BPA_δ are defined in the same way as for BPA . Replace “ BPA ” by “ BPA_δ ” in Definition 2.3.1.13 on page 11 and Definition 2.3.1.16 on page 12.

Definition 2.3.2.8 (Basic Terms of BPA_δ)

We define *standard basic terms* inductively as follows:

- (i). For every $a \in A_\delta$, a is a standard basic term,
- (ii). if $a \in A_\delta$ and t is a standard basic term, then $a \cdot t$ is a standard basic term,
- (iii). if s and t are standard basic terms, then $s + t$ is a standard basic term.

From now on, when we speak of basic terms in the context of BPA_δ , we mean standard basic terms.

☞ The basic terms of BPA_δ are almost the same as those of BPA , the only difference being that a is now allowed to range over A_δ instead of just A . So, all the basic terms of BPA from Example 2.3.1.18 on page 13 are basic terms of BPA_δ too, also when we replace the a , b , or c by a δ . Note that this is different from, e.g., BAETEN AND WEIJLAND [38], where $\delta \cdot t$ is never a basic term. We find that our definition is cleaner, and leads to more straightforward proofs later on. Our definition is also *broader*, but as said before, the goal of basic terms is to facilitate proofs, not to define as “basic” a class of terms possible.

2.3.3 BPA_ε

In this section, we extend the process algebra BPA with the so-called *empty process*. We introduce a new constant, denoted ε , that will stand for the process which does nothing and terminates successfully. The extended process algebra that results is called BPA_ε .

Definition 2.3.3.1 (Signature of BPA_ε)

The signature of BPA_ε consists of the *actions* $\{a \mid a \in A\}$, the *empty process constant* ε , the *alternative composition operator* $+$, and the *sequential composition operator* \cdot .

Definition 2.3.3.2 (Axioms of BPA_ε)

The process algebra BPA_ε is axiomatized by the axioms of BPA given in Definition 2.3.1.6 on page 8 and Axioms A8–A9 shown in Table 2.7: $BPA_\varepsilon = A1\text{--}A5 + A8\text{--}A9$.

$$\begin{array}{ll} x \cdot \varepsilon = x & \text{A8} \\ \varepsilon \cdot x = x & \text{A9} \end{array}$$

Table 2.7: Additional axioms for BPA_ε .

☞ Axioms A8 and A9 are as straightforward as can be; they express that ε is a proper unit element of the sequential composition.

Definition 2.3.3.3 (Notation regarding Semantics, Part II)

We extend our notation for defining term-deduction systems with a new predicate: let $x \downarrow$ express that x has the option to terminate successfully, and $x \not\downarrow$ that x does not have the option to terminate successfully.

Definition 2.3.3.4 (Semantics of BPA_ε)

The semantics of BPA_ε are given by the term-deduction system $T(BPA_\varepsilon)$ induced by the deduction rules shown in Table 2.8 and Table 2.9 on the next page.

$$a \xrightarrow{a} \varepsilon \quad \varepsilon \downarrow$$

Table 2.8: Deduction rules for untimed actions with empty process.

☞ The deduction rules for BPA_ε are somewhat different from the ones we encountered earlier. The $x \xrightarrow{a} \surd$ predicate that was used to express that x could do an a -step and then terminate is gone, and we now use $x \downarrow$ to express that x has an *option to terminate*. Consequently, we do not anymore say anything about the termination behavior of a single action directly, but instead define that a can do an a -step to the empty process, and then define the termination behavior of the empty process. In this way, we still have that a can do an a -step and then terminate, but now defined in a more indirect way.

$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'}$	$\frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$	$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y}$	$\frac{x \downarrow, y \xrightarrow{a} y'}{x \cdot y \xrightarrow{a} y'}$
$\frac{x \downarrow}{(x + y) \downarrow}$	$\frac{y \downarrow}{(x + y) \downarrow}$	$\frac{x \downarrow, y \downarrow}{(x \cdot y) \downarrow}$	

Table 2.9: Deduction rules for alternative and sequential composition with empty process.

Apart from that, the new deduction rules are hardly surprising. The fourth one from Table 2.9, for example, expresses that if x has the option to terminate and y can do an a , then x followed by y can also do an a .

Definition 2.3.3.5 (Bisimulation for BPA_ε)

Bisimulation for BPA_ε is defined as follows; a binary relation R on closed terms is a bisimulation iff the following transfer conditions hold for all closed terms p and q :

- (i). If $R^S(p, q)$ and $T(BPA_\varepsilon) \models p \xrightarrow{a} p'$, where $a \in A$, then there exists a process term q' such that $T(BPA_\varepsilon) \models q \xrightarrow{a} q'$ and $R^S(p', q')$,
- (ii). if $R^S(p, q)$ and $T(BPA_\varepsilon) \models p \downarrow$, then $T(BPA_\varepsilon) \models q \downarrow$.

☞ The introduction of the $x \downarrow$ predicate necessitates a correspondingly changed definition of bisimulation: we now require that the value of this predicate is the same for terms that are related.

Definition 2.3.3.6 (Bisimulation Model for BPA_ε)

The bisimulation model for BPA_ε is defined in the same way as for BPA. Replace “BPA” by “ BPA_ε ” in Definition 2.3.1.16 on page 12.

Definition 2.3.3.7 (Basic Terms of BPA_ε)

We define ε -basic terms inductively as follows:

- (i). The constant ε is an ε -basic term,
- (ii). if $a \in A$ and t is an ε -basic term, then $a \cdot t$ is an ε -basic term,
- (iii). if s and t are ε basic terms, then $s + t$ is an ε -basic term.

From now on, when we speak of basic terms in the context of BPA_ε , we mean ε -basic terms.

Example 2.3.3.8 (Basic Terms of BPA_ε)

These are all basic terms of BPA_ε :

$$a \cdot \varepsilon, a \cdot b \cdot \varepsilon, a \cdot \varepsilon + b \cdot \varepsilon, a \cdot (b \cdot \varepsilon + c \cdot \varepsilon)$$

These, however, are not:

$$a, a \cdot b, a + b$$

☞ The basic terms of BPA_ε are built “starting from ε ”. Therefore, every basic term of BPA_ε contains an ε , and none of the basic terms of BPA is a basic term of BPA_ε . We could have included all the basic terms of BPA in the definition of basic terms of BPA_ε , but in this case defining basic terms *narrowly* makes our life easier: in induction proofs we now only have one base case (namely: ε) to deal with.

2.3.4 $\text{BPA}_{\delta\varepsilon}$

In this section, we combine the deadlock process and empty process of Sections 2.3.2 and 2.3.3. The extended process algebra that results is called $\text{BPA}_{\delta\varepsilon}$.

Definition 2.3.4.1 (Signature of $\text{BPA}_{\delta\varepsilon}$)

The signature of $\text{BPA}_{\delta\varepsilon}$ consists of the *actions* $\{a \mid a \in A\}$, the *deadlock constant* δ , the *empty process constant* ε , the *alternative composition operator* $+$, and the *sequential composition operator* \cdot .

Definition 2.3.4.2 (Axioms of $\text{BPA}_{\delta\varepsilon}$)

The process algebra $\text{BPA}_{\delta\varepsilon}$ is axiomatized by the axioms of BPA_δ that are given in Definition 2.3.2.2 on page 14 and Axioms A8–A9 shown in Table 2.7 on page 16: $\text{BPA}_{\delta\varepsilon} = \text{A1–A9}$.

Example 2.3.4.3 (Cola Machine in $\text{BPA}_{\delta\varepsilon}$)

We use the empty process to further refine cola-machine_4 of Example 2.3.2.4. Suppose our cola machine breaks down further, and now chances are it even will not give cola when it formerly did. We could model this as follows:

$$\text{cola-machine}_5 = (i_{10} \cdot (i_5 + i_{10} \cdot \delta) + i_5 \cdot (i_{10} + i_5 \cdot (i_5 + i_{10} \cdot \delta))) \cdot p \cdot (s + \varepsilon)$$

This is just like cola-machine_4 , except that after having paid 15 cents, we run the risk of nothing happening. Note that cola-machine_5 cannot be expressed in BPA_δ ; the introduction of the empty process enlarges the class of processes we can define.

☞ The choice between serving cola and doing nothing that is expressed by $s + \varepsilon$ in cola-machine_5 says nothing about the *probabilities* of either alternative being chosen. In general, process algebra does not concern itself with quantitative analysis of these probabilities, it merely states *which* choices are possible. However, see BAETEN, BERGSTRA, AND SMOLKA [30] for an extension to process algebra that differs in this respect.

Definition 2.3.4.4 (Semantics of $\text{BPA}_{\delta\varepsilon}$)

The semantics of $\text{BPA}_{\delta\varepsilon}$ are given by the term-deduction system $T(\text{BPA}_{\delta\varepsilon})$ induced by the deduction rules shown in Table 2.8 on page 16 and Table 2.9 on the page before.

Definition 2.3.4.5 (Bisimulation and Bisimulation Model for $\text{BPA}_{\delta\varepsilon}$)

Bisimulation for $\text{BPA}_{\delta\varepsilon}$ and the corresponding bisimulation model for $\text{BPA}_{\delta\varepsilon}$ are defined in the same way as for BPA_ε . Replace “ BPA_ε ” by “ $\text{BPA}_{\delta\varepsilon}$ ” in Definition 2.3.3.5 on the preceding page and Definition 2.3.3.6 on the page before.

Definition 2.3.4.6 (Basic Terms of $BPA_{\delta\varepsilon}$)

We define (δ, ε) -basic terms inductively as follows:

- (i). The constant ε is a (δ, ε) -basic term,
- (ii). if $a \in A_\delta$ and t is a (δ, ε) -basic term, then $a \cdot t$ is a (δ, ε) -basic term,
- (iii). if s and t are (δ, ε) basic terms, then $s + t$ is a (δ, ε) -basic term.

From now on, when we speak of basic terms in the context of $BPA_{\delta\varepsilon}$, we mean (δ, ε) -basic terms.

2.3.5 BPA_δ

In this section, we extend the process algebra BPA with the so-called *immediate deadlock process*. We introduce a new constant, denoted $\dot{\delta}$, that acts very much like the δ we encountered in Section 2.3.2, but differs from it in the fact that it also inhibits progress in other concurrent components of a process. Since we have not yet introduced concurrency (this will be done in Section 2.4), this difference is at the moment academic. However, because all our process algebras build upon basic process algebras, we already introduce it here, in a BPA setting. The extended process algebra that results is called BPA_δ ; it was first introduced by BAETEN AND BERGSTRA [24].

Definition 2.3.5.1 (Signature of BPA_δ)

The signature of BPA_δ consists of the actions $\{a \mid a \in A\}$, the deadlock constant δ , the immediate deadlock constant $\dot{\delta}$, the alternative composition operator $+$, and the sequential composition operator \cdot .

☞ Observe that both δ and $\dot{\delta}$ are present in the signature of BPA_δ . As we will see later, δ can often be expressed in terms of $\dot{\delta}$; therefore it makes little sense to introduce the latter in a setting that does not include the former.

Definition 2.3.5.2 (Axioms of BPA_δ)

The process algebra BPA_δ is axiomatized by the axioms of BPA given in Definition 2.3.1.6 on page 8, Axiom A6A shown in Table 2.6 on page 14, Axiom A7 shown in Table 2.5 on page 14, and Axioms A6ID–A7ID shown in Table 2.10: $BPA_\delta = A1\text{--}A5 + A6A + A7 + A6ID\text{--}A7ID$.

$$\begin{array}{ll} x + \dot{\delta} = x & \text{A6ID} \\ \dot{\delta} \cdot x = \dot{\delta} & \text{A7ID} \end{array}$$

Table 2.10: Additional axioms for BPA_δ .

☞ These axioms need some explaining. First, Axioms A6ID and A7ID express that $\dot{\delta}$ is a proper zero element with respect to the alternative composition, and a left-zero element with respect to the sequential composition. So apparently $\dot{\delta}$ here takes over the rôle δ

used to fulfill in BPA_δ . This means, of course, that δ cannot be a zero element with respect to the alternative composition any longer, since we now have committed ourselves to $BPA_\delta \vdash \delta + \dot{\delta} = \delta$. Hence, we have to drop Axiom A6 from our axioms. We replace it by a weakened version of it, Axiom A6A. In the resulting axiomatization we still have that $BPA_\delta \vdash x + \delta = x$ for process terms x that are not derivably equal to $\dot{\delta}$. So, BPA_δ is a *conservative extension* of BPA_δ : all equalities between closed terms of BPA_δ that hold in BPA_δ also hold in BPA_δ , and vice versa. Viewed in that light, the rôle of δ has not changed at all; we simply have extended our signature.

Definition 2.3.5.3 (Semantics of BPA_δ)

The semantics of BPA_δ are given by the term-deduction system $T(BPA_\delta)$ induced by the deduction rules for BPA_δ given in Definition 2.3.2.6 on page 15 and the deduction rules for the immediate-deadlock predicate shown in Table 2.11.

$$\text{ID}(\dot{\delta}) \quad \frac{\text{ID}(x), \text{ID}(y)}{\text{ID}(x + y)} \quad \frac{\text{ID}(x)}{\text{ID}(x \cdot y)}$$

Table 2.11: Deduction rules for immediate-deadlock predicate.

☞ Note that in the term-deduction system $T(BPA_\delta)$ the constants δ and $\dot{\delta}$ are indistinguishable except for the fact that ID holds for $\dot{\delta}$, while it does not for δ . This difference will be crucial when we introduce concurrency later on.

Definition 2.3.5.4 (Bisimulation for BPA_δ)

Bisimulation for BPA_δ is defined as follows; a binary relation R on closed terms is a bisimulation iff the following transfer conditions hold for all closed terms p and q :

- (i). If $R^S(p, q)$ and $T(BPA_\delta) \models p \xrightarrow{a} p'$, where $a \in A$, then there exists a process term q' such that $T(BPA_\delta) \models q \xrightarrow{a} q'$ and $R^S(p', q')$,
- (ii). if $R^S(p, q)$ and $T(BPA_\delta) \models p \xrightarrow{a} \surd$, where $a \in A$, then $T(BPA_\delta) \models q \xrightarrow{a} \surd$,
- (iii). if $R^S(p, q)$ and $T(BPA_\delta) \models \text{ID}(p)$, then $T(BPA_\delta) \models \text{ID}(q)$.

☞ Bisimulation for BPA_δ is like bisimulation for BPA, but now we also require that the value of the immediate-deadlock predicate is the same for terms that are related.

Definition 2.3.5.5 (Bisimulation Model for BPA_δ)

The bisimulation model for BPA_δ is defined in the same way as for BPA. Replace “BPA” by “ BPA_δ ” in Definition 2.3.1.16 on page 12.

Definition 2.3.5.6 (Basic Terms of BPA_δ)

We define $(\delta, \dot{\delta})$ -basic terms inductively as follows:

- (i). The constant $\dot{\delta}$ is a $(\delta, \dot{\delta})$ -basic term,
- (ii). if $a \in A_\delta$, then a is a $(\delta, \dot{\delta})$ -basic term,

- (iii). if $a \in A_\delta$ and t is a (δ, δ') -basic term, then $a \cdot t$ is a (δ, δ') -basic term,
- (iv). if s and t are (δ, δ') basic terms, then $s + t$ is an (δ, δ') -basic term.

From now on, when we speak of basic terms in the context of BPA_δ , we mean (δ, δ') -basic terms.

☞ Finally, we would like to note that there is no basic process algebra that combines both the empty process and the immediate deadlock process. Although such a process algebra is conceivable, it leads to quite contorted axioms and deduction rules. We will return to this problem in Chapter 6.

2.4 Process Algebras

In this section, we will extend the basic process algebras we introduced in Section 2.3 with the so-called *free merge* operator. This operator allows two processes to execute side-by-side, simultaneously, in an interleaving manner. The resulting class is historically known as the class of *Process Algebras (PA's)* (note the capitals).

2.4.1 PA

We first define the process algebra PA, which is based on BPA. There are two new operators: the *free merge operator*, denoted \parallel , and the *left merge operator*, denoted \llcorner . The latter is an *auxiliary operator*, an operator that in itself is not very useful, but nevertheless is needed in order to define other operators, in this case the free merge (see MOLLER [143]).

Definition 2.4.1.1 (Signature of PA)

The signature of PA consists of the *actions* $\{a \mid a \in A\}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *free merge operator* \parallel , and the *left merge operator* \llcorner .

Definition 2.4.1.2 (Axioms of PA)

The process algebra PA is axiomatized by the axioms of BPA given in Definition 2.3.1.6 on page 8 and Axioms M1–M4 shown in Table 2.12: PA = A1–A5 + M1–M4.

$x \parallel y = x \llcorner y + y \llcorner x$	M1
$a \llcorner x = a \cdot x$	M2
$a \cdot x \llcorner y = a \cdot (x \parallel y)$	M3
$(x + y) \llcorner z = x \llcorner z + y \llcorner z$	M4

Table 2.12: Additional axioms for PA.

☞ In the process term $x \parallel y$, the free merge operator executes x and y in parallel, generating all possible interleavings of the two. In a similar manner, in the process term

$x \parallel y$, the left merge operator executes x and y in parallel, with the restriction that the first action *must necessarily come from* x . Now we can explain the Axiom M1: it expresses that the parallel execution of x and y starts either with x , or with y . Axiom M2 expresses that if we execute a and x in parallel, starting with a , this is the same as executing a followed by x . Similarly, Axiom M3 expresses that if we execute $a \cdot x$ and y in parallel, starting with $a \cdot x$, this is the same as executing a followed by the execution of x and y in parallel. Finally, Axiom M4 expresses that the left merge right-distributes over the alternative composition. The reason that we do not have a corresponding left-distributivity is, as with the sequential composition, that if we would, our semantics would collapse to a trace semantics. Note that distributivity does not hold for the free merge, which is why the left merge is needed for a finite axiomatization of the free merge (see also MOLLER [143]).

Not obvious on first sight, but still true, is the fact that these axioms lead to a free merge that is *associative*: for all closed terms x , y , and z we have $\text{PA} \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$. This means we can leave out the parentheses in such expressions. Furthermore, the free merge is *commutative*: for all terms x and y we have $\text{PA} \vdash x \parallel y = y \parallel x$.

Example 2.4.1.3 (Calculating in PA)

Let $s \equiv a \cdot a$ and $t \equiv b \cdot b$. Then we have:

$$\begin{aligned}
 \text{PA} \vdash s \parallel t &= a \cdot a \parallel b \cdot b \\
 &= a \cdot a \parallel b \cdot b + b \cdot b \parallel a \cdot a \\
 &= a \cdot (a \parallel b \cdot b) + b \cdot (b \parallel a \cdot a) \\
 &= a \cdot (a \parallel b \cdot b + b \cdot b \parallel a) + b \cdot (b \parallel a \cdot a + a \cdot a \parallel b) \\
 &= a \cdot (a \cdot b \cdot b + b \cdot (b \parallel a)) + b \cdot (b \cdot a \cdot a + a \cdot (a \parallel b)) \\
 &= a \cdot (a \cdot b \cdot b + b \cdot (b \parallel a + a \parallel b)) + b \cdot (b \cdot a \cdot a + a \cdot (a \parallel b + b \parallel a)) \\
 &= a \cdot (a \cdot b \cdot b + b \cdot (b \cdot a + a \cdot b)) + b \cdot (b \cdot a \cdot a + a \cdot (a \cdot b + b \cdot a))
 \end{aligned}$$

☞ As should be obvious from this, fully expanding a concurrent process leads to an exponential explosion (see pages 73–75 of BAETEN AND WEIJLAND [38] for an expansion of $a \cdot a \cdot a \parallel b \cdot b \cdot b \parallel c \cdot c \cdot c$, if you dare ...). Luckily, often we do not have to expand such a process, because there are algebraic properties (so-called *expansion theorems*) we can exploit to keep expansion to an absolute minimum. In Chapter 7 we will see an example of this.

Example 2.4.1.4 (Cola Machine in PA)

Suppose the 10 cent coin gets withdrawn because it costs only 6 cents to counterfeit it. As a temporary measure, the central bank introduces new coins of denominations 1, 2, 3, and 4 cent (these are just as easy to counterfeit, but not at a profit). We need to adapt cola-machine_3 , so that it accepts 15 cents in the form of one coin of each denomination. Using the free merge we can do this quite easily, as follows:

$$\text{cola-machine}_6 = (i_1 \parallel i_2 \parallel i_3 \parallel i_4 \parallel i_5) \cdot p \cdot s$$

Had we been forced to do this in BPA, the specification of cola-machine_6 would have contained $5! = 120$ subterms, in order to specify all possible ways to insert the five coins.

Definition 2.4.1.5 (Semantics of PA)

The semantics of PA are given by the term-deduction system $T(\text{PA})$ induced by the deduction rules for BPA given in Definition 2.3.1.11 on page 10 and the deduction rules for the free merge shown in Table 2.13.

$$\begin{array}{ccc}
 \frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} & \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'} & \frac{x \xrightarrow{a} x'}{x \llcorner y \xrightarrow{a} x' \parallel y} \\
 \\
 \frac{x \xrightarrow{a} \surd}{x \parallel y \xrightarrow{a} y} & \frac{y \xrightarrow{a} \surd}{x \parallel y \xrightarrow{a} x} & \frac{x \xrightarrow{a} \surd}{x \llcorner y \xrightarrow{a} y}
 \end{array}$$

Table 2.13: Deduction rules for free merge.

☞ The deduction rules speak for themselves. Note that a left merge turns into a free merge after it has done an action. Furthermore, after one component of a free merge has successfully terminated, the free merge disappears and the other component continues on its own.

Definition 2.4.1.6 (Bisimulation and Bisimulation Model for PA)

Bisimulation for PA and the corresponding bisimulation model for PA are defined in the same way as for BPA. Replace “BPA” by “PA” in Definition 2.3.1.13 on page 11 and Definition 2.3.1.16 on page 12.

Definition 2.4.1.7 (Basic Terms of PA)

When we speak of basic terms in the context of PA, we mean deadlock-free basic terms as defined in Definition 2.3.1.17 on page 13.

2.4.2 PA_δ

In this section, we add the deadlock constant δ to PA. As this is entirely straightforward, we give hardly any comments. The resulting process algebra is called PA_δ .

Definition 2.4.2.1 (Signature of PA_δ)

The signature of PA_δ consists of the *actions* $\{a \mid a \in A\}$, the *deadlock constant* δ , the *alternative composition operator* $+$, the *sequential composition operator* $;$, the *free merge operator* \parallel , and the *left merge operator* \llcorner .

Definition 2.4.2.2 (Axioms of PA_δ)

The process algebra PA_δ is axiomatized by the axioms of PA given in Definition 2.4.1.2 on page 21 and Axioms A6–A7 shown in Table 2.5 on page 14: $\text{PA}_\delta = \text{A1–A7} + \text{M1–M4}$.

☞ Note that the a in Axioms M2 and M3 now also can take the value δ , as δ is now part of the signature.

Definition 2.4.2.3 (Semantics of PA_δ)

The semantics of PA_δ are given by the term-deduction system $T(PA_\delta)$ induced by the deduction rules for BPA_δ given in Definition 2.3.2.6 on page 15 and the deduction rules for the free merge shown in Table 2.13 on the page before.

Definition 2.4.2.4 (Bisimulation and Bisimulation Model for PA_δ)

Bisimulation for PA_δ and the corresponding bisimulation model for PA_δ are defined in the same way as for BPA. Replace “BPA” by “ PA_δ ” in Definition 2.3.1.13 on page 11 and Definition 2.3.1.16 on page 12.

Definition 2.4.2.5 (Basic Terms of PA_δ)

When we speak of basic terms in the context of PA_δ , we mean standard basic terms as defined in Definition 2.3.2.8 on page 15.

2.4.3 PA_ε

In this section, we add the empty process constant ε to PA. The resulting process algebra is called PA_ε .

Definition 2.4.3.1 (Signature of PA_ε)

The signature of PA_ε consists of the *actions* $\{a \mid a \in A\}$, the *deadlock constant* δ , the *empty process constant* ε , the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *free merge operator* \parallel , and the *left merge operator* \ll .

Definition 2.4.3.2 (Axioms of PA_ε)

The process algebra PA_ε is axiomatized by the axioms of PA_δ given in Definition 2.4.2.2 on the page before *minus* Axiom M2, *plus* Axioms A8–A9 shown in Table 2.7 on page 16, and Axioms ME1–ME3 shown in Table 2.14: $PA_\varepsilon = A1\text{--}A9 + M1 + M3\text{--}M4 + ME1\text{--}ME3$.

$\varepsilon \ll \varepsilon = \varepsilon$	ME1
$\varepsilon \ll a \cdot x = \delta$	ME2
$\varepsilon \ll (x + y) = \varepsilon \ll x + \varepsilon \ll y$	ME3

Table 2.14: Additional axioms for PA_ε .

☞ The additional axioms for PA_ε are solely needed to assure that $x \parallel y$ contains an ε -summand iff both x and y contain one; all axioms of PA_δ remain valid in PA_ε . Note that δ can be defined in terms of the ε and the \ll : we have $PA_\varepsilon \vdash \varepsilon \ll a \cdot a = \delta$, so it is logical to include δ in the signature.

Axiom ME1 expresses that if both sides of a left merge have the option to terminate, then the left merge has that option also. Axiom ME2 may be more surprising. The rationale behind it is that intuitively the left merge can only commence executing its right argument after it has done at least one action of its left argument, and since that is impossible when the left argument is ε , the whole expression is equal to deadlock. Axiom ME3 expresses that $\varepsilon \ll$ left-distributes over the alternative composition. This is needed,

along with the right-distributivity of Axiom M4, to propagate ε -summands down, to a level where either Axiom ME1 or Axiom ME2 can be applied. Note that $\varepsilon \ll x$ can take the values ε and δ : for all closed terms x we either have $\text{PA}_\varepsilon \vdash \varepsilon \ll x = \varepsilon$ (namely, for those x that contain an ε -summand, i.e., x such that $\text{PA}_\varepsilon \vdash x + \varepsilon = x$), or $\text{PA}_\varepsilon \vdash \varepsilon \ll x = \delta$ (for all other x). So, Axiom ME3 does in no way jeopardize our bisimulation semantics, as for example Axiom LD from Table 2.2 on page 9 would.

Then, note that Axiom M2 is not present anymore in PA_ε . This is because, for closed terms, it has become derivable: substituting ε for x in Axiom M3 gives Axiom M2. In general, we often encounter axioms that are “twins”, i.e., one axiom handles the case where some subexpression is a , the other one where it is $a \cdot x$. In a setting where we have the empty process, we can drop the former, as it is a special case of the latter (namely the case $x = \varepsilon$). A similar observation holds for the deduction rules. In this way the empty process often makes our axiomatizations and semantics simpler, or at least shorter.

Finally, these axioms lead to a free merge that had ε as a unit element: for all closed terms x we have $\text{PA}_\varepsilon \vdash \varepsilon \parallel x = x$ and $\text{PA}_\varepsilon \vdash x \parallel \varepsilon = x$. Furthermore, ε is a right-unit element of the left merge: for all closed terms x we have $\text{PA}_\varepsilon \vdash x \ll \varepsilon = x$. It is *not* a left-unit element, as we e.g. have $\text{PA} \vdash \varepsilon \ll a = \delta$.

Definition 2.4.3.3 (Semantics of PA_ε)

The semantics of PA_ε are given by the term-deduction system $T(\text{PA}_\varepsilon)$ induced by the deduction rules for $\text{BPA}_{\delta\varepsilon}$ given in Definition 2.3.4.4 on page 18 and the deduction rules for the free merge shown in Table 2.15.

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} \quad \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'} \quad \frac{x \xrightarrow{a} x'}{x \ll y \xrightarrow{a} x' \parallel y}$$

$$\frac{x \downarrow, y \downarrow}{(x \parallel y) \downarrow} \quad \frac{x \downarrow, y \downarrow}{(x \ll y) \downarrow}$$

Table 2.15: Deduction rules for free merge with empty process.

☞ The deduction rules for the free merge with the empty process are as expected. Note that $x \parallel y$ and $x \ll y$ can terminate iff both x and y can.

Definition 2.4.3.4 (Bisimulation and Bisimulation Model for PA_ε)

Bisimulation for PA_ε and the corresponding bisimulation model for PA_ε are defined in the same way as for BPA_ε . Replace “ BPA_ε ” by “ PA_ε ” in Definition 2.3.3.5 on page 17 and Definition 2.3.3.6 on page 17.

Definition 2.4.3.5 (Basic Terms of PA_ε)

When we speak of basic terms in the context of PA_ε , we mean (δ, ε) -basic terms as defined in Definition 2.3.4.6 on page 19.

Remark 2.4.3.6 (Comparison with the Literature)

Our definitions for PA_ε are based on those given by VRANCKEN [197]. A different approach is taken by BAETEN AND WEIJLAND [38], who do not encode the behavior of the empty process with respect to the free merge within the left merge as we do. Instead, they define an auxiliary operator, the *termination operator*, denoted \surd , such that $\surd(x)$ is equal to ε if x contains an ε -summand, and δ otherwise (this can be expressed in our process algebra by $\varepsilon \parallel x$). Both approaches are equally valid, and our preference for the first method is probably a matter of taste.

2.4.4 PA_δ

In this section, we combine the free merge with the $\dot{\delta}$ of BPA_δ treated in Section 2.3.5. The resulting process algebra is called PA_δ . The difference between δ and $\dot{\delta}$, that was somewhat academic in the setting of BPA_δ , becomes very clear when the free merge is present.

Definition 2.4.4.1 (Signature of PA_δ)

The signature of PA_δ consists of the *actions* $\{a \mid a \in A\}$, the *deadlock constant* δ , the *immediate deadlock constant* $\dot{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *free merge operator* \parallel , and the *left merge operator* \llcorner .

Definition 2.4.4.2 (Axioms of PA_δ)

The process algebra PA_δ is axiomatized by the axioms of BPA_δ given in Definition 2.3.5.2 on page 19, Axioms M1 and M4 shown in Table 2.12 on page 21, and Axioms M2ID–M3ID and MID1–MID2 shown in Table 2.16: $PA_\delta = A1\text{--}A5 + A6A + A7 + A6ID\text{--}A7ID + M1 + M2ID\text{--}M3ID + M4 + MID1\text{--}MID2$.

$a \llcorner (x + \delta) = a \cdot (x + \delta)$	M2ID
$a \cdot x \llcorner (y + \delta) = a \cdot (x \parallel (y + \delta))$	M3ID
$x \llcorner \dot{\delta} = \dot{\delta}$	MID1
$\dot{\delta} \llcorner x = \dot{\delta}$	MID2

Table 2.16: Additional axioms for PA_δ .

☞ These axioms need quite some explaining. First of all, in a context with the free merge, the immediate deadlock should inhibit all further progress if it is the only summand that is left on one of the sides. So, whereas we have, as usual, that $PA_\delta \vdash a \parallel \delta = a \cdot \delta$, for the immediate deadlock we should have $PA_\delta \vdash a \parallel \dot{\delta} = \dot{\delta}$. In other words, the immediate deadlock represents a *catastrophic stop*, leading to a complete halt of the entire process. That is, of course, if no other summands are present; $PA_\delta \vdash a \parallel (b + \dot{\delta}) = a \cdot b + b \cdot a$, as the immediate deadlock can be avoided in this case.

This intuition immediately disqualifies Axioms M2 and M3 from inclusion in PA_δ , as they “do not check” the right argument of the left merge for immediate deadlock before executing an action from the left argument. We weaken them to Axioms M2ID and M3ID,

respectively. For basic terms x such that $\text{BPA}_\delta \not\vdash x = \dot{\delta}$, we have $\text{BPA}_\delta \vdash x + \delta = x$, and therefore for such terms Axioms M2ID and M3ID are just as strong as Axioms M2 and M3 were. However, terms of the form $x \parallel \dot{\delta}$ cannot be rewritten by Axioms M2ID and M3ID. For this purpose we define Axioms MID1 and MID2. They make sure that terms that just contain an immediate deadlock on either side of the left merge, reduce to immediate deadlock themselves.

The above construction will return several times, so we paraphrase it once more. The expression $x + \delta$ should be intuitively interpreted as “no $\dot{\delta}$ allowed here”. Axioms that contain $x + \delta$ do so as a systematic trick to weaken them, in order to prevent their application to a lone $\dot{\delta}$ summand.

Example 2.4.4.3 (Cola Machine in PA_δ)

We extend cola-machine_3 from page 10 in such a way that it will automatically shut down when it starts overheating. Let o denote the action that results when some temperature sensor measures too high a temperature. We get the following specification:

$$\text{cola-machine}_7 = (i_{10} \cdot i_5 + i_5 \cdot (i_{10} + i_5 \cdot i_5)) \cdot p \cdot s \parallel o \cdot \dot{\delta}$$

So, cola-machine_7 behaves like cola-machine_3 as long as no overheating occurs. If overheating occurs, i.e., if o gets executed, cola-machine_7 comes to a halt immediately.

Definition 2.4.4.4 (Semantics of PA_δ)

The semantics of PA_δ are given by the term-deduction system $T(\text{PA}_\delta)$ induced by the deduction rules for BPA_δ given in Definition 2.3.5.3 on page 20 and the deduction rules for the free merge shown in Table 2.17.

$\frac{x \xrightarrow{a} x', \neg \text{ID}(y)}{x \parallel y \xrightarrow{a} x' \parallel y}$	$\frac{y \xrightarrow{a} y', \neg \text{ID}(x)}{x \parallel y \xrightarrow{a} x \parallel y'}$	$\frac{x \xrightarrow{a} x', \neg \text{ID}(y)}{x \parallel\!\!\! \parallel y \xrightarrow{a} x' \parallel y}$
$\frac{x \xrightarrow{a} \surd, \neg \text{ID}(y)}{x \parallel y \xrightarrow{a} y}$	$\frac{y \xrightarrow{a} \surd, \neg \text{ID}(x)}{x \parallel y \xrightarrow{a} x}$	$\frac{x \xrightarrow{a} \surd, \neg \text{ID}(y)}{x \parallel\!\!\! \parallel y \xrightarrow{a} y}$
$\frac{\text{ID}(x)}{\text{ID}(x \parallel y)}$	$\frac{\text{ID}(y)}{\text{ID}(x \parallel y)}$	
$\frac{\text{ID}(x)}{\text{ID}(x \parallel\!\!\! \parallel y)}$	$\frac{\text{ID}(y)}{\text{ID}(x \parallel\!\!\! \parallel y)}$	

Table 2.17: Deduction rules for free merge with immediate deadlock.

☞ The deduction rules of PA_δ define the immediate-deadlock predicate $\text{ID}(x)$ that is true iff $\text{PA}_\delta \vdash x = \dot{\delta}$. Using this predicate, it then becomes easy to define the deduction rules for the free merge and the left merge: they are very much like those we defined for PA

(see Table 2.13 on page 23), with the exception that we now require that one side of the free merge or left merge can only execute an action if the immediate-deadlock predicate does not hold for the other side. An additional four rules are needed to ensure that the immediate-deadlock predicate is defined on all process terms of PA_δ .

Definition 2.4.4.5 (Bisimulation and Bisimulation Model for PA_δ)

Bisimulation for PA_δ and the corresponding bisimulation model for PA_δ are defined in the same way as for BPA_δ . Replace “ BPA_δ ” by “ PA_δ ” in Definition 2.3.5.4 on page 20 and Definition 2.3.5.5 on page 20.

Definition 2.4.4.6 (Basic Terms of PA_δ)

When we speak of basic terms in the context of PA_δ , we mean (δ, δ^\dagger) basic terms as defined in Definition 2.3.5.6 on page 20.

2.5 Algebras of Communicating Processes

In this section, we will extend the free merge of the Process Algebras we introduced in Section 2.4 to the *merge* operator. This operator allows two processes to execute side-by-side, simultaneously, in an interleaving manner (as did the free merge), but in addition allows these processes to *communicate* with each other or *synchronize*. This class is historically known as the class of *Algebras of Communicating Processes (ACP's)*. Combined with the class of Process Algebras, it is known as the class of *Concurrent Process Algebras*.

2.5.1 ACP

We first define the process algebra ACP, which is based on BPA_δ (as δ can be defined in terms of the merge, we are forced to include it in the signature). There are three merge-like operators: the *merge operator*, denoted \parallel , which is an extension of the free-merge, the *left merge operator*, denoted \llcorner , and the *communication merge operator*, denoted \mid . Finally, there is a completely new operator: the *encapsulation operator*, denoted ∂_H , which provides a means for prohibiting certain actions from occurring.

Definition 2.5.1.1 (Communication Function)

For this section, and all sections to come, we presume the existence of a fixed, commutative, associative, complete function $\gamma : A_\delta \times A_\delta \rightarrow A_\delta$, that can be considered a parameter of the respective process algebra. The function γ has to be *strict* in the sense that it should always evaluate to δ when one or both of its arguments is δ .

☞ The function γ is meant to define what happens if two action a and b execute simultaneously on both sides of the merge (this simultaneous execution is also called *communication* or *synchronization*). If, for example, we have that $\gamma(a, b) = c$, then this should be interpreted as “if we execute a and b simultaneously, together they result in the execution of the action c ”. If $\gamma(a, b) = \delta$, this should be interpreted as “ a and b cannot execute simultaneously”.

The function γ should be commutative and associative because in a complex process more than two actions may be executed simultaneously, and in that case we do not want

the outcome to depend on the order of these actions. It should be strict because we do not want δ to communicate with any action (this would have extremely undesirable results).

Note that our definition of γ is different from the one given by BAETEN AND WEIJLAND [38] in the sense that in their definition γ is undefined where we have $\gamma(a, b) = \delta$. We find their definition less convenient, as it leads to unnecessary case distinctions in proofs and the duplication of some axioms.

Definition 2.5.1.2 (Signature of ACP)

The signature of ACP consists of the actions $\{a \mid a \in A\}$, the *deadlock constant* δ , the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *merge operator* \parallel , the *left merge operator* \llcorner , the *communication merge operator* \mid , and the *encapsulation operator* ∂_H .

Definition 2.5.1.3 (Axioms of ACP)

The process algebra ACP is axiomatized by the axioms of PA_δ given in Definition 2.4.2.2 on page 23 *minus* Axiom M1, *plus* Axioms CM1–CM6, CF, and D1–D4 shown in Table 2.18: $ACP = A1\text{--}A7 + M2\text{--}M4 + CM1\text{--}CM6 + CF$.

$x \parallel y = x \llcorner y + y \llcorner x + x \mid y$	CM1
$a \mid b \cdot x = (a \mid b) \cdot x$	CM2
$a \cdot x \mid b = (a \mid b) \cdot x$	CM3
$a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel y)$	CM4
$(x + y) \mid z = x \mid z + y \mid z$	CM5
$x \mid (y + z) = x \mid y + x \mid z$	CM6
$a \mid b = c \quad \text{if } \gamma(a, b) = c$	CF
$\partial_H(a) = a \quad \text{if } a \notin H$	D1
$\partial_H(a) = \delta \quad \text{if } a \in H$	D2
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D4

Table 2.18: Additional axioms for ACP.

☞ We go through the axioms one by one. First, Axiom M1 is replaced by Axiom CM1, expressing that $x \parallel y$ either executes an action from x , or an action from y , or combines actions from x and y in a communication. The last option is represented by $x \mid y$, the process that combines an action from x with an action from y , and then continues as the merge of the remainders. Axioms CM2, CM3, and CM4 express that if in $x \mid y$ both x and y only have one action at the head, these two action are necessarily the ones that communicate with each other. Axioms CM5 and CM6 express that the communication merge both left-distributes and right-distributes over the alternative composition. Axiom CF expresses that the communication merge of two actions a and b is simply the outcome

of $\gamma(a, b)$. As in PA, these axioms lead to a merge that is *associative* and *commutative* for closed terms. Note, however, that the free merge of PA is commutative even for open terms, while the merge of ACP is only commutative for closed terms.

Then the encapsulation operator. The intuition behind it is that given a certain set of symbols $H \subseteq A$, the process $\partial_H(x)$ should behave as x , but refrain from any actions a such that $a \in H$. Or, in other words, $\partial_H(x)$ should rename all such actions to δ . Axioms D1 and D2 express this: $\partial_H(a)$ should be a if $a \notin H$ and δ if $a \in H$ (note that this also defines $\partial_H(\delta) = \delta$, as a ranges over A_δ in axioms). Finally, Axioms D3 and D4 say that the encapsulation distributes over the alternative and sequential composition. In this way the encapsulation propagates down to the action level, where either Axiom D1 or D2 can be applied. Notice that the encapsulation does not distribute over the merge (see Example 2.5.1.4).

Example 2.5.1.4 (Calculating in ACP)

Let $s \equiv a \cdot a$ and $t \equiv b \cdot b$, and suppose that $\gamma(a, b) = c$. Then we have:

$$\begin{aligned}
\text{ACP} \vdash s \parallel t &= a \cdot a \parallel b \cdot b \\
&= a \cdot a \parallel b \cdot b + b \cdot b \parallel a \cdot a + a \cdot a \mid b \cdot b \\
&= a \cdot (a \parallel b \cdot b) + b \cdot (b \parallel a \cdot a) + (a \mid b) \cdot (a \parallel b) \\
&= a \cdot (a \parallel b \cdot b + b \cdot b \parallel a + a \mid b \cdot b) + \\
&\quad b \cdot (b \parallel a \cdot a + a \cdot a \parallel b + b \mid a \cdot a) + \\
&\quad c \cdot (a \parallel b + b \parallel a + a \mid b) \\
&= a \cdot (a \cdot b \cdot b + b \cdot (b \parallel a) + (a \mid b) \cdot b) + \\
&\quad b \cdot (b \cdot a \cdot a + a \cdot (a \parallel b) + (b \mid a) \cdot a) + \\
&\quad c \cdot (a \cdot b + b \cdot a + c) \\
&= a \cdot (a \cdot b \cdot b + b \cdot (b \parallel a + a \parallel b + b \mid a) + c \cdot b) + \\
&\quad b \cdot (b \cdot a \cdot a + a \cdot (a \parallel b + b \parallel a + a \mid b) + c \cdot a) + \\
&\quad c \cdot (a \cdot b + b \cdot a + c) \\
&= a \cdot (a \cdot b \cdot b + b \cdot (b \cdot a + a \cdot b + c) + c \cdot b) + \\
&\quad b \cdot (b \cdot a \cdot a + a \cdot (a \cdot b + b \cdot a + c) + c \cdot a) + \\
&\quad c \cdot (a \cdot b + b \cdot a + c)
\end{aligned}$$

If we now define $H = \{a, b\}$, then we get:

$$\begin{aligned}
\text{ACP} \vdash \partial_H(s \parallel t) &= \dots \\
&= \delta \cdot (\delta \cdot \delta \cdot \delta + \delta \cdot (\delta \cdot \delta + \delta \cdot \delta + c) + c \cdot \delta) + \\
&\quad \delta \cdot (\delta \cdot \delta \cdot \delta + \delta \cdot (\delta \cdot \delta + \delta \cdot \delta + c) + c \cdot \delta) + \\
&\quad c \cdot (\delta \cdot \delta + \delta \cdot \delta + c) \\
&= \dots \\
&= c \cdot c
\end{aligned}$$

As we can see, the encapsulation effectively prohibits the actions a and b from occurring, and hence they are forced to communicate with each other, leading to the execution of $c \cdot c$.

Example 2.5.1.5 (Cola Machine in ACP)

Using ACP, we can now define cola-machine_3 from page 10 in a simpler way. Define the communication function γ as follows: $\gamma(i_5, i_5) = i_{10}$, and δ everywhere else. Then define:

$$\text{cola-machine}_8 = (i_5 \parallel i_5 \parallel i_5) \cdot p \cdot s$$

So, cola-machine_8 wants three 5 cent coins, but it is also willing to accept a 10 cent coin as if it were two 5 cent coins. Note that we have $\text{ACP} \vdash \text{cola-machine}_3 = \text{cola-machine}_8$.

Suppose a 15 cent coin also exists, and that for some reason we do not want to accept 10 cent coins anymore. Then we can extend our communication function such that $\gamma(i_5, i_{10}) = \gamma(i_{10}, i_5) = i_{15}$, let $H = \{i_{10}\}$, and define:

$$\text{cola-machine}_9 = \partial_H(i_5 \parallel i_5 \parallel i_5) \cdot p \cdot s$$

Now cola-machine_9 accepts either three 5 cent coins, or one 15 cent coin, but nothing else. Note that the action i_{15} results from the simultaneous execution of all three i_5 actions.

Definition 2.5.1.6 (Semantics of ACP)

The semantics of ACP are given by the term-deduction system $T(\text{ACP})$ induced by the deduction rules for PA_δ given in Definition 2.4.2.3 on page 24, the additional deduction rules for the merge shown in Table 2.19, and the deduction rules for the encapsulation shown in Table 2.20 on the following page.

$\frac{x \xrightarrow{a} x', y \xrightarrow{b} y', \gamma(a, b) = c}{x \parallel y \xrightarrow{c} x' \parallel y'}$	$\frac{x \xrightarrow{a} x', y \xrightarrow{b} y', \gamma(a, b) = c}{x \mid y \xrightarrow{c} x' \parallel y'}$
$\frac{x \xrightarrow{a} x', y \xrightarrow{b} \surd, \gamma(a, b) = c}{x \parallel y \xrightarrow{c} x'}$	$\frac{x \xrightarrow{a} x', y \xrightarrow{b} \surd, \gamma(a, b) = c}{x \mid y \xrightarrow{c} x'}$
$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} y', \gamma(a, b) = c}{x \parallel y \xrightarrow{c} y'}$	$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} y', \gamma(a, b) = c}{x \mid y \xrightarrow{c} y'}$
$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} \surd, \gamma(a, b) = c}{x \parallel y \xrightarrow{c} \surd}$	$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} \surd, \gamma(a, b) = c}{x \mid y \xrightarrow{c} \surd}$

Table 2.19: Additional deduction rules for merge.

☞ The additional deduction rules for the merge are straightforward, but have to be defined in quadruplicate because both x and y may or may not have the possibility to do a terminating step. The deduction rules for the encapsulation are even simpler: $\partial_H(x)$ behaves like x , provided it only does steps that are not prohibited.

$$\frac{x \xrightarrow{a} x', a \notin H}{\partial_H(x) \xrightarrow{a} \partial_H(x')} \quad \frac{x \xrightarrow{a} \surd, a \notin H}{\partial_H(x) \xrightarrow{a} \surd}$$

Table 2.20: Deduction rules for encapsulation.

Definition 2.5.1.7 (Bisimulation and Bisimulation Model for ACP)

Bisimulation for ACP and the corresponding bisimulation model for ACP are defined in the same way as for BPA. Replace “BPA” by “ACP” in Definition 2.3.1.13 on page 11 and Definition 2.3.1.16 on page 12.

Definition 2.5.1.8 (Basic Terms of ACP)

When we speak of basic terms in the context of ACP, we mean standard basic terms as defined in Definition 2.3.2.8 on page 15.

2.5.2 ACP_ε

In this section, we extend PA_ε to include the merge operator. The resulting process algebra is called ACP_ε.

Definition 2.5.2.1 (Signature of ACP_ε)

The signature of ACP_ε consists of the *actions* {a | a ∈ A}, the *deadlock constant* δ, the *empty process constant* ε, the *alternative composition operator* +, the *sequential composition operator* ·, the *merge operator* ||, the *left merge operator* ⌋, the *communication merge operator* |, and the *encapsulation operator* ∂_H.

Definition 2.5.2.2 (Axioms of ACP_ε)

The process algebra ACP_ε is axiomatized by the axioms of PA_ε given in Definition 2.4.3.2 on page 24 *minus* Axiom M1, *plus* Axioms CM1, CM4-CM6, CF, and D1-D4 shown in Table 2.18 on page 29, and Axioms ME4-ME5 and D5 shown in Table 2.21: ACP_ε = A1-A9 + M3-M4 + ME1-ME5 + CM1 + CM4-CM6 + CF + D1-D5.

$$\begin{array}{ll} x | \varepsilon = \delta & \text{ME4} \\ \varepsilon | x = \delta & \text{ME5} \\ \partial_H(\varepsilon) = \varepsilon & \text{D5} \end{array}$$

Table 2.21: Additional axioms for ACP_ε.

☞ The axioms of ACP_ε are entirely straightforward. Because we have the empty process, Axioms M2, CM2, and CM3 become derivable for closed terms, and can be dropped. We introduce Axioms ME4 and ME5 to express that the empty process does not communicate in any way, and Axiom D5 to express that the empty process is immune to encapsulation.

Note that, as in PA_ε , the empty process is a unit element of the merge, and a right-unit element of the left merge.

Definition 2.5.2.3 (Semantics of ACP_ε)

The semantics of ACP_ε are given by the term-deduction system $T(ACP_\varepsilon)$ induced by the deduction rules for PA_ε given in Definition 2.4.3.3 on page 25, the additional deduction rules for the merge shown in Table 2.22, and the deduction rules for the encapsulation shown in Table 2.23.

$$\frac{x \xrightarrow{a} x', y \xrightarrow{b} y', \gamma(a, b) = c}{x \parallel y \xrightarrow{c} x' \parallel y'} \quad \frac{x \xrightarrow{a} x', y \xrightarrow{b} y', \gamma(a, b) = c}{x \mid y \xrightarrow{c} x' \parallel y'}$$

Table 2.22: Additional deduction rules for merge with empty process.

$$\frac{x \xrightarrow{a} x', a \notin H}{\partial_H(x) \xrightarrow{a} \partial_H(x')} \quad \frac{x \downarrow}{\partial_H(x) \downarrow}$$

Table 2.23: Deduction rules for encapsulation with empty process.

☞ The deduction rules for the merge with the empty process are similar to those without the empty process (shown in Table 2.19 on page 31). Because we have the empty process, quadruplication is not necessary. Again we see that the introduction of the empty process often simplifies matters.

The deduction rules for the encapsulation with the empty process are very similar to those of Table 2.20 on the preceding page; $\partial_H(x)$ behaves like x , provided it only does steps that are not forbidden. In particular, $\partial_H(x)$ has the option to terminate iff x has the option to terminate.

Remark 2.5.2.4 (Deduction rules for ACP_ε)

Note that there is no deduction rule of the form:

$$\frac{x \downarrow, y \downarrow}{(x \mid y) \downarrow}$$

because the process $\varepsilon \mid \varepsilon$ cannot successfully terminate: we have that $ACP_\varepsilon \vdash \varepsilon \mid \varepsilon = \delta$, and $T(ACP_\varepsilon) \models \delta \downarrow$, hence $T(ACP_\varepsilon) \models (\varepsilon \mid \varepsilon) \downarrow$.

Definition 2.5.2.5 (Bisimulation and Bisimulation Model for ACP_ε)

Bisimulation for ACP_ε and the corresponding bisimulation model for ACP_ε are defined in the same way as for BPA_ε . Replace “ BPA_ε ” by “ ACP_ε ” in Definition 2.3.3.5 on page 17 and Definition 2.3.3.6 on page 17.

Definition 2.5.2.6 (Basic Terms of ACP_ε)

When we speak of basic terms in the context of ACP_ε , we mean (δ, ε) -basic terms as defined in Definition 2.3.4.6 on page 19.

2.5.3 ACP_δ

In this section, we extend PA_δ to include the merge operator. The resulting process algebra is called ACP_δ .

Definition 2.5.3.1 (Signature of ACP_δ)

The signature of ACP_δ consists of the *actions* $\{a \mid a \in A\}$, the *deadlock constant* δ , the *immediate deadlock constant* $\dot{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *merge operator* \parallel , the *left merge operator* \llcorner , the *communication merge operator* \lrcorner , and the *encapsulation operator* ∂_H .

Definition 2.5.3.2 (Axioms of ACP_δ)

The process algebra ACP_δ is axiomatized by the axioms of PA_δ given in Definition 2.4.4.2 on page 26 *minus* Axiom M1, *plus* Axioms CM1–CM6, CF, and D1–D4 shown in Table 2.18 on page 29, and Axioms MID3–MID4 and D6 shown in Table 2.24: $ACP_\delta = A1\text{--}A5 + A6A + A7 + A6ID\text{--}A7ID + M2ID\text{--}M3ID + M4 + CM1\text{--}CM6 + CF + MID1\text{--}MID4 + D1\text{--}D4 + D6$.

$x \mid \dot{\delta} = \dot{\delta}$	MID3
$\dot{\delta} \mid x = \dot{\delta}$	MID4
$\partial_H(\dot{\delta}) = \dot{\delta}$	D6

Table 2.24: Additional axioms for ACP_δ .

☞ The axioms of ACP_δ are a straightforward combination of axioms we have already encountered. We only have to add Axioms MID3 and MID4 to express that the immediate deadlock prevents any further progress when it is the left or right argument of the communicating merge, as was also the case for the left merge (compare Axioms MID1 and MID2 on page 26), and Axiom D6 to express that the immediate deadlock is immune to encapsulation.

Definition 2.5.3.3 (Semantics of ACP_δ)

The semantics of ACP_δ are given by the term-deduction system $T(ACP_\delta)$ induced by the deduction rules for PA_δ given in Definition 2.4.4.4 on page 27, the additional deduction rules for the merge shown in Table 2.19 on page 31, and the additional deduction rules for the immediate-deadlock predicate shown in Table 2.25 on the facing page and Table 2.26 on the next page.

☞ Also the deduction rules are a straightforward combination of rules we have already encountered. We only need to add three additional rules to define the immediate-deadlock predicate on $x \mid y$ and $\partial_H(x)$.

$$\frac{\text{ID}(x)}{\text{ID}(x \mid y)} \quad \frac{\text{ID}(y)}{\text{ID}(x \mid y)}$$

Table 2.25: Additional deduction rules for communication merge with immediate-deadlock predicate.

$$\frac{\text{ID}(x)}{\text{ID}(\partial_H(x))}$$

Table 2.26: Additional deduction rules for encapsulation with immediate-deadlock predicate.

Definition 2.5.3.4 (Bisimulation and Bisimulation Model for ACP_δ)

Bisimulation for ACP_δ and the corresponding bisimulation model for ACP_δ are defined in the same way as for BPA_δ . Replace “ BPA_δ ” by “ ACP_δ ” in Definition 2.3.5.4 on page 20 and Definition 2.3.5.5 on page 20.

Definition 2.5.3.5 (Basic Terms of ACP_δ)

When we speak of basic terms in the context of ACP_δ , we mean (δ, δ) -basic terms as defined in Definition 2.3.5.6 on page 20.

2.6 Properties

As we said in the introduction, we give no proofs in this chapter. However, that does not keep us from stating some properties of the process algebras we introduced. In this section, we list a number of important properties that are desirable in a process algebra, and list the process algebras that have this property. For every property, we give a formal definition, and an intuitive justification why that property is desirable.

Property 2.6.1.1 (Congruence)

Given a process algebra P and a binary relation R on closed terms of P , we call R a *congruence* if it satisfies the following requirements:

- (i). R is an equivalence relation (i.e., reflexive, symmetric, and transitive),
- (ii). For every $n \in \mathbb{N}$ and every n -ary operator \otimes of P , we have:

$$(\forall_{i \leq n} R(t_i, t'_i)) \implies R(\otimes(t_1, \dots, t_n), \otimes(t'_1, \dots, t'_n))$$

for all closed terms $t_1, \dots, t_n, t'_1, \dots, t'_n$ of P .

This property holds for all process algebras given in this chapter with respect to bisimulation.

Example 2.6.1.2 (Congruence)

We take the process algebra BPA for P , and the corresponding bisimulation \sim as R . Congruence then says that if:

$$s \sim s' \text{ and } t \sim t'$$

then also:

$$s + t \sim s' + t' \text{ and } s \cdot t \sim s' \cdot t'$$

So, since $a + b \sim b + a$ and $c \sim c$, we should also have $(a + b) \cdot c \sim (b + a) \cdot c$.

☞ This property is vital for any process algebra. If we would not have it, we would not be able to build a bisimulation model.

Property 2.6.1.3 (Elimination to a Process Algebra)

Given process algebras P and P' such that the signature of P' is a subset of the signature of P , we say that P has the *elimination to P' property* if for every closed term s of P there exists a closed term t of P' such that $P \vdash s = t$.

This property holds for all Process Algebras and Algebras of Communicating Process in this chapter with respect to their corresponding Basic Process Algebras.

Example 2.6.1.4 (Elimination to BPA)

We take the process algebra PA for P , and the closed term $a \cdot a \parallel b \cdot b$ for s . Now, by the elimination to BPA property, there must exist a BPA term t such that $\text{PA} \vdash s = t$. As we have shown in Example 2.4.1.3 on page 22, such a term does indeed exist.

Property 2.6.1.5 (Elimination to Basic Terms)

Given a process algebra P we say that P has the *elimination to basic terms property* if for every closed term s of P there exists a basic term t of P such that $P \vdash s = t$.

This property holds for all process algebras given in this chapter.

☞ The elimination to basic terms property often allows us to use induction on basic terms when we want to prove some property for closed terms. We can do this because every closed term has a corresponding basic term, and so every closed term is directly or indirectly covered in the induction. For example, if we want to prove for all closed terms x and y of PA_δ that $\text{PA}_\delta \vdash x \parallel y \cdot \delta = x \cdot \delta \parallel y$, then we may assume, without loss of generality, that x and y are basic terms.

Remark 2.6.1.6 (Elimination)

Properties 2.6.1.3 and 2.6.1.5 are often simply called *elimination*. When we say “ P has the elimination property”, it will always be clear from the context which variant of elimination we intend.

Property 2.6.1.7 (Soundness)

Given a process algebra P and a structure M , we say that M is a *model of P* if every equality between two closed terms implied by P does also hold in M . This is also called the *soundness property* of P with respect to M .

This property holds for all bisimulation models given in this chapter with respect to their corresponding process algebras.

Example 2.6.1.8 (Soundness)

We take the process algebra PA for P and the bisimulation model of PA for M . As we have $PA \vdash a \parallel b = a \cdot b + b \cdot a$, by the soundness property we should also have $a \parallel b \sim_{PA} a \cdot b + b \cdot a$. This is indeed the case, as can be checked with the deduction rules of PA.

☞ The soundness property expresses that a certain structure is indeed a model with respect to a certain process algebra. As such, it is vital.

Property 2.6.1.9 (Completeness)

Given a process algebra P and a model M , we say the P is *complete with respect to* M if every two closed terms that have the same interpretation in M , are also derivably equal in P . This is also called the *completeness property* of P with respect to M .

This property holds for all process algebras given in this chapter with respect to their corresponding bisimulation models.

Example 2.6.1.10 (Completeness)

We take the process algebra PA for P and the bisimulation model of PA for M . As we have $a \parallel b \sim_{PA} a \cdot b + b \cdot a$, by the completeness property we should also have $PA \vdash a \parallel b = a \cdot b + b \cdot a$. This is indeed the case, as can be checked with the axioms of PA.

☞ The completeness property expresses that a certain process algebra captures all structure that is present in the model associated with it. This is desirable, but not vital. It is conceivable that the model contains more structure than we need to know, and therefore we do not want to capture completely in our process algebra. For example, our models may contain so-called *junk*, elements that are not represented by any closed term. This can be done on purpose, e.g., because these elements represent solutions of recursive equations.

The soundness and completeness properties are each other's duals. If both hold, P and M exactly capture the same "mathematical world". In Chapter 4 we will examine these properties in more detail.

Property 2.6.1.11 (Axioms of Standard Concurrency)

Given a process algebra P that contains a \parallel operator, we say that *the axioms of standard concurrency hold in* P if for all closed terms x , y , and z of P the following six equalities are derivable:

- (i). $P \vdash x \mid y = y \mid x$
- (ii). $P \vdash x \parallel y = y \parallel x$
- (iii). $P \vdash (x \mid y) \mid z = x \mid (y \mid z)$
- (iv). $P \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$
- (v). $P \vdash x \mid (y \parallel z) = (x \mid y) \parallel z$
- (vi). $P \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$

If \parallel or \mid are not present in the signature of P , the equalities above that contain them are left out of the consideration.

The axioms of standard concurrency hold for all process algebras given in this chapter that contain a \parallel operator.

Remark 2.6.1.12 (Axioms of Standard Concurrency)

The equalities of Property 2.6.1.11 on the preceding page are historically known as the *axioms of standard concurrency* (see for example page 97 of BAETEN AND WEIJLAND [38]). However, as their status is not that of *axioms*, but that of from the axioms derivable *equalities* (for closed terms), the name is quite misplaced and misleading.

Property 2.6.1.13 (Embedding)

Given two process algebras P and P' and an injective *projection function* π from the signature of P into the signature of P' , we call π an *embedding of P into P'* if all derivable equalities between closed terms of P remain derivable equalities when projected into P' by π . So, for all closed terms s and t of P , we should have:

$$P \vdash s = t \implies P' \vdash \pi(s) = \pi(t)$$

If such a projection exists, we say that P *can be embedded into P'* , denoted by the *embedding relation* $P \subseteq P'$. This relation is reflexive and transitive, and thus constitutes a *preorder*.

The following embeddings hold between the process algebras given in this chapter:

- (i). $BPA \subseteq BPA_\delta \subseteq BPA_{\delta\varepsilon}$, $BPA \subseteq BPA_\varepsilon \subseteq BPA_{\delta\varepsilon}$, $BPA_\delta \subseteq BPA_\delta$
- (ii). $PA \subseteq PA_\delta \subseteq PA_\varepsilon$, $PA_\delta \subseteq PA_\delta$
- (iii). $ACP \subseteq ACP_\varepsilon$, $ACP \subseteq ACP_\delta$
- (iv). $BPA \subseteq PA \subseteq ACP$
- (v). $BPA_\delta \subseteq PA_\delta \subseteq ACP$
- (vi). $BPA_{\delta\varepsilon} \subseteq PA_\varepsilon \subseteq ACP_\varepsilon$
- (vii). $BPA_\delta \subseteq PA_\delta \subseteq ACP_\delta$

Note that where a process algebra that has communication is used in the above embeddings, we also need to specify the communication function γ . In the embedding $PA \subseteq ACP$, for example, we want the merge of ACP to behave as the free merge of PA , in order to make a proper embedding. Therefore, we choose $\gamma(a, b)$ to be δ for all $a, b \in A_\delta$. This is called the *empty communication function*, which is the only communication function possible in this case. In the embedding $ACP \subseteq ACP_\varepsilon$, however, there is no need to commit ourselves to a specific communication function. In this case, any communication function will do, as both sides of the embedding exhibit the same communication behavior.

So, whenever we write down an embedding, we will use the convention that in an embedding $P \subseteq P'$ where P' has communication while P has not, the communication function is implicitly understood to be empty. When P and P' both have communication, we implicitly take the embedding to be universally quantified over all communication functions.

Example 2.6.1.14 (Embedding)

All the embeddings shown above can be implemented by taking the identity as the projection π (although this will not always be the case in what follows). As an example, we take the embedding $BPA_{\delta\varepsilon} \subseteq PA_\varepsilon$. As $BPA_{\delta\varepsilon} \vdash \varepsilon \cdot (a + \delta) = a$, we should also have $PA_\varepsilon \vdash \varepsilon \cdot (a + \delta) = a$. This is indeed the case.

Property 2.6.1.15 (Conservativity)

Given two process algebras P and P' such that the signature of P is fully contained in P' , we say that P' is a *conservative extension* of P , if for all closed terms s and t of P we have:

$$P \vdash s = t \iff P' \vdash s = t$$

Note the difference with embedding: here we also require that P' does not derive any *new* equalities that did not hold in P . So, every conservative extension gives rise to an embedding using the identity as projection function, but an embedding does not necessarily give rise to a conservative extension.

Nevertheless, for all embeddings $P \subseteq P'$ given in Property 2.6.1.13, we also have that P' is a conservative extension of P .

2.7 Other Topics

Of course this chapter could be extended and made more complete. We mention some topics of untimed process algebra we have *not* treated.

To begin with, we have said nothing about *abstraction*. Abstraction is a means of declaring certain actions “internal” or “silent”, in order to reach a level of abstraction where the identity of those actions is not relevant anymore. Process algebra without abstraction or the empty process is called *concrete* process algebra, while process algebra with abstraction or the empty process is called *abstract* process algebra (BAETEN AND BERGSTRÄ [13]).

Furthermore, we have not mentioned *projection*, a means for limiting the depth of processes, *renaming*, an operator that changes the identity of actions, and *state operators*, which introduce automata in the realm of process algebra. Also not mentioned are *recursion* and *iteration*, methods for constructing infinite processes.

And then, of course, ACP-style process algebra is neither the first nor the only algebraic process theory ever invented. We name three others: the *Calculus of Communicating Systems* (CCS), as described by MILNER [142], *Communicating Sequential Processes* (CSP), as described by HOARE [98], and the *Algebraic Theory of Processes*, as described by HENNESSY [94].

All the above topics are treated, at least briefly, in BAETEN AND WEIJLAND [38] or BAETEN AND VERHOEF [37], except for the iteration operator, which was introduced by BERGSTRÄ, BETHKE, AND PONSE [41], and is also treated in the dissertation of FOKKINK [80]. For more information about the state operator, see the dissertation of BLANCO [48].

3

Discrete-Time Process Algebra

3.1 Introduction

Untimed process algebra, as described in Chapter 2, lacks the ability to quantitatively reason about *time*: although we can express the *sequence* in which actions take place, we cannot make explicit *the moment in time* at which they occur.

One can reason, and very validly so, that abstracting from timing aspects is justifiable because often timing aspects are irrelevant in the verification of systems. That this is no idle claim can be observed from the many verifications in untimed process algebra that have been published—see for example the ones given in [11, 120, 148, 167, 168, 188, 200]. However, sometimes the correctness of a protocol hinges on delicate timing aspects (e.g., in *Fischer's Protocol*, FISCHER [79], and see also Chapter 7), or the timing aspects themselves of a protocol are the object of study (e.g., in HILLEBRAND [97]). In such cases quantitative analysis of timing aspects cannot be avoided, and untimed process algebra is therefore of little help. We can try to artificially encode timing aspects using untimed process algebra, as is for example done by CERONE, *et al.* [61] (using the process algebra *CIRCAL*, see MILNE [141]) and GROOTE AND VAN DE POL [90]. But this is stretching untimed process algebra to its limits; clearly it was never built for these purposes.

So, we need to extend process algebra with constructs that allow the explicit specification of timing aspects. One way to do this would be to parameterize our actions with real numbers that represent time stamps (i.e., labels indicating the time of execution), leading to so-called *real-time process algebra*. This is for example done by BAETEN AND BERGSTRA [16, 22], and in the dissertations of KLUSENER [117] and FOKKINK [80]. Although this is a very powerful and general method, its power also backfires: the resulting theories are very complicated.

Another way of introducing time is to cut up time into a countably infinite number of *time-slices*. Between these time-slices, a *clock tick* takes place that is explicitly indicated. This leads to so-called *discrete-time process algebra*: the passing of time is modeled as being *discrete*, consisting of the passage of a number of distinct time-slices. This is for example done by BAETEN AND BERGSTRA [24, 25] and GROOTE [88]. The theories resulting from this approach are much friendlier, and many theoretical results can be obtained. Of course discrete-time process algebras are also less expressive than real-time ones.

Whether this is an impediment to their practical application varies with the application one has in mind. As is shown by BOS AND RENIERS [53], discrete-time process algebra is certainly not a toy theory, and can be successfully applied in the analysis of real-life problems.

In this chapter we will give an introduction in so-called *relative-time discrete-time process algebra*. We will not give many theorems or proofs; most of these will be given in Chapters 4, 5, and 6. Also we restrict ourselves to the treatment of Basic Process Algebras only. The issues involving concurrency are complex enough to justify dedicating a whole chapter to them: Chapter 5.

3.2 Basic Process Algebras

We introduce several relative-time discrete-time basic process algebras. To make the passage of time explicit, we introduce the so-called *time-unit delay operator* σ_{rel} . The process $\sigma_{\text{rel}}(x)$ behaves like x , except for the fact that everything happens one time-slice later.

3.2.1 $\text{BPA}_{\text{drt}}^- - \delta$

In this section, we define the process algebra $\text{BPA}_{\text{drt}}^- - \delta$. It is based on BPA, so there is no deadlock present in its signature.

Definition 3.2.1.1 (Signature of $\text{BPA}_{\text{drt}}^- - \delta$)

The signature of $\text{BPA}_{\text{drt}}^- - \delta$ consists of the *undelayable actions* $\{\underline{\underline{a}} \mid a \in A\}$, the *alternative composition operator* $+$, the *sequential composition operator* $;$, and the *time-unit delay operator* σ_{rel} .

Remark 3.2.1.2 (Undelayable Actions)

By a doubly-underlined symbol, e.g. $\underline{\underline{a}}$, we denote a so-called *undelayable action*. An undelayable action can execute only in the time-slice it is initialized in; when the process it is part of moves into the following time-slice, it is lost.

Note that in BAETEN AND BERGSTRA [21, 24, 25] the notation $\text{cts}(a)$ (current time-slice a) is used to denote the undelayable action $\underline{\underline{a}}$.

Remark 3.2.1.3 (Time-Unit Delay Operator)

The *time-unit delay operator*, notation $\sigma_{\text{rel}}(x)$, postpones the execution of a process to the next time-slice. The subscript rel in σ_{rel} indicates we are working in so-called *relative time*, where time is measured relative to the moment a process is started (as opposed to *absolute time* (see BAETEN AND BERGSTRA [24], or Section 8.2.2) where time is measured relative to some fixed point in time, irrespective of the moment a process is started.) So the process $\sigma_{\text{rel}}(\underline{\underline{a}})$, when initialized in the fifth time-slice, will execute an a in the sixth time-slice.

Finally, note that the use of the greek letter σ to denote idling until the next time-slice is due to HENNESSY AND REGAN [95] (see also Section 8.6.2), who in turn were inspired by a notation due to PHILIPS [163]

Remark 3.2.1.4 (Time Factorization)

An important concept we use is the concept of *time factorization*, which enforces that when a process moves into the next time-slice, this does not determine a choice. Or, in

other words, when a process moves into the next-time-slice, no summands of the process are lost.

We distinguish two variants: *weak time factorization*, and *strong time factorization*. In *weak time factorization*, we only demand that no summands that start with moving to the next time-slice get lost when a process moves into the next-time slice. In our setting, this means that no summands of the form $\sigma_{\text{rel}}(x)$ should get lost. In *strong time factorization*, we demand that no summands at all get lost.

We give two examples. First, consider the following process P :

$$P \equiv \underline{\underline{a}} + \sigma_{\text{rel}}(\underline{\underline{b}}) + \sigma_{\text{rel}}(\sigma_{\text{rel}}(\underline{\underline{c}}))$$

When we have weak time factorization, the execution of the σ_{rel} in the second summand of P should not lead to the disappearance of the third summand, and vice versa. So, in weak time factorization, P should be equal to:

$$\underline{\underline{a}} + \sigma_{\text{rel}}(\underline{\underline{b}} + \sigma_{\text{rel}}(\underline{\underline{c}}))$$

In this way, the scope of the either one of the σ_{rel} operators at the head of a summand of P effectively spans both σ_{rel} -summands of the process P

When we have strong time factorization, the execution of the σ_{rel} in the second or third summand of P is not possible at all, as that would lead to the disappearance of the first summand, $\underline{\underline{a}}$. So, in strong time factorization, P should be equal to just:

$$\underline{\underline{a}}$$

Due to this behavior, where a summand in the current time-slice is always chosen over a summand that idles till the next time slice, strong time factorization is sometimes also called *maximal progress* (e.g., in BAETEN AND BERGSTRA [24]). However, as maximal progress is also often used to denote entirely different notions, we prefer to use the term strong time factorization.

Secondly, consider the following process Q :

$$Q \equiv \sigma_{\text{rel}}(\underline{\underline{a}}) + \sigma_{\text{rel}}(\sigma_{\text{rel}}(\underline{\underline{b}}))$$

As this process only contains summands of the form $\sigma_{\text{rel}}(x)$, the notions of weak and strong time factorization coincide. In both settings, Q should be equal to:

$$\sigma_{\text{rel}}(\underline{\underline{a}} + \sigma_{\text{rel}}(\underline{\underline{b}}))$$

Of course there is a third alternative: no time factorization at all. All three alternatives occur in the literature. To name examples of each: NICOLLIN AND SIFAKIS [154] adhere to strong time factorization, BAETEN AND BERGSTRA [24] adhere to weak time factorization, and GROOTE [87] has no time factorization at all. It is even possible to combine weak time factorization and strong time factorization in one process algebra, by using two different choice operators. For an example of this, see MOLLER AND TOFTS [145].

We choose to follow the weak time factorization approach, as we find strong time factorization too restrictive, and no time factorization counterintuitive (because we feel that the mere progress of time should not, in an invisible manner, introduce choices).

$$\begin{aligned}\sigma_{\text{rel}}(x) + \sigma_{\text{rel}}(y) &= \sigma_{\text{rel}}(x + y) && \text{DRT1} \\ \sigma_{\text{rel}}(x) \cdot y &= \sigma_{\text{rel}}(x \cdot y) && \text{DRT2}\end{aligned}$$

Table 3.1: Additional axioms for $\text{BPA}_{\text{drt}}^- - \delta$.

Definition 3.2.1.5 (Axioms of $\text{BPA}_{\text{drt}}^- - \delta$)

The process algebra $\text{BPA}_{\text{drt}}^- - \delta$ is axiomatized by the axioms of BPA given in Definition 2.3.1.6 on page 8 and Axioms DRT1–DRT2 shown in Table 3.1: $\text{BPA}_{\text{drt}}^- - \delta = \text{A1–A5} + \text{DRT1–DRT2}$.

☞ Axiom DRT1 expresses the *weak time factorization* property mentioned before: moving on to the next time-slice, when no actions are enabled, does not determine a choice. Axiom DRT2 expresses the *relative-time character* of $\text{BPA}_{\text{drt}}^- - \delta$: once a process has moved to a following time-slice, the complete remaining part of the process has also moved on, irrespective of the scope of the time-unit delay operator.

Definition 3.2.1.6 (Notation Regarding Semantics III)

Besides the deduction rule notations $x \xrightarrow{a} x'$, $x \xrightarrow{a} \surd$, $x \xrightarrow{a}$, and $x \nrightarrow$ introduced in Definition 2.3.1.10 on page 10, we now also use $x \xrightarrow{\sigma} x'$ to denote that x can do a σ -step (also called σ -transition or time step) to x' (i.e., move to the following time-slice and become x'), and $x \nrightarrow^{\sigma}$ to denote that x cannot do a σ -step.

Definition 3.2.1.7 (Semantics of $\text{BPA}_{\text{drt}}^- - \delta$)

The semantics of $\text{BPA}_{\text{drt}}^- - \delta$ are given by the term-deduction system $T(\text{BPA}_{\text{drt}}^- - \delta)$ induced by the deduction rules shown in Table 2.4 on page 11 and Table 3.2.

$$\begin{array}{ccc} \underline{a} \xrightarrow{a} \surd & \sigma_{\text{rel}}(x) \xrightarrow{\sigma} x & \frac{x \xrightarrow{\sigma} x'}{x \cdot y \xrightarrow{\sigma} x' \cdot y} \\ \\ \frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} x' + y'} & \frac{x \xrightarrow{\sigma} x', y \nrightarrow^{\sigma}}{x + y \xrightarrow{\sigma} x'} & \frac{x \nrightarrow^{\sigma}, y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} y'}\end{array}$$

Table 3.2: Deduction rules for untimed actions and time-unit delay.

☞ The deduction rules for \underline{a} and $\sigma_{\text{rel}}(x)$ are self-evident; note that $\sigma_{\text{rel}}(x)$ can never do a terminating step. The additional deduction rule for sequential composition is also clear: with respect to sequential composition a σ -step behaves just like an action step.

The deduction rules for the alternative composition are less trivial. By the principle of weak time factorization a σ -step may not determine a choice between summands of the form $\sigma_{\text{rel}}(x)$. So, if we have process terms x and y , such that x can do a σ -step to

x' , then $x + y$ can *only* do a σ step to x' provided y does not have the possibility to do a σ -step. If y can do a σ -step to y' , then $x + y$ can do a σ -step to $x' + y'$, and to nothing else.

Definition 3.2.1.8 (Bisimulation for $BPA_{\text{drt}}^- - \delta$)

Bisimulation for $BPA_{\text{drt}}^- - \delta$ is defined as follows; a binary relation R on closed $BPA_{\text{drt}}^- - \delta$ terms is a bisimulation if the following transfer conditions hold for all closed $BPA_{\text{drt}}^- - \delta$ terms p and q :

- (i). if $R^S(p, q)$ and $T(BPA_{\text{drt}}^- - \delta) \models p \xrightarrow{a} p'$, where $a \in A$, then there exists a closed term q' , such that $T(BPA_{\text{drt}}^- - \delta) \models q \xrightarrow{a} q'$ and $R^S(p', q')$,
- (ii). if $R^S(p, q)$ and $T(BPA_{\text{drt}}^- - \delta) \models p \xrightarrow{\sigma} p'$, then there exists a closed term q' , such that $T(BPA_{\text{drt}}^- - \delta) \models q \xrightarrow{\sigma} q'$ and $R^S(p', q')$,
- (iii). if $R^S(p, q)$ and $T(BPA_{\text{drt}}^- - \delta) \models p \xrightarrow{a} \surd$, where $a \in A$, then $T(BPA_{\text{drt}}^- - \delta) \models q \xrightarrow{a} \surd$.

Two $BPA_{\text{drt}}^- - \delta$ terms p and q are bisimilar, notation $p \sim_{BPA_{\text{drt}}^- - \delta} q$, if there exists a bisimulation relation R such that $R(p, q)$.

☞ Extending the definition of bisimulation to incorporate σ -steps is very straightforward: with respect to bisimulation σ -steps are treated as if they were action steps.

Remark 3.2.1.9 (Bisimulation for $BPA_{\text{drt}}^- - \delta$)

In Definition 3.2.1.8 we can treat time transitions as if they were action transitions because we have carefully constructed $T(BPA_{\text{drt}}^- - \delta)$ in such a way that a closed term never has more than one outgoing time transition. Therefore, weak time factorization is always assured: since no closed term ever has the choice between different time steps, time steps never introduce non-determinism. Another option would be to treat time transitions as action transitions in the *deduction rules*, and enforce weak time factorization by means of the definition of *bisimulation*. This leads to a simpler term-deduction system, but a more difficult definition of bisimulation. Such an approach is, for example, taken by BERGSTRA, FOKKINK, AND MIDDELBURG [42] (see also Section 8.2.8).

Lemma 3.2.1.10 (Closures of Bisimulation Relations)

Let $R \subseteq C(BPA_{\text{drt}}^- - \delta) \times C(BPA_{\text{drt}}^- - \delta)$ be a bisimulation relation on closed $BPA_{\text{drt}}^- - \delta$ terms. Then we have:

- (i). R^R is again a bisimulation relation on closed $BPA_{\text{drt}}^- - \delta$ terms,
- (ii). R^S is again a bisimulation relation on closed $BPA_{\text{drt}}^- - \delta$ terms,
- (iii). R^T is again a bisimulation relation on closed $BPA_{\text{drt}}^- - \delta$ terms.

Proof

- (i). By the definition of reflexive closure, we have $R^R = R \cup \{(x, x) \mid x \in C(BPA_{\text{drt}}^- - \delta)\}$. We now have to prove that the transfer conditions of Definition 3.2.1.8 hold with respect to R^R for all closed $BPA_{\text{drt}}^- - \delta$ terms p and q . First, for all $(p, q) \in R^R$ such that $(p, q) \in R$ the transfer conditions are fulfilled because R is a bisimulation relation. Secondly, for all $(p, p) \in R^R$, the transfer conditions are trivially fulfilled because p and p can do exactly the same steps.

- (ii). Trivially fulfilled because all transfer conditions are invariant under the symmetric closure of R .
- (iii). See BASTEN [40].

■

Remark 3.2.1.11 (Closures of Bisimulation Relations)

Lemma 3.2.1.10 on the page before does not only hold in the context of bisimulation for $\text{BPA}_{\text{drt}}^- - \delta$, but also in the context of any other bisimulation definition in this thesis. In each case, the proof is the same as for $\text{BPA}_{\text{drt}}^- - \delta$.

Finally, note that the proof of Lemma 3.2.1.10(iii) is trivial in the absence of silent actions (to be treated later, in Chapter 7), but highly non-trivial in their presence.

Definition 3.2.1.12 (Bisimulation Model for $\text{BPA}_{\text{drt}}^- - \delta$)

The bisimulation model for $\text{BPA}_{\text{drt}}^- - \delta$ is defined in the same way as for BPA. Replace “BPA” by “ $\text{BPA}_{\text{drt}}^- - \delta$ ” in Definition 2.3.1.16.

Definition 3.2.1.13 (Basic Terms of $\text{BPA}_{\text{drt}}^- - \delta$)

We define σ -basic terms inductively as follows:

- (i). For every $a \in A$, the undelayable action \underline{a} is a σ -basic term,
- (ii). if $a \in A$ and t is a σ -basic term, then $\underline{a} \cdot t$ is a σ -basic term,
- (iii). if t and s are σ -basic terms, then $t + s$ is a σ -basic term,
- (iv). if t is a basic term, then $\sigma_{\text{rel}}(t)$ is a σ -basic term.

From now on, when we speak of basic terms in the context of $\text{BPA}_{\text{drt}}^- - \delta$, we mean σ -basic terms.

☞ The basic terms of $\text{BPA}_{\text{drt}}^- - \delta$ are like the basic terms of BPA, with σ_{rel} added as new constructor.

Definition 3.2.1.14 (Number of Symbols of a $\text{BPA}_{\text{drt}}^- - \delta$ Term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). For $a \in A$, we define $n(\underline{a}) = 1$,
- (ii). for closed $\text{BPA}_{\text{drt}}^- - \delta$ terms x and y , we define $n(x + y) = n(x \cdot y) = n(x) + n(y) + 1$,
- (iii). for a closed $\text{BPA}_{\text{drt}}^- - \delta$ term x , we define $n(\sigma_{\text{rel}}(x)) = n(x) + 1$.

☞ We will $n(x)$ use in induction proofs where the induction is on the number of symbols of a closed term.

3.2.2 BPA_{drt}⁻-ID

In this section, we add an undelayable deadlock constant, denoted $\underline{\delta}$, to BPA_{drt}⁻- δ . Also we introduce the so-called “now” operator, denoted ν_{rel} . This operator is defined such that $\nu_{\text{rel}}(x)$ denotes the part of x that can do an action within the first time-slice (no restrictions on later actions). The resulting process algebra is called BPA_{drt}⁻-ID.

Definition 3.2.2.1 (Signature of BPA_{drt}⁻-ID)

The signature of BPA_{drt}⁻-ID consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , and the “now” operator ν_{rel} .

☞ Because $\underline{\delta}$ is definable in terms of ν_{rel} and σ_{rel} , we could not have introduced ν_{rel} in BPA_{drt}⁻- δ .

Definition 3.2.2.2 (Axioms of BPA_{drt}⁻-ID)

The process algebra BPA_{drt}⁻-ID is axiomatized by the axioms of BPA_{drt}⁻- δ given in Definition 3.2.1.5 on page 44, Axioms DRT3–DRT5 shown in Table 3.3, and Axioms DCS1–DCS4 shown in Table 3.4: BPA_{drt}⁻-ID = A1–A5 + DRT1–DRT5 + DCS1–DCS4.

$\underline{\delta} \cdot x = \underline{\delta}$	DRT3
$\underline{a} + \underline{\delta} = \underline{a}$	DRT4
$\sigma_{\text{rel}}(x) + \underline{\delta} = \sigma_{\text{rel}}(x)$	DRT5

Table 3.3: Additional axioms for BPA_{drt}⁻-ID

$\nu_{\text{rel}}(\underline{a}) = \underline{a}$	DCS1
$\nu_{\text{rel}}(x + y) = \nu_{\text{rel}}(x) + \nu_{\text{rel}}(y)$	DCS2
$\nu_{\text{rel}}(x \cdot y) = \nu_{\text{rel}}(x) \cdot y$	DCS3
$\nu_{\text{rel}}(\sigma_{\text{rel}}(x)) = \underline{\delta}$	DCS4

Table 3.4: Axioms for “now” operator.

☞ Axiom DRT3 is the discrete-time counterpart of Axiom A7 we already encountered on page 14. Axioms DRT4 and DRT5 serve to ensure that for all closed terms x , we have BPA_{drt}⁻-ID $\vdash x + \underline{\delta} = x$. We could have made this equality an axiom itself, but then we would have to drop it again when we introduce the immediate deadlock (see also Remark 2.3.2.3 on page 14 and Remark 3.2.2.3 on the following page).

Axiom DCS1 expresses the fact that an undelayable action always starts in the current time-slice, so the “now” operator has no effect on it. Axiom DCS2 expresses that

the part of $x + y$ that starts in the current time-slice consists of the alternative composition of the parts of x and y that start in the current time-slice, i.e., the “now” operator distributes over the alternative composition. Axiom DCS3 expresses that the part of $x \cdot y$ that starts in the current time-slice consists of the part of x that starts in the current time-slice, followed by y (which need not start in the current time-slice). Axiom DCS4, finally, expresses that $\sigma_{\text{rel}}(x)$ cannot start in the current time-slice.

Remark 3.2.2.3 (DRT4 and DRT5 versus DRT4A)

Note that for *closed* $\text{BPA}_{\text{drt}}^-$ -ID terms, Axioms DRT4–DRT5 shown in Table 3.3 on the page before are equivalent with Axiom DRT4A shown in Table 3.5. Therefore we could replace Axioms DRT4–DRT5 in $\text{BPA}_{\text{drt}}^-$ -ID by DRT4A without affecting the soundness or completeness of the resulting process algebra. This is for example done in BOS AND REINIERS [53].

One reason for doing so, could be the fact that DRT4A is a straightforward reformulation of Axiom A6 from BPA_{δ} , in a setting with undelayable actions. However, we prefer Axioms DRT4–DRT5 over DRT4A because the latter does not extend to the discrete-time process algebras with immediate deadlock we will describe later on. If we need the equality of DRT4A, we can derive it as a lemma.

$$x + \underline{\underline{\delta}} = x \quad \text{DRT4A}$$

Table 3.5: Alternative axiom for undelayable deadlock.

☞ If we, for example, want to derive $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash \underline{\underline{a}} \cdot \underline{\underline{b}} + \underline{\underline{\delta}} = \underline{\underline{a}} \cdot \underline{\underline{b}}$, we do so as follows: $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash \underline{\underline{a}} \cdot \underline{\underline{b}} + \underline{\underline{\delta}} = \underline{\underline{a}} \cdot \underline{\underline{b}} + \underline{\underline{\delta}} \cdot \underline{\underline{b}} = (\underline{\underline{a}} + \underline{\underline{\delta}}) \cdot \underline{\underline{b}} = \underline{\underline{a}} \cdot \underline{\underline{b}}$.

Definition 3.2.2.4 (Semantics of $\text{BPA}_{\text{drt}}^-$ -ID)

The semantics of $\text{BPA}_{\text{drt}}^-$ -ID are given by the term-deduction system $T(\text{BPA}_{\text{drt}}^- \text{-ID})$, induced by the deduction rules for $\text{BPA}_{\text{drt}}^- \text{-}\delta$ given in Definition 3.2.1.7 on page 44, and the deduction rules for ν_{rel} shown in Table 3.6.

$$\frac{x \xrightarrow{a} x'}{\nu_{\text{rel}}(x) \xrightarrow{a} x'} \quad \frac{x \xrightarrow{a} \surd}{\nu_{\text{rel}}(x) \xrightarrow{a} \surd}$$

Table 3.6: Deduction rules for “now” operator.

☞ The deduction rules for the “now” operator speak for themselves: all transitions that x can do, can also be done by $\nu_{\text{rel}}(x)$, with the exception of σ -transitions.

Definition 3.2.2.5 (Bisimulation and Bisimulation Model for $\text{BPA}_{\text{drt}}^-$ -ID)

Bisimulation for $\text{BPA}_{\text{drt}}^-$ -ID and the corresponding bisimulation model are defined in the same way as for $\text{BPA}_{\text{drt}}^- \text{-}\delta$ and BPA respectively. Replace “ $\text{BPA}_{\text{drt}}^- \text{-}\delta$ ” by “ $\text{BPA}_{\text{drt}}^- \text{-ID}$ ” in Definition 3.2.1.8 on page 45 and “BPA” by “ $\text{BPA}_{\text{drt}}^- \text{-ID}$ ” in Definition 2.3.1.16 on page 12.

Definition 3.2.2.6 (Basic Terms of $\text{BPA}_{\text{drt}}^-$ -ID)

We define $(\sigma, \underline{\delta})$ -basic terms inductively as follows:

- (i). For every $a \in A_\delta$, \underline{a} is a $(\sigma, \underline{\delta})$ -basic term,
- (ii). if $a \in A_\delta$ and t is a $(\sigma, \underline{\delta})$ -basic term, then $\underline{a} \cdot t$ is a $(\sigma, \underline{\delta})$ -basic term,
- (iii). if t and s are $(\sigma, \underline{\delta})$ -basic terms, then $t + s$ is a $(\sigma, \underline{\delta})$ -basic term,
- (iv). if t is a $(\sigma, \underline{\delta})$ -basic term, then $\sigma_{\text{rel}}(t)$ is a $(\sigma, \underline{\delta})$ -basic term.

From now on, when we speak of basic terms in the context of $\text{BPA}_{\text{drt}}^-$ -ID, we mean $(\sigma, \underline{\delta})$ -basic terms.

Definition 3.2.2.7 (Summation Convention)

We will use the meta-notation $\sum_{i \in I} t_i$ to denote the *summation* over some finite index set $I = \{1, \dots, n\}$:

$$\sum_{i \in I} t_i \stackrel{\text{def}}{=} t_1 + \dots + t_n$$

Note that due to the commutativity and associativity of the alternative composition, the order of the summands is immaterial. The summation over the empty set yields the undelayable deadlock:

$$\sum_{i \in \emptyset} t_i \stackrel{\text{def}}{=} \underline{\delta}$$

Note however, that we use the convention that an empty summation disappears in the presence of other summands. So:

$$s + \sum_{i \in \emptyset} t_i \stackrel{\text{def}}{=} s$$

Here s and t_i denote arbitrary closed terms.

☞ We will use the summation notation to give an alternative characterization of basic terms.

Theorem 3.2.2.8 (General Form of Basic Terms of $\text{BPA}_{\text{drt}}^-$ -ID)

Modulo the commutativity and associativity of the $+$, all basic terms t of $\text{BPA}_{\text{drt}}^-$ -ID are of the form:

$$t \equiv \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{b}_j + \sum_{k < p} \sigma_{\text{rel}}(u_k)$$

for $m, n, p \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and basic terms s_i and u_k .

Proof Trivial, by inspection of the definition of basic terms, Definition 3.2.2.6. Observe that the general form of basic terms is closed under the formation rules given in Definition 3.2.2.6. ■

☞ Some proofs by induction on basic terms do not work very well, especially when the induction is on several variables simultaneously, leading to a unmanageable number of case distinctions. In these cases it is sometimes easier to use induction on the number of symbols in the general form of basic terms. For examples of such proofs, see BAETEN AND WEIJLAND [38].

Definition 3.2.2.9 (Number of Symbols of a $\text{BPA}_{\text{drt}}^-$ -ID Term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). For $a \in A_\delta$, we define $n(\underline{a}) = 1$,
- (ii). for closed $\text{BPA}_{\text{drt}}^-$ -ID terms x and y , we define $n(x + y) = n(x \cdot y) = n(x) + n(y) + 1$,
- (iii). for a closed $\text{BPA}_{\text{drt}}^-$ -ID term x , we define $n(\sigma_{\text{rel}}(x)) = n(\nu_{\text{rel}}(x)) = n(x) + 1$.

Lemma 3.2.2.10 (Representation of $\text{BPA}_{\text{drt}}^-$ -ID Terms)

Let t be a basic term. Then either $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash t = \nu_{\text{rel}}(t)$, or there exists a basic term s such that $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash t = \nu_{\text{rel}}(t) + \sigma_{\text{rel}}(s)$ and $n(s) < n(t)$.

Proof Let t be a basic term. By Theorem 3.2.2.8, we may now proceed by case analysis on the general form of basic terms:

- (i). Either we have no σ_{rel} -summands ($p = 0$ in Theorem 3.2.2.8):

$$t \equiv \sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j}$$

for $m, n \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and basic terms s_i . Then we have the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt}}^- \text{-ID} \vdash t &= \sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} \\ &= \sum_{i < m} \nu_{\text{rel}}(\underline{a_i}) \cdot s_i + \sum_{j < n} \nu_{\text{rel}}(\underline{b_j}) \\ &= \sum_{i < m} \nu_{\text{rel}}(\underline{a_i} \cdot s_i) + \sum_{j < n} \nu_{\text{rel}}(\underline{b_j}) \\ &= \nu_{\text{rel}} \left(\sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} \right) \\ &= \nu_{\text{rel}}(t) \end{aligned}$$

- (ii). Or we have at least one σ_{rel} -summand ($p \geq 1$ in Theorem 3.2.2.8):

$$t \equiv \sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} + \sum_{k < p} \sigma_{\text{rel}}(u_k)$$

for $m, n, p \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and basic terms s_i and u_k . Then we have the following computation:

$$\text{BPA}_{\text{drt}}^- \text{-ID} \vdash t = \sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} + \sum_{k < p} \sigma_{\text{rel}}(u_k)$$

$$\begin{aligned}
&= \sum_{i < m} \nu_{\text{rel}}(\underline{a_i}) \cdot s_i + \sum_{j < n} \nu_{\text{rel}}(\underline{b_j}) + \sum_{k < p} \sigma_{\text{rel}}(u_k) \\
&= \sum_{i < m} \nu_{\text{rel}}(\underline{a_i} \cdot s_i) + \sum_{j < n} \nu_{\text{rel}}(\underline{b_j}) + \sum_{k < p} \sigma_{\text{rel}}(u_k) \\
&= \nu_{\text{rel}}\left(\sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j}\right) + \sigma_{\text{rel}}\left(\sum_{k < p} u_k\right) \\
&= \nu_{\text{rel}}\left(\sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} + \sum_{k < p} \sigma_{\text{rel}}(u_k)\right) + \sigma_{\text{rel}}\left(\sum_{k < p} u_k\right) \\
&= \nu_{\text{rel}}(t) + \sigma_{\text{rel}}(s)
\end{aligned}$$

Where we define:

$$s \equiv \sum_{k < p} u_k$$

Note that $n(s) < n(t)$ is now trivially satisfied, as for every summand u_k of s , there is a corresponding summand $\sigma_{\text{rel}}(u_k)$ of t , and at least one such summand exists as $p \geq 1$.

■

☞ The main use of Lemma 3.2.2.10 will be in induction proofs regarding the (not yet treated) process algebras PA_{drt}^- -ID and $\text{ACP}_{\text{drt}}^-$ -ID.

3.2.3 $\text{BPA}_{\text{drt}}^-$

In this section, we extend $\text{BPA}_{\text{drt}}^-$ -ID with the immediate deadlock constant. The resulting process algebra is called $\text{BPA}_{\text{drt}}^-$.

Definition 3.2.3.1 (Signature of $\text{BPA}_{\text{drt}}^-$)

The signature of $\text{BPA}_{\text{drt}}^-$ consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *immediate deadlock constant* δ , the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , and the *“now” operator* ν_{rel} .

Definition 3.2.3.2 (Axioms of $\text{BPA}_{\text{drt}}^-$)

The process algebra $\text{BPA}_{\text{drt}}^-$ is axiomatized by the axioms of $\text{BPA}_{\text{drt}}^-$ -ID given in Definition 3.2.2.2 on page 47, Axioms A6ID–A7ID shown in Table 2.10 on page 19, and Axioms DRTSID and DCSID shown in Table 3.7 on the next page: $\text{BPA}_{\text{drt}}^- = \text{A1-A5} + \text{A6ID} + \text{A7ID} + \text{DRT1-DRT5} + \text{DRTSID} + \text{DCS1-DCS4} + \text{DCSID}$.

☞ The axioms of $\text{BPA}_{\text{drt}}^-$ are a straightforward combination of the axioms we have already encountered in $\text{BPA}_{\text{drt}}^-$ -ID and BPA_{δ} . We only need to introduce two new axioms to define the time-unit delay and the “now” operator on the immediate deadlock. The first, Axiom DRTSID, expresses that an immediate deadlock in the next time-slice is equivalent to an undelayable deadlock in the current time-slice: we identify the end point of

$$\begin{array}{ll} \sigma_{\text{rel}}(\dot{\delta}) = \underline{\underline{\delta}} & \text{DRTSID} \\ \nu_{\text{rel}}(\dot{\delta}) = \dot{\delta} & \text{DCSID} \end{array}$$

Table 3.7: Additional axioms for $\text{BPA}_{\text{drt}}^-$.

the current time-slice with the begin point of the next time-slice. The second, Axiom DCSID, expresses that the “now” operator leaves the immediate deadlock untouched, as the immediate deadlock cannot move to a following time-slice.

Remark 3.2.3.3 (Derivability of DRT3 and DRT5)

Note that from Axioms A6ID, DRT1, DRT2, and DRTSID we can derive Axioms DRT3 and DRT5, as we have:

$$\underline{\underline{\delta}} \cdot x = \sigma_{\text{rel}}(\dot{\delta}) \cdot x = \sigma_{\text{rel}}(\dot{\delta} \cdot x) = \sigma_{\text{rel}}(\dot{\delta}) = \underline{\underline{\delta}}$$

and, similarly:

$$\sigma_{\text{rel}}(x) + \underline{\underline{\delta}} = \sigma_{\text{rel}}(x) + \sigma_{\text{rel}}(\dot{\delta}) = \sigma_{\text{rel}}(x + \dot{\delta}) = \sigma_{\text{rel}}(x)$$

However, we still choose to include DRT3 and DRT5 even for process algebras that do contain $\dot{\delta}$, as our goal is not to find a *minimal* axiomatization, but instead to find a *convenient* one with regard to ease of proofs and calculations.

Earlier, in the axiomatization of PA_ε (see Section 2.4.3 on page 24), we chose to drop Axiom M2 because it had become derivable. Here, we choose to maintain two derivable axioms, Axioms DRT3 and DRT5. This difference can be explained as follows. First, in PA_ε , we could derive Axiom M2 in a *systematic way*, namely by substituting ε for x in its “twin axiom”. Here, the derivation is not so systematic, but more like a trick. Secondly, as we will see in Chapters 4 and 5, maintaining Axiom M2 would lead to several annoying proof obligations with respect to proving soundness. Proving the soundness of Axioms DRT3 and DRT5 is much easier, as they only involve simple operators.

Concluding: we will maintain derivable axioms when the cost in additional proof obligations is low, and their derivability is “accidental”. However, a derivable axiom that can be systematically derived from the other axioms, and also has a high cost in additional proof obligations, will be dropped as soon as possible.

Definition 3.2.3.4 (Semantics of $\text{BPA}_{\text{drt}}^-$)

The semantics of $\text{BPA}_{\text{drt}}^-$ are given by the term-deduction system $T(\text{BPA}_{\text{drt}}^-)$ induced by the deduction rules for $\text{BPA}_{\text{drt}}^-$ -ID given in Definition 3.2.2.4 on page 48 *minus* the deduction rule $\sigma_{\text{rel}}(x) \xrightarrow{\sigma} x$, *plus* the deduction rules shown in Table 3.8 on the facing page, and the deduction rules shown in Table 3.9 on the next page.

☞ The deduction rules of Table 3.8 on the facing page are exactly the same as in the untimed case, so we only comment on the rules of Table 3.9 on the next page.

In view of our identification of the end point of the current time-slice with the begin point of the next time-slice, we cannot maintain the deduction rule $\sigma_{\text{rel}}(x) \xrightarrow{\sigma} x$. Otherwise, $\sigma_{\text{rel}}(\dot{\delta})$ would be able to do a σ -step, while $\underline{\underline{\delta}}$ would not, and that would violate the

$$\text{ID}(\delta) \quad \frac{\text{ID}(x), \text{ID}(y)}{\text{ID}(x + y)} \quad \frac{\text{ID}(x)}{\text{ID}(x \cdot y)}$$

Table 3.8: Deduction rules for immediate-deadlock predicate.

$$\frac{\neg \text{ID}(x)}{\sigma_{\text{rel}}(x) \xrightarrow{\sigma} x} \quad \frac{\text{ID}(x)}{\text{ID}(v_{\text{rel}}(x))}$$

Table 3.9: Additional deduction rules for immediate-deadlock predicate and discrete time.

equality expressed by Axiom DRTSID. So, we weaken it such that $\sigma_{\text{rel}}(x)$ can only do a σ -step if x is not derivably equal to immediate deadlock.

Finally, we add one new rule to define the immediate deadlock predicate on the “now” operator: if x has immediate deadlock, it also has immediate deadlock when we restrict ourselves to the part that starts in the current time slice.

Definition 3.2.3.5 (Bisimulation for $\text{BPA}_{\text{drt}}^-$)

Bisimulation for $\text{BPA}_{\text{drt}}^-$ is defined as follows; a binary relation R on closed $\text{BPA}_{\text{drt}}^-$ terms is a bisimulation if the following transfer conditions hold for all closed $\text{BPA}_{\text{drt}}^-$ terms p and q :

- (i). If $R^S(p, q)$ and $T(\text{BPA}_{\text{drt}}^-) \models p \xrightarrow{a} p'$, where $a \in A$, then there exists a closed term q' , such that $T(\text{BPA}_{\text{drt}}^-) \models q \xrightarrow{a} q'$ and $R^S(p', q')$,
- (ii). if $R^S(p, q)$ and $T(\text{BPA}_{\text{drt}}^-) \models p \xrightarrow{\sigma} p'$, then there exists a closed term q' , such that $T(\text{BPA}_{\text{drt}}^-) \models q \xrightarrow{\sigma} q'$ and $R^S(p', q')$,
- (iii). if $R^S(p, q)$ and $T(\text{BPA}_{\text{drt}}^-) \models p \xrightarrow{a} \surd$, where $a \in A$, then $T(\text{BPA}_{\text{drt}}^-) \models q \xrightarrow{a} \surd$,
- (iv). if $R^S(p, q)$ and $T(\text{BPA}_{\text{drt}}^-) \models \text{ID}(p)$, then $T(\text{BPA}_{\text{drt}}^-) \models \text{ID}(q)$.

Two $\text{BPA}_{\text{drt}}^-$ terms p and q are bisimilar, notation $p \sim_{\text{BPA}_{\text{drt}}^-} q$, if there exists a bisimulation relation R such that $R(p, q)$.

☞ Bisimulation is defined as expected: we now require the action steps, terminating steps, time steps, and the immediate deadlock predicate all to match for related terms.

Definition 3.2.3.6 (Bisimulation Model for $\text{BPA}_{\text{drt}}^-$)

The bisimulation model for $\text{BPA}_{\text{drt}}^-$ is defined in the same way as for BPA. Replace “BPA” by “ $\text{BPA}_{\text{drt}}^-$ ” in Definition 2.3.1.16 on page 12.

Definition 3.2.3.7 (Basic Terms of $\text{BPA}_{\text{drt}}^-$)

We define $(\sigma, \underline{\delta}, \dot{\delta})$ -basic terms inductively as follows:

- (i). Immediate deadlock $\dot{\delta}$ is a $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic term,
- (ii). if $a \in A_\delta$, then \underline{a} is a $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic term,
- (iii). if $a \in A_\delta$ and t is a $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic term, then $\underline{a} \cdot t$ is a $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic term,
- (iv). if t and s are $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic terms, then $t + s$ is a $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic term,
- (v). if t is a $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic term, then $\sigma_{\text{rel}}(t)$ is a $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic term.

From now on, when we speak of basic terms in the context of $\text{BPA}_{\text{drt}}^-$, we mean $(\sigma, \underline{\underline{\delta}}, \dot{\delta})$ -basic terms.

Definition 3.2.3.8 (Summation Convention with Respect to Immediate Deadlock)

In a setting with immediate deadlock, we will use the convention that a summation over the empty set yields the immediate deadlock:

$$\sum_{i \in \emptyset} t_i \stackrel{\text{def}}{=} \dot{\delta}$$

See Definition 3.2.2.7 on page 49 for other aspects of our summation convention.

Definition 3.2.3.9 (Number of Symbols of a $\text{BPA}_{\text{drt}}^-$ term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). We define $n(\dot{\delta}) = 1$,
- (ii). for $a \in A_\delta$, we define $n(\underline{a}) = 1$,
- (iii). for closed $\text{BPA}_{\text{drt}}^-$ terms x and y , we define $n(x + y) = n(x \cdot y) = n(x) + n(y) + 1$,
- (iv). for a closed $\text{BPA}_{\text{drt}}^-$ term x , we define $n(\sigma_{\text{rel}}(x)) = n(\nu_{\text{rel}}(x)) = n(x) + 1$.

3.2.4 $\text{BPA}_{\text{drt}}^+$

In this section, we extend $\text{BPA}_{\text{drt}}^-$ with so-called *delayable actions*. To define delayable actions, we introduce an auxiliary operator: the *unbounded start delay operator*, denoted $[x]^\omega$. Furthermore, we introduce a recursion principle: RSP(USD). The resulting process algebra is called BPA_{drt} without the recursion principle, and $\text{BPA}_{\text{drt}}^+$ with the recursion principle.

Definition 3.2.4.1 (Signature of BPA_{drt})

The signature of BPA_{drt} consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *delayable actions* $\{a \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\underline{\delta}}$, the *delayable deadlock constant* δ , the *immediate deadlock constant* $\dot{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , the “now” operator ν_{rel} , and the *unbounded start delay operator* $[\]^\omega$.

Remark 3.2.4.2 (Delayable Actions)

By lowercase Roman letters from the beginning of the alphabet (a, b, c , etc.) we denote *delayable actions*. These actions are delayable in the sense that they can postpone their execution an arbitrary number of time-slices, whereas undelayable actions vanish when

they fail to execute in the time-slice in which they are initialized. Correspondingly, we also introduce the *delayable deadlock*, denoted δ , the process that can idle indefinitely, but nothing else.

Note that the notation for delayable actions and delayable deadlock clashes with both the notation for symbols from the alphabet A , and that for untimed actions (see Remark 2.3.1.4 on page 7). In BAETEN AND BERGSTRA [21] the notation $ats(a)$ (*any time-slice a*) is used for the delayable action a . This has the advantage of avoiding the confusion described above, but the disadvantage of cluttering up the formulae.

Remark 3.2.4.3 (Unbounded Start Delay)

The *unbounded start delay* of a process x , denoted $\lfloor x \rfloor^\omega$, is the process that behaves as $\nu_{\text{rel}}(x)$, except that it may delay the execution of its initial action (an initial action of $\nu_{\text{rel}}(x)$ that is, *not* one of x) by an arbitrary number of time-slices.

Note that this implies that $\lfloor \sigma_{\text{rel}}(x) \rfloor^\omega$ can never do an action, as $\nu_{\text{rel}}(\sigma_{\text{rel}}(x))$ cannot do any actions. So, only being able to idle, $\lfloor \sigma_{\text{rel}}(x) \rfloor^\omega$ should be equal to δ .

Remark 3.2.4.4 (Iterated Delay)

In BAETEN AND BERGSTRA [25], BAETEN AND RENIERS [35], and KLEIJN [115] we find the *iterated delay* operator, denoted $\sigma_{\text{rel}}^*(x)$, the process that behaves as x , except that it may delay the execution of its initial action by an arbitrary number of time-slices. There, we have that $\sigma_{\text{rel}}^*(\sigma_{\text{rel}}(x))$ *can* do an action, provided x can.

Which one is “better”, the unbounded start delay, or the iterated delay? Our position is that both operators have their advantages and disadvantages; the unbounded start delay being somewhat counterintuitive, but quite useful in practice, while the iterated delay has a cleaner “look and feel” on first sight, an impression however that quickly evaporates once one tries to use it. The outcome is undecided; that we use the unbounded start delay is probably a matter of taste.

Definition 3.2.4.5 (Axioms of BPA_{drt})

The process algebra BPA_{drt} is axiomatized by the axioms of $\text{BPA}_{\text{drt}}^-$ given in Definition 3.2.3.2 on page 51, and Axioms ATS and USD shown in Table 3.10: $\text{BPA}_{\text{drt}} = \text{A1-A5} + \text{A6ID-A7ID} + \text{DRT1-DRT5} + \text{DRTSID} + \text{DCS1-DCS4} + \text{DCSID} + \text{ATS} + \text{USD}$.

$a = \lfloor \underline{a} \rfloor^\omega$	ATS
$\lfloor x \rfloor^\omega = \nu_{\text{rel}}(x) + \sigma_{\text{rel}}(\lfloor x \rfloor^\omega)$	USD

Table 3.10: Additional axioms for BPA_{drt} .

☞ Axiom ATS defines the delayable action a in terms of the undelayable action \underline{a} and the unbounded start delay: a is like \underline{a} , but it may delay its execution for an arbitrary number of time-slices. Axiom USD defines the unbounded start delay: $\lfloor x \rfloor^\omega$ can either execute an initial action of x (expressed by $\nu_{\text{rel}}(x)$), or move on to the next time-slice, and continue as itself (expressed by $\sigma_{\text{rel}}(\lfloor x \rfloor^\omega)$).

Definition 3.2.4.6 (Recursion Principle for $\text{BPA}_{\text{drt}}^+$)

Besides the axioms mentioned in Definition 3.2.4.5, the system $\text{BPA}_{\text{drt}}^+$ also contains the *recursion principle* RSP(USD) shown in Table 3.11. For more information on recursion principles and their status with respect to axioms, see BAETEN AND WEIJLAND [38].

$$y = \nu_{\text{rel}}(x) + \sigma_{\text{rel}}(y) \quad \Longrightarrow \quad y = [x]^\omega \quad \text{RSP(USD)}$$

Table 3.11: Recursive specification principle for unbounded start delay.

☞ We need RSP(USD) because certain equalities cannot be derived from the axioms (for example, $\text{BPA}_{\text{drt}} \not\vdash [\underline{a} + \underline{b}]^\omega = [\underline{a}]^\omega + [\underline{b}]^\omega$), while they do hold in the bisimulation model. RSP(USD) gives us the means to derive such equalities. As we will see later on in Chapters 4 and 5, it does so very systematically: it gives us exactly all equalities that hold in the model and that are not derivable from the axioms.

RSP(USD) is named after the recursion principle RSP (see BAETEN AND WEIJLAND [38]), as it shares with RSP the quality that it guarantees the uniqueness of a certain process ($y = [x]^\omega$, in this case), given a definition of y in terms of itself (in this case: $y = \nu_{\text{rel}}(x) + \sigma_{\text{rel}}(y)$). Note that this has nothing to do with recursion in the sense of defining processes by a system of recursive equations; we merely require an equality to hold between two open process terms.

Alternatively, we could view RSP(USD) not as a recursion principle, but as a *conditional axiom*: an axiom that can only be applied provided a certain condition holds. Viewed that way, the question arises whether we could replace this conditional axiom by a finite number of unconditional axioms. In Chapters 4 and 5 we will examine this issue, and give the restrictions under which this is possible.

Finally, note that axioms like Axioms CF, D1, and D2 on page 29 are not conditional axioms, but *axiom schemes*, as are all axioms that contain an action. The difference is that an axiom scheme merely provides syntactic sugar to denote a finite number of equalities (namely: zero or one for every $a \in A_\delta$) between process terms, while a conditional axiom may capture an infinite number of equalities between process terms that cannot be expressed by a finite number of unconditional axioms.

Remark 3.2.4.7 (Notation $\text{BPA}_{\text{drt}}^+$)

By superscripting a process algebra with a “+” (e.g. $\text{BPA}_{\text{drt}}^+$), we indicate the presence of the recursion principle RSP(USD). Note that in BAETEN AND BERGSTRA [23] and BAETEN AND RENIERS [35] the notation $\text{BPA}_{\text{drt}} + \text{RSP(USD)}$ is used instead of $\text{BPA}_{\text{drt}}^+$. We find that notation cumbersome.

Example 3.2.4.8 (Use of RSP(USD))

Suppose we want to derive the equality $[x]^\omega + [y]^\omega = [x + y]^\omega$. First, we try to rewrite $[x]^\omega + [y]^\omega$ to a process term of the form $\nu_{\text{rel}}(\dots) + \sigma_{\text{rel}}(\dots)$. Consider the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt}} \vdash [x]^\omega + [y]^\omega &= \nu_{\text{rel}}(x) + \sigma_{\text{rel}}([x]^\omega) + \nu_{\text{rel}}(y) + \sigma_{\text{rel}}([y]^\omega) \\ &= \nu_{\text{rel}}(x + y) + \sigma_{\text{rel}}([x]^\omega + [y]^\omega) \end{aligned}$$

Then, we can apply RSP(USD), and we obtain:

$$\text{BPA}_{\text{drt}}^+ \vdash [x]^\omega + [y]^\omega = [x + y]^\omega$$

Definition 3.2.4.9 (Semantics of BPA_{drt})

The semantics of BPA_{drt} are given by the term-deduction system $T(\text{BPA}_{\text{drt}})$ induced by the deduction rules for $\text{BPA}_{\text{drt}}^-$ given in Definition 3.2.3.4 on page 52 and the deduction rules shown in Table 3.12.

$$\begin{array}{ccc} a \xrightarrow{a} \surd & a \xrightarrow{\sigma} a & \delta \xrightarrow{\sigma} \delta \\ \frac{x \xrightarrow{a} x'}{[x]^\omega \xrightarrow{a} x'} & \frac{x \xrightarrow{a} \surd}{[x]^\omega \xrightarrow{a} \surd} & [x]^\omega \xrightarrow{\sigma} [x]^\omega \end{array}$$

Table 3.12: Deduction rules for delayable actions and unbounded start delay.

☞ The deduction rules for delayable actions are as expected. Then deduction rules for the unbounded start delay: first, action steps and terminating steps of x can also be performed by $[x]^\omega$. Secondly, σ -steps of x cannot be performed by $[x]^\omega$, but $[x]^\omega$ can always perform a σ -step to itself. Finally, note that for any closed BPA_{drt} term x , we have that $\text{BPA}_{\text{drt}}^+ \not\vdash [x]^\omega = \delta$. So, $\text{ID}([x]^\omega)$ should never hold, and hence there is no deduction rule for it.

Definition 3.2.4.10 (Bisimulation and Bisimulation Model for BPA_{drt})

Bisimulation for BPA_{drt} and the corresponding bisimulation model are defined in the same way as for $\text{BPA}_{\text{drt}}^-$ and BPA respectively. Replace “ $\text{BPA}_{\text{drt}}^-$ ” by “ BPA_{drt} ” in Definition 3.2.3.5 on page 53 and “ BPA ” by “ BPA_{drt} ” in Definition 2.3.1.16 on page 12.

Definition 3.2.4.11 (Basic Terms of BPA_{drt})

We define $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic terms inductively as follows:

- (i). Immediate deadlock $\dot{\delta}$ is a $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic term,
- (ii). if $a \in A_\delta$, then \underline{a} and a are $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic terms,
- (iii). if $a \in A_\delta$ and t is a $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic term, then $\underline{a} \cdot t$ and $a \cdot t$ are $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic terms,
- (iv). if t and s are $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic terms, then $t + s$ is a $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic term,
- (v). if t is a $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic term, then $\sigma_{\text{rel}}(t)$ is a $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic term.

From now on, if we speak of basic terms in the context of BPA_{drt} , we mean $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic terms.

Theorem 3.2.4.12 (General Form of Basic Terms of BPA_{drt})

Modulo the commutativity and associativity of the $+$, and modulo superfluous δ summands, all basic terms t of BPA_{drt} are of the form:

$$t \equiv \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{b}_j + \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l + \sum_{o < r} \sigma_{rel}(v_o)$$

for $m, n, p, q, r \in \mathbb{N}$, $a_i, b_j, c_k, d_l \in A_\delta$, and basic terms s_i, u_k , and v_o .

Proof Trivial, by inspection of the definition of basic terms, Definition 3.2.4.11 on the preceding page. Observe that the general form of basic terms is closed under the formation rules given in Definition 3.2.4.11. \blacksquare

Remark 3.2.4.13 (General Form of Basic Terms of BPA_{drt})

Note that the case $t = \delta$ is generated when we take $m = n = p = q = r = 0$. See also Definition 3.2.3.8 on page 54.

Proposition 3.2.4.14 (Properties of BPA_{drt}^+ , Part I)

For BPA_{drt} terms x and y , and any $a \in A_\delta$, we have the following equalities:

- (i). $BPA_{drt}^+ \vdash [a]^\omega = a$
- (ii). $BPA_{drt}^+ \vdash [x \cdot y]^\omega = [x]^\omega \cdot y$
- (iii). $BPA_{drt}^+ \vdash [x + y]^\omega = [x]^\omega + [y]^\omega$
- (iv). $BPA_{drt}^+ \vdash [\sigma_{rel}(x)]^\omega = \delta$
- (v). $BPA_{drt}^+ \vdash [\delta]^\omega = \delta$
- (vi). $BPA_{drt} \vdash \nu_{rel}(a) = \underline{a}$
- (vii). $BPA_{drt} \vdash [x]^\omega + \underline{\delta} = [x]^\omega$

Proof

- (i). Consider the following computation:

$$\begin{aligned} BPA_{drt} \vdash a &= [\underline{a}]^\omega \\ &= \nu_{rel}(\underline{a}) + \sigma_{rel}([\underline{a}]^\omega) \\ &= \underline{a} + \sigma_{rel}(a) \\ &= \underline{a} + \underline{\delta} + \sigma_{rel}(a) \\ &= \nu_{rel}(\underline{a}) + \nu_{rel}(\sigma_{rel}([\underline{a}]^\omega)) + \sigma_{rel}(a) \\ &= \nu_{rel}(\underline{a} + \sigma_{rel}([\underline{a}]^\omega)) + \sigma_{rel}(a) \\ &= \nu_{rel}(\nu_{rel}(\underline{a}) + \sigma_{rel}([\underline{a}]^\omega)) + \sigma_{rel}(a) \\ &= \nu_{rel}([\underline{a}]^\omega) + \sigma_{rel}(a) \\ &= \nu_{rel}(a) + \sigma_{rel}(a) \end{aligned}$$

Using RSP(USD), we obtain:

$$BPA_{drt}^+ \vdash a = [a]^\omega$$

(ii). Consider the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt}} \vdash [x]^\omega \cdot y &= (\nu_{\text{rel}}(x) + \sigma_{\text{rel}}([x]^\omega)) \cdot y \\ &= \nu_{\text{rel}}(x) \cdot y + \sigma_{\text{rel}}([x]^\omega) \cdot y \\ &= \nu_{\text{rel}}(x \cdot y) + \sigma_{\text{rel}}([x]^\omega \cdot y) \end{aligned}$$

Using RSP(USD) (with x instantiated by $x \cdot y$ and y by $[x]^\omega \cdot y$), we obtain:

$$\text{BPA}_{\text{drt}}^+ \vdash [x]^\omega \cdot y = [x \cdot y]^\omega$$

(iii). Consider the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt}} \vdash [x]^\omega + [y]^\omega &= \nu_{\text{rel}}(x) + \sigma_{\text{rel}}([x]^\omega) + \nu_{\text{rel}}(y) + \sigma_{\text{rel}}([y]^\omega) \\ &= \nu_{\text{rel}}(x + y) + \sigma_{\text{rel}}([x]^\omega + [y]^\omega) \end{aligned}$$

Using RSP(USD), we obtain:

$$\text{BPA}_{\text{drt}}^+ \vdash [x]^\omega + [y]^\omega = [x + y]^\omega$$

(iv). Consider the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt}} \vdash \delta &= [\underline{\underline{\delta}}]^\omega \\ &= \nu_{\text{rel}}(\underline{\underline{\delta}}) + \sigma_{\text{rel}}([\underline{\underline{\delta}}]^\omega) \\ &= \underline{\underline{\delta}} + \sigma_{\text{rel}}([\underline{\underline{\delta}}]^\omega) \\ &= \sigma_{\text{rel}}(\delta) \\ &= \nu_{\text{rel}}(\sigma_{\text{rel}}(x)) + \sigma_{\text{rel}}(\delta) \end{aligned}$$

Using RSP(USD), we obtain:

$$\text{BPA}_{\text{drt}}^+ \vdash \delta = [\sigma_{\text{rel}}(x)]^\omega$$

(v). Consider the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt}} \vdash \delta &= [\underline{\underline{\delta}}]^\omega \\ &= \nu_{\text{rel}}(\underline{\underline{\delta}}) + \sigma_{\text{rel}}([\underline{\underline{\delta}}]^\omega) \\ &= \underline{\underline{\delta}} + \sigma_{\text{rel}}(\delta) \\ &= \sigma_{\text{rel}}(\delta) \\ &= \dot{\delta} + \sigma_{\text{rel}}(\delta) \\ &= \nu_{\text{rel}}(\dot{\delta}) + \sigma_{\text{rel}}(\delta) \end{aligned}$$

Using RSP(USD) we obtain:

$$\text{BPA}_{\text{drt}}^+ \vdash \delta = [\dot{\delta}]^\omega$$

(vi). Consider the following computation:

$$\begin{aligned}
\text{BPA}_{\text{drt}} \vdash \nu_{\text{rel}}(a) &= \nu_{\text{rel}}(\lfloor \underline{a} \rfloor^\omega) \\
&= \nu_{\text{rel}}(\nu_{\text{rel}}(\underline{a}) + \sigma_{\text{rel}}(\lfloor \underline{a} \rfloor^\omega)) \\
&= \nu_{\text{rel}}(\underline{a} + \sigma_{\text{rel}}(a)) \\
&= \nu_{\text{rel}}(\underline{a}) + \nu_{\text{rel}}(\sigma_{\text{rel}}(a)) \\
&= \underline{a} + \underline{\delta} \\
&= \underline{a}
\end{aligned}$$

So we obtain:

$$\text{BPA}_{\text{drt}} \vdash \nu_{\text{rel}}(a) = \underline{a}$$

(vii). Consider the following computation:

$$\text{BPA}_{\text{drt}} \vdash \lfloor x \rfloor^\omega + \underline{\delta} = \nu_{\text{rel}}(x) + \sigma_{\text{rel}}(\lfloor x \rfloor^\omega) + \underline{\delta} = \nu_{\text{rel}}(x) + \sigma_{\text{rel}}(\lfloor x \rfloor^\omega) = \lfloor x \rfloor^\omega$$

Note the use of Axiom DRT5 in the second step.

■

Remark 3.2.4.15 (Properties of $\text{BPA}_{\text{drt}}^+$, Part I)

The equalities of Proposition 3.2.4.14 on page 58 are not new, but have been described before, see for example NICOLLIN AND SIFAKIS [154].

Proposition 3.2.4.16 (Properties of $\text{BPA}_{\text{drt}}^+$, Part II)

For any BPA_{drt} term x we have the following equality:

$$\text{BPA}_{\text{drt}}^+ \vdash \delta \cdot x = \delta$$

Proof Using Proposition 3.2.4.14(ii) we derive:

$$\text{BPA}_{\text{drt}}^+ \vdash \delta \cdot x = \lfloor \underline{\delta} \rfloor^\omega \cdot x = \lfloor \underline{\delta} \cdot x \rfloor^\omega = \lfloor \underline{\delta} \rfloor^\omega = \delta$$

■

⇒ The properties of Propositions 3.2.4.14 and 3.2.4.16 are all derivable equalities involving delayable actions or the unbounded start delay. We could have given these as axioms, but we try to avoid such axioms wherever possible, as by means of RSP(USD) all such equalities should follow from the “undelayable axioms”. In this way, at the cost of introducing a recursion principle, we gain a substantial reduction in the size of the axiomatization, in a systematic way.

Definition 3.2.4.17 (Number of Symbols of a BPA_{drt} term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). We define $n(\delta) = 1$,
- (ii). for $a \in A_\delta$, we define $n(\underline{a}) = n(a) = 1$,

- (iii). for closed BPA_{drt} terms x and y , we define $n(x + y) = n(x \cdot y) = n(x) + n(y) + 1$,
- (iv). for a closed BPA_{drt} term x , we define $n(\sigma_{\text{rel}}(x)) = n(\nu_{\text{rel}}(x)) = n(\lfloor x \rfloor^\omega) = n(x) + 1$.

Lemma 3.2.4.18 (Representation of $\text{BPA}_{\text{drt}}^+$ Terms)

Let t be a basic term. Then either:

- (i). $\text{BPA}_{\text{drt}}^+ \vdash t = \dot{\delta}$, or,
- (ii). $\text{BPA}_{\text{drt}}^+ \vdash t = \nu_{\text{rel}}(t) + \underline{\underline{\delta}}$, or,
- (iii). $\text{BPA}_{\text{drt}}^+ \vdash t = \lfloor t \rfloor^\omega$, or,
- (iv). there exists a basic term s such that $\text{BPA}_{\text{drt}}^+ \vdash t = \nu_{\text{rel}}(t) + \sigma_{\text{rel}}(s)$ and $n(s) < n(t)$.

Proof Let t be a basic term. By Theorem 3.2.2.8, we may now proceed by case analysis on the form of basic terms. Suppose, by Theorem 3.2.4.12, that t has the following general form:

$$t \equiv \sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l + \sum_{o < r} \sigma_{\text{rel}}(\nu_o)$$

for $m, n, p, q, r \in \mathbb{N}$, $a_i, b_j, c_k, d_l \in A_\delta$, and basic terms s_i, u_l , and ν_o . We distinguish four cases:

- (i). There are no summands: $p = q = m = n = r = 0$.
- (ii). Every, and at least one, summand starts with an undelayable action: $m + n \geq 1$ and $p = q = r = 0$.
- (iii). Every, and at least one, summand starts with a delayable action: $p + q \geq 1$ and $m = n = r = 0$.
- (iv). Neither of the above; there are both summands that start with delayable action and ones the start with undelayable actions, or there are summands that start with the time-unit delay operator: $p + q + r \geq 1$ and $m + n + r \geq 1$

As can be easily seen, this covers all cases. We now prove that the four cases we distinguish correspond to the four cases in the formulation of the theorem:

- (i). We have $p = q = m = n = r = 0$. So, by Definition 3.2.3.8, $t \equiv \dot{\delta}$ and $\text{BPA}_{\text{drt}}^+ \vdash t = \dot{\delta}$.
- (ii). We have $m + n \geq 1$ and $p = q = r = 0$. So:

$$t \equiv \sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{b_j}}$$

for $m, n \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and basic terms s_i . Then we have the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt}}^+ \vdash t &= \sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{b_j}} = \sum_{i < m} \nu_{\text{rel}}(\underline{\underline{a_i}}) \cdot s_i + \sum_{j < n} \nu_{\text{rel}}(\underline{\underline{b_j}}) \\ &= \sum_{i < m} \nu_{\text{rel}}(\underline{\underline{a_i}} \cdot s_i) + \sum_{j < n} \nu_{\text{rel}}(\underline{\underline{b_j}}) = \nu_{\text{rel}} \left(\sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{b_j}} \right) \end{aligned}$$

$$\begin{aligned}
&= \nu_{\text{rel}} \left(\sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} (\underline{\underline{b_j}} + \underline{\underline{\delta}}) \right) = \nu_{\text{rel}} \left(\sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{b_j}} + \underline{\underline{\delta}} \right) \\
&= \nu_{\text{rel}}(t + \underline{\underline{\delta}}) = \nu_{\text{rel}}(t) + \nu_{\text{rel}}(\underline{\underline{\delta}}) = \nu_{\text{rel}}(t) + \underline{\underline{\delta}}
\end{aligned}$$

(iii). We have $p + q \geq 1$ and $m = n = r = 0$. So:

$$t \equiv \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l$$

for $p, q \in \mathbb{N}$, $c_k, d_l \in A_\delta$, and basic terms u_k . Using Proposition 3.2.4.14(i)-(iii) we then have the following computation:

$$\begin{aligned}
\text{BPA}_{\text{drt}}^+ \vdash t &= \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l = \sum_{k < p} \lfloor c_k \rfloor^\omega \cdot u_k + \sum_{l < q} \lfloor d_l \rfloor^\omega \\
&= \sum_{k < p} \lfloor c_k \cdot u_k \rfloor^\omega + \sum_{l < q} \lfloor d_l \rfloor^\omega = \left[\sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l \right]^\omega \\
&= \lfloor t \rfloor^\omega
\end{aligned}$$

(iv). We have $p + q + r \geq 1$ and $m + n + r \geq 1$. So:

$$t \equiv \sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l + \sum_{o < r} \sigma_{\text{rel}}(v_o)$$

for $m, n, p, q, r \in \mathbb{N}$, $a_i, b_j, c_k, d_l \in A_\delta$, and basic terms s_i, u_l , and v_o . Then we have the following computation:

$$\begin{aligned}
\text{BPA}_{\text{drt}}^+ \vdash t &= \sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l + \sum_{o < r} \sigma_{\text{rel}}(v_o) \\
&= \sum_{i < m} \nu_{\text{rel}}(\underline{\underline{a_i}}) \cdot s_i + \sum_{j < n} \nu_{\text{rel}}(\underline{\underline{b_j}}) + \\
&\quad \sum_{k < p} \lfloor \underline{\underline{c_k}} \rfloor^\omega \cdot u_k + \sum_{l < q} \lfloor \underline{\underline{d_l}} \rfloor^\omega + \sum_{o < r} \sigma_{\text{rel}}(v_o) \\
&= \sum_{i < m} \nu_{\text{rel}}(\underline{\underline{a_i}} \cdot s_i) + \sum_{j < n} \nu_{\text{rel}}(\underline{\underline{b_j}}) + \\
&\quad \sum_{k < p} (\nu_{\text{rel}}(\underline{\underline{c_k}}) + \sigma_{\text{rel}}(\lfloor \underline{\underline{c_k}} \rfloor^\omega)) \cdot u_k + \sum_{l < q} (\nu_{\text{rel}}(\underline{\underline{d_l}}) + \sigma_{\text{rel}}(\lfloor \underline{\underline{d_l}} \rfloor^\omega)) + \\
&\quad \sum_{o < r} \sigma_{\text{rel}}(v_o) \\
&= \sum_{i < m} \nu_{\text{rel}}(\underline{\underline{a_i}} \cdot s_i) + \sum_{j < n} \nu_{\text{rel}}(\underline{\underline{b_j}}) + \\
&\quad \sum_{k < p} (\nu_{\text{rel}}(\underline{\underline{c_k}}) + \sigma_{\text{rel}}(c_k)) \cdot u_k + \sum_{l < q} (\nu_{\text{rel}}(\underline{\underline{d_l}}) + \sigma_{\text{rel}}(d_l)) + \\
&\quad \sum_{o < r} \sigma_{\text{rel}}(v_o) \\
&= \sum_{i < m} \nu_{\text{rel}}(\underline{\underline{a_i}} \cdot s_i) + \sum_{j < n} \nu_{\text{rel}}(\underline{\underline{b_j}}) +
\end{aligned}$$

$$\begin{aligned}
& \sum_{k < p} (\nu_{\text{rel}}(\underline{c_k}) \cdot \underline{u_k} + \sigma_{\text{rel}}(c_k) \cdot \underline{u_k}) + \sum_{l < q} (\nu_{\text{rel}}(\underline{d_l}) + \sigma_{\text{rel}}(d_l)) + \\
& \sum_{o < r} \sigma_{\text{rel}}(\nu_o) \\
= & \sum_{i < m} \nu_{\text{rel}}(\underline{a_i} \cdot s_i) + \sum_{j < n} \nu_{\text{rel}}(\underline{b_j}) + \\
& \sum_{k < p} (\nu_{\text{rel}}(\underline{c_k} \cdot \underline{u_k}) + \sigma_{\text{rel}}(c_k \cdot \underline{u_k})) + \sum_{l < q} (\nu_{\text{rel}}(\underline{d_l}) + \sigma_{\text{rel}}(d_l)) + \\
& \sum_{o < r} \sigma_{\text{rel}}(\nu_o) \\
= & \nu_{\text{rel}} \left(\sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} + \sum_{k < p} \underline{c_k} \cdot \underline{u_k} + \sum_{l < q} \underline{d_l} \right) + \\
& \sigma_{\text{rel}} \left(\sum_{k < p} c_k \cdot \underline{u_k} + \sum_{l < q} d_l + \sum_{o < r} \nu_o \right) \\
= & \nu_{\text{rel}} \left(\sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} + \right. \\
& \left. \sum_{k < p} (\underline{c_k} \cdot \underline{u_k} + \sigma_{\text{rel}}(\lfloor \underline{c_k} \rfloor^\omega) \cdot \underline{u_k}) + \sum_{l < q} (\underline{d_l} + \sigma_{\text{rel}}(\lfloor \underline{d_l} \rfloor^\omega)) + \right. \\
& \left. \sum_{o < r} \sigma_{\text{rel}}(\nu_o) \right) + \\
& \sigma_{\text{rel}} \left(\sum_{k < p} c_k \cdot \underline{u_k} + \sum_{l < q} d_l + \sum_{o < r} \nu_o \right) \\
= & \nu_{\text{rel}} \left(\sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} + \right. \\
& \left. \sum_{k < p} (\underline{c_k} + \sigma_{\text{rel}}(\lfloor \underline{c_k} \rfloor^\omega)) \cdot \underline{u_k} + \sum_{l < q} (\underline{d_l} + \sigma_{\text{rel}}(\lfloor \underline{d_l} \rfloor^\omega)) + \right. \\
& \left. \sum_{o < r} \sigma_{\text{rel}}(\nu_o) \right) + \\
& \sigma_{\text{rel}} \left(\sum_{k < p} c_k \cdot \underline{u_k} + \sum_{l < q} d_l + \sum_{o < r} \nu_o \right) \\
= & \nu_{\text{rel}} \left(\sum_{i < m} \underline{a_i} \cdot s_i + \sum_{j < n} \underline{b_j} + \right. \\
& \left. \sum_{k < p} \lfloor \underline{c_k} \rfloor^\omega \cdot \underline{u_k} + \sum_{l < q} \lfloor \underline{d_l} \rfloor^\omega + \sum_{o < r} \sigma_{\text{rel}}(\nu_o) \right) + \\
& \sigma_{\text{rel}} \left(\sum_{k < p} c_k \cdot \underline{u_k} + \sum_{l < q} d_l + \sum_{o < r} \nu_o \right)
\end{aligned}$$

$$\begin{aligned}
&= \nu_{\text{rel}} \left(\sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{b}_j + \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l + \sum_{o < r} \sigma_{\text{rel}}(v_o) \right) + \\
&\quad \sigma_{\text{rel}} \left(\sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l + \sum_{o < r} v_o \right) \\
&= \nu_{\text{rel}}(t) + \sigma_{\text{rel}}(s)
\end{aligned}$$

Where we define:

$$s \equiv \sum_{k < p} c_k \cdot u_k + \sum_{l < q} d_l + \sum_{o < r} v_o$$

Note that $n(s) < n(t)$ is now trivially satisfied: every summand of s also appears as a subterm of t , and by $m + n + r \geq 1$, t must contain summands that do not appear in s . Therefore, t must contain at least 2 more symbols than s . ■

Lemma 3.2.4.19 (Simplified Representation of $\text{BPA}_{\text{drt}}^+$ Terms)

Let t be a basic term. Then either:

- (i). $\text{BPA}_{\text{drt}}^+ \vdash t = \underline{\delta}$, or,
- (ii). $\text{BPA}_{\text{drt}}^+ \vdash t = t + \underline{\underline{\delta}}$.

Proof This lemma follows almost immediately from Lemma 3.2.4.18; case (i) mentioned there corresponds to case (i) here, and cases (ii)–(iv) mentioned there correspond to case (ii) here. We distinguish the four cases from Lemma 3.2.4.18:

- (i). $\text{BPA}_{\text{drt}}^+ \vdash t = \underline{\delta}$.
- (ii). $\text{BPA}_{\text{drt}}^+ \vdash t = \nu_{\text{rel}}(t) + \underline{\underline{\delta}}$. Then we have, using Axiom A3:

$$\text{BPA}_{\text{drt}}^+ \vdash t = \nu_{\text{rel}}(t) + \underline{\underline{\delta}} = \nu_{\text{rel}}(t) + \underline{\underline{\delta}} + \underline{\underline{\delta}} = t + \underline{\underline{\delta}}$$

- (iii). $\text{BPA}_{\text{drt}}^+ \vdash t = [t]^\omega$. Then we have, using Proposition 3.2.4.14(vii):

$$\text{BPA}_{\text{drt}}^+ \vdash t = [t]^\omega = [t]^\omega + \underline{\underline{\delta}} = t + \underline{\underline{\delta}}$$

- (iv). $\text{BPA}_{\text{drt}}^+ \vdash t = \nu_{\text{rel}}(t) + \sigma_{\text{rel}}(s)$. Then we have, using Axiom DRT5:

$$\text{BPA}_{\text{drt}}^+ \vdash t = \nu_{\text{rel}}(t) + \sigma_{\text{rel}}(s) = \nu_{\text{rel}}(t) + \sigma_{\text{rel}}(s) + \underline{\underline{\delta}} = t + \underline{\underline{\delta}}$$

☞ The main use of Lemmata 3.2.4.18 and 3.2.4.19 will be in induction proofs regarding the (yet to be treated) process algebras PA_{drt}^+ and $\text{ACP}_{\text{drt}}^+$. ■

3.3 Properties

In this section, we list an important property that is desirable in a discrete-time process algebra, and give the process algebras that have this property. We give a formal definition, and an intuitive justification why this property is desirable. Furthermore, we give the embeddings between the discrete-time process algebras treated in this chapter. Elimination, soundness, and completeness properties will be treated in Chapter 4.

Property 3.3.1.1 (Time Determinism)

Given a discrete-time process algebra P and a corresponding term-deduction system $T(P)$, we say that $T(P)$ has the *time-determinism* property if for all closed terms x , y , and y' of P we have that:

$$T(P) \models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y' \implies y \equiv y'$$

This property holds for all process algebras described in this chapter.

☞ The time determinism property is quite closely related to time factorization (see Remark 3.2.1.4 on page 42): both state that the progress of time should only influence the state of a process in a limited way. Both weak and strong time factorization are sufficient conditions to ensure time determinism. For every process algebra P that contains time-factorization axioms, the term-deduction system $T(P)$ should exhibit time determinism, lest the corresponding bisimulation model be not sound.

Remark 3.3.1.2 (Embeddings)

The following embeddings hold between the process algebras given in this chapter, and between the and the untimed process algebras:

- (i). $BPA \subseteq BPA_{\text{drt}}^- - \delta$
- (ii). $BPA_{\delta} \subseteq BPA_{\text{drt}}^- - \text{ID}$
- (iii). $BPA_{\delta} \subseteq BPA_{\text{drt}}^-$
- (iv). $BPA_{\delta} \subseteq BPA_{\text{drt}}^+$
- (v). $BPA_{\text{drt}}^- - \delta \subseteq BPA_{\text{drt}}^- - \text{ID} \subseteq BPA_{\text{drt}}^- \subseteq BPA_{\text{drt}} \subseteq BPA_{\text{drt}}^+$

Embedding (i) is achieved by projecting the untimed process a onto the undelayable process \underline{a} for $a \in A$, and everything else onto itself. Embeddings (ii) and (iii) are achieved like embedding (i), but now for $a \in A_{\delta}$. The embeddings of (v) are achieved by projecting everything onto itself.

Embedding (iv) is special, as it can be achieved in two different ways: either by projecting the untimed process a onto the undelayable process \underline{a} for $a \in A_{\delta}$, and everything else onto itself, or by projecting the untimed process a onto the delayable process a for $a \in A_{\delta}$, and everything else onto itself. In the first way, BPA_{δ} is projected onto the first time-slice of BPA_{drt}^+ . In the second way, BPA_{δ} is projected onto the entire time domain.

4

Soundness and Completeness

4.1 Introduction

In this chapter we will give elimination, soundness, and completeness results for several discrete-time process algebras. We restrict ourselves to concrete process algebras (i.e. without a silent action τ , and without the empty process ε), relative time, closed terms (for open terms, i.e. for ω -completeness, see GROOTE [86]), and basic process algebras. We do treat delayable actions and immediate deadlock.

With the recent appearance of BAETEN AND BERGSTRA [24, 25], BAETEN AND VERHOEF [37], and BAETEN AND RENIERS [35], discrete-time process algebra has reached a decent state of maturity. To be more precise: a stable enough state to justify a detailed analysis of the soundness and completeness issues involved with it. To our knowledge, no such results have been published in the context of discrete-time process algebra before. And although we never really doubted the soundness and completeness of the respective process algebra, we felt that it would not hurt to prove these beliefs explicitly. Rightly so: it turned out that the axiomatizations we started out with were neither sound nor complete.

4.2 Techniques

In this section we will give a short overview of the proof techniques we will use to prove elimination, soundness, and completeness. For each, we briefly list the proof outlines we will use, and then describe these outlines in some detail. The proof outlines we give in this section will be used several times in this and later chapters.

4.2.1 Proving Elimination

To prove elimination to basic terms (Property 2.6.1.5 on page 36) for a process algebra P , we must show that for any closed term s of P , there exists a basic term t such that $P \vdash s = t$. To do this, we apply two different techniques:

- (i). Term-rewriting analysis using the lexicographical path ordering method,

(ii). the direct method.

Proof Outline 4.2.1.1 (Elimination: Lexicographical Path Ordering)

Given a process algebra P , we can associate a *term-rewriting system* T with it such that every reduction path in T corresponds to a derivation in P . If we now also carefully construct T such that it is *strongly terminating*, and its *normal forms* are contained in the basic terms of P , then we have a reduction from every closed term s to a basic term t . Since this reduction corresponds to a derivation in P , we have proved elimination.

There are two crucial steps in this method: first, we have to construct a T with the desired properties, and secondly, we have to prove that T does indeed have these properties.

The first step is usually done by turning a select number of axioms into term-rewriting rules by giving them a direction. The requirement that every reduction in T corresponds to a derivation in P is then trivially fulfilled, as every reduction step now corresponds to an application of the corresponding axiom. Strong termination is mostly also fulfilled, although this is not trivial at all. At the least, we have to be careful to avoid axioms like Axiom A1, which would, when turned into term-rewriting rules, lead to a non strongly terminating T . At this point, we may already have a suitable strongly-terminating T , namely, one whose normal forms are basic terms of P . If this is not the case, we need to introduce additional rewriting rules. To ensure that every reduction path of T still corresponds to a derivation of P , we must ensure that the additional rewriting rules each correspond to a derivable equality in P , which must be separately proven.

Now the second step. Once we have constructed T , we must prove it is strongly terminating, and prove that its normal forms are basic terms of P . We prove strong termination using the so-called *lexicographical path ordering technique*, the details of which are far beyond the scope of this thesis. See, for example, KLOP [116] for more information. Finally, the fact that the normal forms of T are indeed basic terms of P , is proven by showing that for every closed term s of P , either s is a basic term of P , or not a normal form of T .

☞ This proof outline is taken from BAETEN AND VERHOEF [37]. We will first use it in the proof of Theorem 4.3.1.2 on page 72.

Proof Outline 4.2.1.2 (Elimination: Direct Method)

For some process algebras, the method outlined in Proof Outline 4.2.1.1 does not work, as the term-rewriting system we arrive at is either not strongly terminating, or does not have basic terms as normal forms, and the term-rewriting system remains so no matter how many additional rewriting rules we add. In these cases, we apply a direct method: we simply prove that for all closed terms elimination can be achieved by examining all possible cases using induction.

To do this, we first prove elimination to a Basic Process Algebra P . Elimination to basic terms then follows then as a corollary, on the basis of an already known elimination (to basic terms) result for P .

Although conceptually simple, this method often gives rise to very lengthy proofs, exponentially so for process algebras that contain many features.

☞ We will first use this proof outline in the proof of Theorem 5.2.1.7 on page 107.

4.2.2 Proving Soundness

To prove soundness (Property 2.6.1.7 on page 36) for a process algebra P with respect to its bisimulation model M , we must show that for all closed terms s and t of P such that $P \vdash s = t$, we also have $M \models s \sim_p t$. To do this, we apply three different techniques:

- (i). The direct method,
- (ii). the indirect method,
- (iii). the ground equivalence method.

Proof Outline 4.2.2.1 (Soundness: Direct Method)

To prove soundness, it is sufficient to prove that each axiom is sound, i.e., prove that for all closed instantiations of the axiom, both sides of the axiom correspond to the same element of the bisimulation model.

Let P be a process algebra. To prove the soundness of a certain axiom of P of the general form:

$$t_l(x_1, \dots, x_n) = t_r(x_1, \dots, x_n)$$

where t_l and t_r are process terms in the free variables x_1, \dots, x_n for some $n \in \mathbb{N}$, with respect to the bisimulation model of P , we proceed as follows. First, we give a relation R , which will be a binary relation on closed terms. Then, we show that this R is a bisimulation relation that for all closed instantiations of x_1, \dots, x_n relates the left-hand and right-hand side of the axiom. This involves two steps:

- (i). R should relate both sides of the axiom for all closed terms, i.e., for all closed instantiations of x_1, \dots, x_n we should have that

$$(t_l(x_1, \dots, x_n), t_r(x_1, \dots, x_n)) \in R.$$

This is mostly so trivial that we do not mention it at all.

- (ii). R should be a bisimulation. In order to prove that, we show that for all closed terms s and t such that $R(s, t)$, the transfer conditions from the definition of bisimulation are satisfied.

For example, in proving soundness of an axiom of $\text{BPA}_{\text{drt}}^- - \delta$, we have to show that for all closed terms s and t such that $(s, t) \in R$, we have that for any transition $s \xrightarrow{u} s'$ (where $u \in A_\sigma$), there is a corresponding transition $t \xrightarrow{u} t'$ such that $(s', t') \in R$, and vice versa, for any transition $t \xrightarrow{u} t'$, there is a corresponding transition $s \xrightarrow{u} s'$ such that again $(s', t') \in R$. This part of the proof is done using case distinction on the different kinds of steps that are possible (an action step, a time step, or a terminating step).

Note that the “vice versa” part of proof obligation (ii) results from the fact that the transfer conditions for bisimulation (see for example Definition 3.2.1.8 on page 45) are defined with respect to the symmetric closure of R .

☞ This proof outline is taken from BAETEN AND VERHOEF [37]. We will first use it in the proof of Theorem 4.3.1.4 on page 73.

Proof Outline 4.2.2.2 (Soundness: Indirect Method)

Next to proving soundness directly, in the manner of Proof Outline 4.2.2.1 on the page before, we can also prove soundness indirectly, by deriving it from the soundness of a related process algebra. Suppose we have a process algebra P that is sound with respect to its bisimulation model M . We now construct a process algebra P' over the same signature, such that all axioms of P' are for closed terms derivable in P . Since all axioms of P were sound with respect to M , necessarily all axioms of P' must also be sound with respect to M . Hence, P' is a sound axiomatization of M .

☞ We will first use this proof outline in the proof of Corollary 4.3.5.4 on page 103.

Proof Outline 4.2.2.3 (Soundness: Ground-Equivalence Method)

Suppose we have two process algebras P and P' with identical signatures, such that P and P' define the same derivable equalities for closed terms. So, for all closed terms s and t over the signature, we have $P \vdash s = t$ iff $P' \vdash s = t$. Such P and P' are said to be *axiom ground equivalent*. Furthermore, suppose that the same equalities also hold in the bisimulation models M and M' of P and P' , i.e., for all closed terms s and t over the signature, we have $s \sim_P t$ iff $s \sim_{P'} t$. Then, P and P' are also called *term-deduction system ground equivalent*.

If we now have that P is sound and complete with respect to M , then necessarily P' must also be sound and complete with respect to M' .

Note that this outline is a special case of Proof Outline 4.2.2.2. Its added value lies in the fact that if we have P , P' , M , and M' as described above, we can derive both the soundness and completeness of P' with respect to M' at once.

☞ We will first use this proof outline in the proof of Corollary 5.2.2.17 on page 120.

4.2.3 Proving Completeness

To prove completeness (Property 2.6.1.9 on page 37) for a process algebra P with respect to its bisimulation model M , we must show that for all closed terms s and t of P such that $M \models s \sim_P t$, we also have $P \vdash s = t$. To do this, we apply four different techniques:

- (i). The direct method,
- (ii). the indirect method,
- (iii). the ground equivalence method,
- (iv). Verhoef's method.

Proof Outline 4.2.3.1 (Completeness: Direct Method)

Given a process algebra P , we first derive a lemma “Towards Completeness of P ” that contains sublemmata of the general form:

$$M \models \dots \implies P \vdash \dots$$

Typically, each sublemma relates a certain transition in $T(P)$ with a certain equality in P (some sublemmata deviate slightly from this format).

Armed with the implications proven in the lemma, we then set out to actually prove completeness. This is done by proving that for all closed terms s and t of P we have that:

$$M \models s \sim_p t \implies P \vdash s = t$$

As we will show later, in Lemma 4.3.1.6 on page 76, a sufficient condition for this is that for all closed terms s and t of P we have that:

$$M \models s + t \sim_p t \implies P \vdash s + t = t$$

The last implication is proven by induction on the number of symbols in s , using case distinction on the form of s , which we may assume to be a basic term if P has the elimination property. The “Towards . . . ” sublemmata are chosen in such a way that each case we encounter in completing our proof is now easily handled.

☞ This proof outline is taken from BAETEN AND VERHOEF [37]. We will first use it in the proof of Theorem 4.3.1.8 on page 78.

Proof Outline 4.2.3.2 (Completeness: Indirect Method)

Next to proving completeness directly, in the manner of Proof Outline 4.2.3.1 on the preceding page, we can also prove completeness indirectly, by deriving it from the completeness of a related process algebra. Suppose we have a process algebra P that is complete with respect to its bisimulation model M . We now construct a process algebra P' over the same signature, such that all derivable equalities of P that are used in the completeness proof of P , are axioms of P' . In this way, the completeness proof for P is also a valid completeness proof of P' , as all assumptions about derivability in P that the proof uses, are also valid in P' . Hence, P' is complete with respect to M . Finally, note that although P' is complete with respect to M , it need not be sound with respect to M .

☞ We will first use this proof outline in the proof of Corollary 4.3.5.5 on page 103.

Proof Outline 4.2.3.3 (Completeness: Ground-Equivalence Method)

Proving completeness using ground-equivalence method is done simultaneously with proving soundness, see Proof Outline 4.2.2.3 on the preceding page for details.

☞ We will first use this proof outline in the proof of Corollary 5.2.2.19 on page 121.

Proof Outline 4.2.3.4 (Completeness: Verhoef’s Method)

Instead of using one of the previous three proof outlines to prove completeness, we can also use the General Completeness Theorem of VERHOEF [195].

In order to apply this theorem to prove the completeness of a process algebra P with respect to its bisimulation model M , we need a process algebra P' such that P is a conservative extension of P' . Furthermore, for every closed term s of P , there should be a closed term t of P' such that $P \vdash s = t$ (sometimes expressed as: P has the elimination property for P').

If we now have that P' is complete with respect to its bisimulation model M' , P is sound with respect to M , and $T(P)$ is an operationally conservative extension of $T(P')$, then by Verhoef’s General Completeness Theorem, P is also complete with respect to M .

For further details of this method, and a definition of conservativity with respect to term-deduction systems, see [195].

☞ We will first use this proof outline in the proof of 5.2.1.14 on page 112.

4.3 Results

In this section, we will prove soundness and completeness results for the discrete-time process algebras we introduced in Section 3.2, $BPA_{drt}^- - \delta$, $BPA_{drt}^- - ID$, BPA_{drt}^- , and BPA_{drt}^+ , and for one new one, BPA'_{drt} , a variation on BPA_{drt}^+ .

4.3.1 $BPA_{drt}^- - \delta$

We start by proving elimination, soundness, and completeness for $BPA_{drt}^- - \delta$.

Remark 4.3.1.1 (Shorthand for Time-Unit Delay Operator)

The *relative time* time-unit delay operator $\sigma_{rel}(x)$ we introduced in Chapter 3 has an *absolute time* counterpart $\sigma_{abs}(x)$ (BAETEN AND BERGSTRA [24]), which is why the $_{rel}$ subscript is needed to distinguish between the two.

However, as absolute time plays no rôle in this thesis, we will often allow ourselves to write $\sigma(x)$ as a shorthand for $\sigma_{rel}(x)$. In axioms and deduction rules we will still write the unabbreviated $\sigma_{rel}(x)$ form, as to make a comparison with, for example, the axioms of BAETEN AND BERGSTRA [24] not too confusing. A similar convention holds with respect to the “now” operator $\nu_{rel}(x)$, which we will often replace by the shorthand $\nu(x)$.

Theorem 4.3.1.2 (Elimination for $BPA_{drt}^- - \delta$)

Let t be a closed $BPA_{drt}^- - \delta$ term. Then there is a basic term s such that $BPA_{drt}^- - \delta \vdash s = t$.

Proof We use the lexicographical path ordering method we described in Proof Outline 4.2.1.1 on page 68. We select Axioms A4, A5, and DRT2 of $BPA_{drt}^- - \delta$ to be turned into the term-rewriting system for $BPA_{drt}^- - \delta$ shown in Table 4.1. The well-founded or-

$(x + y) \cdot z \rightarrow x \cdot z + y \cdot z$	RA4
$(x \cdot y) \cdot z \rightarrow x \cdot (y \cdot z)$	RA5
$\sigma(x) \cdot y \rightarrow \sigma(x \cdot y)$	RDRT2

Table 4.1: Term-Rewriting System for $BPA_{drt}^- - \delta$.

dering $>$ on constants and function symbols is the following:

$$\underline{a} < \sigma < + < \cdot$$

To \cdot we assign the lexicographical status for the first argument. Now we show that the left-hand side of every rewriting rule is bigger than the right-hand side with respect to the ordering \succ_{lpo} . This is done by the following reductions (taken from BAETEN AND VERHOEF [37]):

$$\begin{aligned} (x + y) \cdot z &\succ_{lpo} (x + y) \cdot \star z \succ_{lpo} (x + y) \cdot \star z + (x + y) \cdot \star z \succ_{lpo} (x + \star y) \cdot z + (x + \star y) \cdot z \\ &\succ_{lpo} x \cdot z + y \cdot z \end{aligned}$$

$$\begin{aligned}
(x \cdot y) \cdot z &\succ_{\text{lpo}} (x \cdot y) \cdot *z \succ_{\text{lpo}} (x \cdot *y) \cdot ((x \cdot y) \cdot *z) \succ_{\text{lpo}} x \cdot ((x \cdot *y) \cdot z) \\
&\succ_{\text{lpo}} x \cdot (y \cdot z) \\
\sigma(x) \cdot y &\succ_{\text{lpo}} \sigma(x) \cdot *y \succ_{\text{lpo}} \sigma(\sigma(x) \cdot *y) \succ_{\text{lpo}} \sigma(\sigma^*(x) \cdot y) \\
&\succ_{\text{lpo}} \sigma(x \cdot y)
\end{aligned}$$

Next, we will prove that the closed normal forms of this term-rewriting system are basic terms. Suppose that s is a normal form, furthermore, suppose that s is not a basic term. Let s' denote the smallest subterm of s which is not a basic term. Note that, consequently, all proper subterms of s' are basic terms. Then we can prove that s' is not a normal form by case analysis. We distinguish all possible cases:

- (i). s' is an undelayable action. But then s' is a basic term. This is in contradiction with the assumption that s' is not a basic term, so this case does not occur.
- (ii). s' is of the form $s'_1 \cdot s'_2$ for basic terms s'_1 and s'_2 . With case analysis on the structure of basic term s'_1 :
 - (a) If s'_1 is an undelayable action \underline{a} then $s'_1 \cdot s'_2$ is a basic term, and so s' is a basic term which again contradicts the assumption that s' is not a basic term. This case can therefore not occur.
 - (b) If s'_1 is of the form $\underline{a} \cdot t$ for some undelayable action \underline{a} and basic term t , then rewriting rule RA5 can be applied. So, s' is not a normal form.
 - (c) If s'_1 is of the form $t_1 + t_2$ for t_1 and t_2 basic terms. Then rewriting rule RA4 is applicable. Therefore, s' is not a normal form.
 - (d) If s'_1 is of the form $\sigma(t)$ for some basic term t . Then rewriting rule RDRT2 is applicable. So, s' is not a normal form.
- (iii). s' is of the form $s'_1 + s'_2$ for basic terms s'_1 and s'_2 . In this case s' would be a basic term, which contradicts the assumption that s' is not a basic term. Therefore, this case cannot occur.
- (iv). s' is of the form $\sigma(s'')$ for some basic term s'' . But then s' is basic term too, so the case does not occur.

In any case that can occur it follows that s' is not a normal form. Since s' is a subterm of s , we conclude that s is not a normal form. This contradicts the assumption that s is a normal form. From this contradiction we conclude that s is a basic term, which completes the proof. ■

Remark 4.3.1.3 (Elimination for $\text{BPA}_{\text{drt}}^- - \delta$)

Elimination for $\text{BPA}_{\text{drt}}^- - \delta$ is also claimed (without proof) in Theorem 2.12.3 of BAETEN AND VERHOEF [37] (where $\text{BPA}_{\text{drt}}^- - \delta$ is called BPA_{dt}).

Theorem 4.3.1.4 (Soundness of $\text{BPA}_{\text{drt}}^- - \delta$)

The set of closed $\text{BPA}_{\text{drt}}^- - \delta$ terms modulo bisimulation equivalence is a model of $\text{BPA}_{\text{drt}}^- - \delta$.

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. Since bisimulation equivalence is a congruence, also for the new operators, we only need to

verify the soundness of every closed instantiation of the axioms. We do this by giving a relation R for every axiom and we prove that this relation is a bisimulation relation for every closed instantiation of the left-hand and right-hand sides of the axiom.

In the setting of BPA soundness of Axioms A1–A5 has already been proven (see for example Theorem 2.2.33 of BAETEN AND VERHOEF [37]).

The process algebra $\text{BPA}_{\text{drt}}^- - \delta$ adds to this the possibility to perform σ -transitions. However, any term headed by the time-unit delay operator added to BPA is not capable of performing an action step. So, we argue that the left-hand side and the right-hand side of Axioms A1–A5 can perform exactly the same action steps in $\text{BPA}_{\text{drt}}^- - \delta$ as in BPA. Therefore, we only consider the transitions labeled by σ for Axioms A1–A5. For Axioms DRT1–DRT2 we, of course, have to consider both the action steps and the time steps.

Finally, note that $\text{BPA}_{\text{drt}}^- - \delta$ contains undelayable actions of the form \underline{a} whereas BPA had untimed actions of the form a . This is however not relevant for the purpose of extending the soundness proofs of Axioms A1–A5 from BPA to $\text{BPA}_{\text{drt}}^- - \delta$, as neither a nor \underline{a} appears in Axioms A1–A5, and both have exactly the same deduction rules (see Table 2.3 on page 10 and Table 3.2 on page 44).

Axiom A1 Take the relation:

$$R = \{(s, s), (s + t, t + s) \mid s, t \in C(\text{BPA}_{\text{drt}}^- - \delta)\}$$

First we look at the transitions of the left-hand side. Suppose $s + t \xrightarrow{\sigma} p$. Then one of the following situations occurs:

- (i). $s \xrightarrow{\sigma} p$ and $t \not\xrightarrow{\sigma}$: then also $t + s \xrightarrow{\sigma} p$, and note that $(p, p) \in R$.
- (ii). $s \not\xrightarrow{\sigma}$ and $t \xrightarrow{\sigma} p$: then also $t + s \xrightarrow{\sigma} p$, and note that $(p, p) \in R$.
- (iii). $s \xrightarrow{\sigma} p_1$ and $t \xrightarrow{\sigma} p_2$ and $p \equiv p_1 + p_2$: then $t + s \xrightarrow{\sigma} p_2 + p_1$, and note that $(p_1 + p_2, p_2 + p_1) \in R$.

The proof for the right-hand side is analogous.

Axiom A2 Take the relation:

$$R = \{(s, s), ((s + t) + u, s + (t + u)) \mid s, t, u \in C(\text{BPA}_{\text{drt}}^- - \delta)\}$$

First we look at the transitions of the left-hand side. Suppose $(s + t) + u \xrightarrow{\sigma} p$. Then one of the following situations occurs:

- (i). $s + t \xrightarrow{\sigma} p$ and $u \not\xrightarrow{\sigma}$: then the transition $s + t \xrightarrow{\sigma} p$ must be due to one of the following:
 - (a) $s \xrightarrow{\sigma} p$ and $t \not\xrightarrow{\sigma}$: in that case $s \xrightarrow{\sigma} p$ and $t + u \not\xrightarrow{\sigma}$. So, $s + (t + u) \xrightarrow{\sigma} p$, and note that $(p, p) \in R$.
 - (b) $s \not\xrightarrow{\sigma}$ and $t \xrightarrow{\sigma} p$: in that case $s \not\xrightarrow{\sigma}$ and $t + u \xrightarrow{\sigma} p$. Therefore, $s + (t + u) \xrightarrow{\sigma} p$.
 - (c) $s \xrightarrow{\sigma} p_1$ and $t \xrightarrow{\sigma} p_2$ and $p \equiv p_1 + p_2$: in that case $s \xrightarrow{\sigma} p_1$ and $t + u \xrightarrow{\sigma} p_2$. Therefore, $s + (t + u) \xrightarrow{\sigma} p_1 + p_2$, and note that $(p, p) \in R$.
- (ii). $s + t \not\xrightarrow{\sigma}$ and $u \xrightarrow{\sigma} p$: then $s \not\xrightarrow{\sigma}$, $t \not\xrightarrow{\sigma}$, and $u \xrightarrow{\sigma} p$. So, $s \not\xrightarrow{\sigma}$ and $t + u \xrightarrow{\sigma} p$. Therefore, $s + (t + u) \xrightarrow{\sigma} p$, and note that $(p, p) \in R$.

- (iii). $s + t \xrightarrow{\sigma} p_1$ and $u \xrightarrow{\sigma} p_2$ and $p \equiv p_1 + p_2$: then the transition $s + t \xrightarrow{\sigma} p_1$ must be due to one of the following:
- (a) $s \xrightarrow{\sigma} p_1$ and $t \not\xrightarrow{\sigma}$: in that case $s \xrightarrow{\sigma} p_1$ and $t + u \xrightarrow{\sigma} p_2$. So, $s + (t + u) \xrightarrow{\sigma} p_1 + p_2$, and note that $(p, p) \in R$.
 - (b) $s \not\xrightarrow{\sigma}$ and $t \xrightarrow{\sigma} p_1$: in that case $s \not\xrightarrow{\sigma}$ and $t + u \xrightarrow{\sigma} p_1 + p_2$. Therefore, $s + (t + u) \xrightarrow{\sigma} p_1 + p_2$, and note that $(p, p) \in R$.
 - (c) $s \xrightarrow{\sigma} q_1$ and $t \xrightarrow{\sigma} q_2$ and $p_1 \equiv q_1 + q_2$: in that case $s \xrightarrow{\sigma} q_1$ and $t + u \xrightarrow{\sigma} q_2 + p_2$. Therefore, $s + (t + u) \xrightarrow{\sigma} q_1 + (q_2 + p_2)$, and note that $((q_1 + q_2) + p_2, q_1 + (q_2 + p_2)) \in R$.

The proof for the right-hand side is analogous.

Axiom A3 Take the relation:

$$R = \{(s, s), (s + s, s) \mid s \in C(\text{BPA}_{\text{drt}}^- - \delta)\}$$

First we look at the transitions of the left-hand side. Suppose $s + s \xrightarrow{\sigma} p$. Then $s \xrightarrow{\sigma} p'$ and $p \equiv p' + p'$. Then $s \xrightarrow{\sigma} p'$, and note that $(p' + p', p') \in R$.

The proof for the right-hand side is analogous.

Axiom A4 Take the relation:

$$R = \{(s, s), ((s + t) \cdot u, s \cdot u + t \cdot u) \mid s, t, u \in C(\text{BPA}_{\text{drt}}^- - \delta)\}$$

First we look at the transitions of the left-hand side. Suppose $(s + t) \cdot u \xrightarrow{\sigma} p$. Then $s + t \xrightarrow{\sigma} p'$ and $p \equiv p' \cdot u$. The transition $s + t \xrightarrow{\sigma} p'$ must be due to one of the following:

- (i). $s \xrightarrow{\sigma} p'$ and $t \not\xrightarrow{\sigma}$: then $s \cdot u + t \cdot u \xrightarrow{\sigma} p' \cdot u$, and note that $(p, p) \in R$.
- (ii). $s \not\xrightarrow{\sigma}$ and $t \xrightarrow{\sigma} p'$: analogous to the previous case.
- (iii). $s \xrightarrow{\sigma} p_1$ and $t \xrightarrow{\sigma} p_2$ and $p' \equiv p_1 + p_2$: then $s \cdot u + t \cdot u \xrightarrow{\sigma} p_1 \cdot u + p_2 \cdot u$, and note that $((p_1 + p_2) \cdot u, p_1 \cdot u + p_2 \cdot u) \in R$.

Secondly, we look at the transitions of the right-hand side. Suppose $s \cdot u + t \cdot u \xrightarrow{\sigma} p$. This must be due to one of the following:

- (i). $s \cdot u \xrightarrow{\sigma} p$ and $t \cdot u \not\xrightarrow{\sigma}$: then $s \xrightarrow{\sigma} p'$ and $p \equiv p' \cdot u$. Also $t \not\xrightarrow{\sigma}$. Therefore, $(s + t) \xrightarrow{\sigma} p'$ and $(s + t) \cdot u \xrightarrow{\sigma} p' \cdot u$, and note that $(p, p) \in R$.
- (ii). $s \cdot u \not\xrightarrow{\sigma}$ and $t \cdot u \xrightarrow{\sigma} p$: analogous to the previous case.
- (iii). $s \cdot u \xrightarrow{\sigma} p_1$ and $t \cdot u \xrightarrow{\sigma} p_2$ and $p \equiv p_1 + p_2$: then $s \xrightarrow{\sigma} q_1$ and $p_1 \equiv q_1 \cdot u$ and $t \xrightarrow{\sigma} q_2$ and $p_2 \equiv q_2 \cdot u$. Therefore, $(s + t) \xrightarrow{\sigma} q_1 + q_2$ and $(s + t) \cdot u \xrightarrow{\sigma} (q_1 + q_2) \cdot u$, and note that $((q_1 + q_2) \cdot u, q_1 \cdot u + q_2 \cdot u) \in R$.

Axiom A5 Take the relation:

$$R = \{(s, s), ((s \cdot t) \cdot u, s \cdot (t \cdot u)) \mid s, t, u \in C(\text{BPA}_{\text{drt}}^- - \delta)\}$$

First we look at the transitions of the left-hand side. Suppose $(s \cdot t) \cdot u \xrightarrow{\sigma} p$. Then this must be due to $s \xrightarrow{\sigma} p'$ and $p \equiv (p' \cdot t) \cdot u$. So, $s \cdot (t \cdot u) \xrightarrow{\sigma} p' \cdot (t \cdot u)$, and note that $((p' \cdot t) \cdot u, p' \cdot (t \cdot u)) \in R$.

The proof for the right-hand side is analogous.

Axiom DRT1 Take the relation:

$$R = \{(s, s), (\sigma(s) + \sigma(t), \sigma(s+t)) \mid s, t \in C(\text{BPA}_{\text{drt}}^- - \delta)\}$$

We look at the transitions of both sides at the same time. First, note that $\sigma(s) + \sigma(t) \xrightarrow{a}$ and $\sigma(s+t) \xrightarrow{a}$. Secondly, $\sigma(s) + \sigma(t) \xrightarrow{\sigma} p$ iff $p \equiv s+t$ iff $\sigma(s+t) \xrightarrow{\sigma} p$, and note that $(p, p) \in R$.

Axiom DRT2 Take the relation:

$$R = \{(s, s), (\sigma(s) \cdot t, \sigma(s \cdot t)) \mid s, t \in C(\text{BPA}_{\text{drt}}^- - \delta)\}$$

We look at the transitions of both sides at the same time. First, note that $\sigma(s) \cdot t \xrightarrow{a}$ and $\sigma(s \cdot t) \xrightarrow{a}$. Secondly, $\sigma(s) \cdot t \xrightarrow{\sigma} p$ iff $p \equiv s \cdot t$ iff $\sigma(s \cdot t) \xrightarrow{\sigma} p$, and note that $(p, p) \in R$.

■

Remark 4.3.1.5 (Soundness of $\text{BPA}_{\text{drt}}^- - \delta$)

Soundness of $\text{BPA}_{\text{drt}}^- - \delta$ is also claimed (without proof) in Theorem 2.12.4 of BAETEN AND VERHOEF [37] (where $\text{BPA}_{\text{drt}}^- - \delta$ is called BPA_{dt}).

Lemma 4.3.1.6 (Sufficient Condition for Completeness)

Let P be a process algebra that contains Axiom A3, \sim_p the corresponding bisimulation congruence, and M the corresponding bisimulation model. Now, in order to prove that for all closed terms s and t of P we have that:

$$s \sim_p t \implies T \vdash s = t \tag{*}$$

it is sufficient to prove that:

$$s + t \sim_p t \implies T \vdash s + t = t. \tag{\dagger}$$

Proof Let s and t be closed terms of P . Now assume (\dagger) and the left-hand side of $(*)$ to hold. Then prove the right-hand side of $(*)$. This is done as follows: by the soundness of P with respect to M , the fact that \sim_p is a congruence, and Axiom A3, we have $s + t \sim_p t + t \sim_p t$ and $t + s \sim_p s + s \sim_p s$. Therefore, by (\dagger) , also $T \vdash s + t = t$ and $T \vdash t + s = s$. But that gives us $T \vdash s = t + s = s + t = t$, and we are done. ■

Lemma 4.3.1.7 (Towards Completeness of $\text{BPA}_{\text{drt}}^- - \delta$)

Let x and y be closed $\text{BPA}_{\text{drt}}^- - \delta$ terms. Then we have:

- (i). $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} \surd \implies \text{BPA}_{\text{drt}}^- - \delta \vdash x = \underline{a} + x$,
- (ii). $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} y \implies \text{BPA}_{\text{drt}}^- - \delta \vdash x = \underline{a} \cdot y + x$,
- (iii). $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{\sigma} y \implies \text{BPA}_{\text{drt}}^- - \delta \vdash x = \sigma(y) + x$,
- (iv). $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} y \implies n(x) > n(y)$,
- (v). $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{\sigma} y \implies n(x) > n(y)$.

Proof For parts (i), (ii), and (iii) we assume, by Theorem 4.3.1.2 and Theorem 4.3.1.4, without loss of generality, that x is a basic term, and apply induction on the structure of basic terms. For parts (iv) and (v) this does not work, as bisimulation obviously is not a congruence for $n(x)$. Therefore, in proving them we use induction on the general structure of terms.

- (i). Case 1: x is an undelayable action. Because $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} \surd$, it must then be the case that $x \equiv a$. So we have $\text{BPA}_{\text{drt}}^- - \delta \vdash x = a = a + a = a + x$. Case 2: x is of the form undelayable action followed by another basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} \surd$, so this case does not occur. Case 3: x is of the form $s + t$, where s and t are again basic terms. As $T(\text{BPA}_{\text{drt}}^- - \delta) \models s + t \xrightarrow{a} \surd$, necessarily $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{a} \surd$ or $T(\text{BPA}_{\text{drt}}^- - \delta) \models t \xrightarrow{a} \surd$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- - \delta \vdash s = a + s$ or $\text{BPA}_{\text{drt}}^- - \delta \vdash t = a + t$. But then in both cases $\text{BPA}_{\text{drt}}^- - \delta \vdash x = s + t = a + s + t = a + x$. Case 4: x is of the form $\sigma(s)$ for some basic term s . As we know that $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} \surd$, this case cannot occur.
- (ii). Case 1: x is an undelayable action. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} y$, so this case does not occur. Case 2: x is of the form undelayable action followed by another basic term. Then, because $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} y$, it must be that $x \equiv a \cdot y$. So, $\text{BPA}_{\text{drt}}^- - \delta \vdash x = a \cdot y = a \cdot y + a \cdot y = a \cdot y + x$. Case 3: x is of the form $s + t$, where s and t are again basic terms. As $T(\text{BPA}_{\text{drt}}^- - \delta) \models s + t \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{a} y$ or $T(\text{BPA}_{\text{drt}}^- - \delta) \models t \xrightarrow{a} y$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- - \delta \vdash s = a \cdot y + s$ or $\text{BPA}_{\text{drt}}^- - \delta \vdash t = a \cdot y + t$. So in both cases $\text{BPA}_{\text{drt}}^- - \delta \vdash x = s + t = a \cdot y + s + t = a \cdot y + x$. Case 4: x is of the form $\sigma(s)$ for some basic term s . As we know that $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} y$, this case cannot occur.
- (iii). Case 1: x is an undelayable action. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: x is of the form undelayable action followed by another basic term. For the same reason, this case cannot occur either. Case 3: x is of the form $s + t$ where s and t are again basic terms. As $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{\sigma} y$, we know that either $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{\sigma} y$, or $T(\text{BPA}_{\text{drt}}^- - \delta) \models t \xrightarrow{\sigma} y$, or both. So, by the induction hypothesis, either $\text{BPA}_{\text{drt}}^- - \delta \vdash t = \sigma(y) + t$, or $\text{BPA}_{\text{drt}}^- - \delta \vdash s = \sigma(y) + s$, or both. So in all cases $\text{BPA}_{\text{drt}}^- - \delta \vdash x = s + t = \sigma(y) + s + t = \sigma(y) + x$. Case 4: x is of the form $\sigma(s)$ for some basic term s . Then necessarily $s \equiv y$. So, $\text{BPA}_{\text{drt}}^- - \delta \vdash x = x + x = \sigma(y) + x = \sigma(s) + x$.
- (iv). Case 1: x is an undelayable action. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} y$, so this case does not occur. Case 2: x is of the form $s \cdot t$, for certain terms s and t . Then, by $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} y$, we either have $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{a} \surd$ and $y \equiv t$, or we have $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{a} s'$ and $y \equiv s' \cdot t$ for some term s' . In the first case, we have $n(x) = n(s \cdot t) = n(s) + n(t) + 1 > n(t) = n(y)$, and in the second we can apply the induction hypothesis to arrive at $n(s) > n(s')$, so we get $n(x) = n(s \cdot t) = n(s) + n(t) + 1 > n(s') + n(t) + 1 = n(s' \cdot t) = n(y)$. Case 3: x is of the form $s + t$, for certain terms s and t . As $T(\text{BPA}_{\text{drt}}^- - \delta) \models s + t \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{a} y$ or $T(\text{BPA}_{\text{drt}}^- - \delta) \models t \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(s) > n(y)$ or $n(t) > n(y)$. As n ranges over the positive naturals only, in both cases $n(x) = n(s + t) = n(s) + n(t) + 1 > n(y)$. Case 4: $x \equiv \sigma(s)$ for a certain term s , does not occur, as $T(\text{BPA}_{\text{drt}}^- - \delta) \models \sigma(s) \not\xrightarrow{a}$.

- (v). Case 1: x is an undelayable action. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: x is of the form $s \cdot t$, for certain terms s and t . Then, necessarily, $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{\sigma} s'$ and $y \equiv s' \cdot t$ for some term s' . We now can apply the induction hypothesis to arrive at $n(s) > n(s')$, so we get $n(x) = n(s \cdot t) = n(s) + n(t) + 1 > n(s') + n(t) + 1 = n(s' \cdot t) = n(y)$. Case 3: x is of the form $s + t$, for certain terms s and t . Now, by $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{\sigma} y$ we know that either $T(\text{BPA}_{\text{drt}}^- - \delta) \models s \xrightarrow{\sigma} y$, or $T(\text{BPA}_{\text{drt}}^- - \delta) \models t \xrightarrow{\sigma} y$, or both. So, by the induction hypothesis, either $n(s) > n(y)$, or $n(t) > n(y)$, or both. So in all cases $n(x) = n(s + t) = n(s) + n(t) + 1 > n(y)$. Case 4: if x is of the form $\sigma(s)$, for a certain term s , it must be the case that $s \equiv y$. So, $n(x) = n(\sigma(s)) = n(\sigma(y)) = n(y) + 1 > n(y)$.

■

Theorem 4.3.1.8 (Completeness of $\text{BPA}_{\text{drt}}^- - \delta$)

The axiom system $\text{BPA}_{\text{drt}}^- - \delta$ is a complete axiomatization of the set of closed $\text{BPA}_{\text{drt}}^- - \delta$ terms modulo bisimulation equivalence.

Proof We use the direct method described in Proof Outline 4.2.3.1 on page 70. Let x and y be bisimilar closed $\text{BPA}_{\text{drt}}^- - \delta$ terms. We have to prove that $\text{BPA}_{\text{drt}}^- - \delta \vdash x = y$. With the aid of Theorem 4.3.1.2 and Theorem 4.3.1.4, it is enough to prove this for basic terms. By Lemma 4.3.1.6 it is even enough to prove for all basic terms x and y that:

$$x + y \sim_{\text{BPA}_{\text{drt}}^- - \delta} y \implies \text{BPA}_{\text{drt}}^- - \delta \vdash x + y = y$$

We will prove this by induction on $n(x)$, using Lemma 4.3.1.7(iv)-(v) and case distinction on the form of basic term x . Case 1: x is of the form \underline{a} , for $a \in A$. Then $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} \checkmark$, so $T(\text{BPA}_{\text{drt}}^- - \delta) \models x + y \xrightarrow{a} \checkmark$, and because $x + y \sim_{\text{BPA}_{\text{drt}}^- - \delta} y$ we have $T(\text{BPA}_{\text{drt}}^- - \delta) \models y \xrightarrow{a} \checkmark$, so with Lemma 4.3.1.7(i) we find that $\text{BPA}_{\text{drt}}^- - \delta \vdash x + y = y$. This proves the basis of our induction. Case 2: x is of the form $\underline{a} \cdot s$, when s is again a basic $\text{BPA}_{\text{drt}}^- - \delta$ term. Then $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{a} s$, and therefore $T(\text{BPA}_{\text{drt}}^- - \delta) \models x + y \xrightarrow{a} s$, so because $x + y \sim_{\text{BPA}_{\text{drt}}^- - \delta} y$ there is an s' with $T(\text{BPA}_{\text{drt}}^- - \delta) \models y \xrightarrow{a} s'$ and $s \sim_{\text{BPA}_{\text{drt}}^- - \delta} s'$. But then by Theorem 4.3.1.4 and Axiom A3 also $s + s' \sim_{\text{BPA}_{\text{drt}}^- - \delta} s'$ and $s' + s \sim_{\text{BPA}_{\text{drt}}^- - \delta} s$ and with induction we find $\text{BPA}_{\text{drt}}^- - \delta \vdash s + s' = s'$ and $\text{BPA}_{\text{drt}}^- - \delta \vdash s' + s = s$. So $\text{BPA}_{\text{drt}}^- - \delta \vdash s = s'$. Now $\text{BPA}_{\text{drt}}^- - \delta \vdash x + y = a \cdot s + y = a \cdot s' + y = y$ with Lemma 4.3.1.7(ii). Case 3: x is of the form $s + t$, for certain basic $\text{BPA}_{\text{drt}}^- - \delta$ terms s and t . Since $x + y \sim_{\text{BPA}_{\text{drt}}^- - \delta} y$, we also have $s + y \sim_{\text{BPA}_{\text{drt}}^- - \delta} y$ and $t + y \sim_{\text{BPA}_{\text{drt}}^- - \delta} y$. Then by the induction hypothesis $\text{BPA}_{\text{drt}}^- - \delta \vdash s + y = y$ and $\text{BPA}_{\text{drt}}^- - \delta \vdash t + y = y$. So $\text{BPA}_{\text{drt}}^- - \delta \vdash x + y = s + t + y = s + y = y$. Case 4: x is of the form $\sigma(x')$, for a certain basic $\text{BPA}_{\text{drt}}^- - \delta$ term x' . Now $T(\text{BPA}_{\text{drt}}^- - \delta) \models x \xrightarrow{\sigma} x'$, and since $x + y \sim_{\text{BPA}_{\text{drt}}^- - \delta} y$, we also have $T(\text{BPA}_{\text{drt}}^- - \delta) \models y \xrightarrow{\sigma} y'$ and $T(\text{BPA}_{\text{drt}}^- - \delta) \models x + y \xrightarrow{\sigma} x' + y'$ for some y' such that $x' + y' \sim_{\text{BPA}_{\text{drt}}^- - \delta} y'$. By Lemma 4.3.1.7(iii) we have $\text{BPA}_{\text{drt}}^- - \delta \vdash y = \sigma(y') + y$. By the induction hypothesis we have $\text{BPA}_{\text{drt}}^- - \delta \vdash x' + y' = y'$. So, $\text{BPA}_{\text{drt}}^- - \delta \vdash x + y = \sigma(x') + y = \sigma(x') + \sigma(y') + y = \sigma(x' + y') + y = \sigma(y') + y = y$.

■

Remark 4.3.1.9 (Completeness of $\text{BPA}_{\text{drt}}^- - \delta$)

Completeness of $\text{BPA}_{\text{drt}}^- - \delta$ is also claimed in Theorem 2.12.5 of BAETEN AND VERHOEF [37] (where $\text{BPA}_{\text{drt}}^- - \delta$ is called BPA_{dt}). The proof given there, however, is incorrect: the (supposedly) bijective mapping φ is not bijective, as $\varphi^{-1}(\sigma)$ is undefined.

4.3.2 BPA_{drt}^- -ID

We prove elimination, soundness, and completeness of BPA_{drt}^- -ID. The respective proofs are like the corresponding proofs for BPA_{drt}^- - δ in Section 4.3.1, albeit that the extension of the signature with the undelayable deadlock and the “now” operator makes the completeness proof more complex.

Theorem 4.3.2.1 (Elimination for BPA_{drt}^- -ID)

Let t be a closed BPA_{drt}^- -ID term. Then there is a basic term s such that BPA_{drt}^- -ID $\vdash s = t$.

Proof We use the lexicographical path ordering method we described in Proof Outline 4.2.1.1 on page 68. We select Axioms A4, A5, DRT2, and DCS1-DCS4 of BPA_{drt}^- -ID to be turned into the term-rewriting system for BPA_{drt}^- -ID shown in Table 4.2.

The rewriting rules of the term-rewriting system for BPA_{drt}^- -ID are given in Table 4.2. The well-founded ordering $>$ on constants and function symbols is the following:

$(x + y) \cdot z \rightarrow x \cdot z + y \cdot z$	RA4
$(x \cdot y) \cdot z \rightarrow x \cdot (y \cdot z)$	RA5
$\sigma(x) \cdot y \rightarrow \sigma(x \cdot y)$	RDRT2
$\nu(\underline{a}) \rightarrow \underline{a}$	RDCS1
$\nu(x + y) \rightarrow \nu(x) + \nu(y)$	RDCS2
$\nu(x \cdot y) \rightarrow \nu(x) \cdot \nu(y)$	RDCS3
$\nu(\sigma(x)) \rightarrow \underline{\delta}$	RDCS4

Table 4.2: Term-Rewriting System for BPA_{drt}^- -ID.

$$\underline{a} < \sigma < \cdot < \nu$$

Moreover, \cdot has the lexicographical status of the first argument. Now we show that the left-hand side of every rewriting rule is bigger than the right-hand side with respect to the ordering \succ_{lpo} . This is done by the following reductions:

$$\begin{aligned}
& \nu(\underline{a}) \succ_{lpo} \nu^*(\underline{a}) \\
& \quad \succ_{lpo} \underline{a} \\
\nu(x + y) & \succ_{lpo} \nu^*(x + y) \succ_{lpo} \nu^*(x + y) + \nu^*(x + y) \\
& \quad \succ_{lpo} \nu(x + \cdot y) + \nu(x + \cdot y) \succ_{lpo} \nu(x) + \nu(y) \\
\nu(x \cdot y) & \succ_{lpo} \nu^*(x \cdot y) \succ_{lpo} \nu^*(x \cdot y) \cdot \nu^*(x \cdot y) \succ_{lpo} \nu(x \cdot \cdot y) \cdot (x \cdot y) \\
& \quad \succ_{lpo} \nu(x) \cdot (x \cdot y) \succ_{lpo} \nu(x) \cdot \cdot (x \cdot y) \succ_{lpo} \nu(x) \cdot (x \cdot \cdot y) \succ_{lpo} \nu(x) \cdot y \\
\nu(\sigma(x)) & \succ_{lpo} \nu^*(\sigma(x)) \\
& \quad \succ_{lpo} \underline{\delta}
\end{aligned}$$

Note that we do not give reductions for RA4, RA5, and RDRT2 as these already have been given in the proof of Theorem 4.3.1.2 on page 72, and since the new ordering is a proper extension of the old one, these proofs remain valid.

Next, we will prove that the normal forms of the closed BPA_{drt}^- -ID terms are basic terms. Thereto, suppose that s is a normal form of some closed BPA_{drt}^- -ID term. Furthermore, suppose that s is not a basic term. Let s' denote the smallest subterm of s which is not a basic term. Then we can prove that s' is not a normal form by case analysis. We distinguish all possible cases:

- (i). s' is an undelayable action or $\underline{\delta}$. But then s' is a basic term. This is in contradiction with the assumption that s' is not a basic term, so this case does not occur.
- (ii). s' is of the form $s'_1 \cdot s'_2$ for basic terms s'_1 and s'_2 . With case analysis on the structure of basic term s'_1 :
 - (a) If s'_1 is an undelayable action or $\underline{\delta}$, then $s'_1 \cdot s'_2$ is a basic term, and so s' is a basic term which again contradicts the assumption that s' is not a basic term. This case can therefore not occur.
 - (b) If s'_1 is of the form $\underline{a} \cdot t$ for some $a \in A_\delta$ and basic term t , then rewriting rule RA5 can be applied. So, s' is not a normal form.
 - (c) If s'_1 is of the form $t_1 + t_2$ for t_1 and t_2 basic terms. Then rewriting rule RA4 is applicable. Therefore, s' is not a normal form.
 - (d) If s'_1 is of the form $\sigma(t)$ for some basic term t . Then rewriting rule RDRT2 is applicable. So, s' is not a normal form.
- (iii). s' is of the form $s'_1 + s'_2$ for basic terms s'_1 and s'_2 . In this case s' would be a basic term, which contradicts the assumption that s' is not a basic term. Therefore, this case cannot occur.
- (iv). s' is of the form $\sigma(t)$ for some basic term t . But then s' is basic term too, so the case does not occur.
- (v). s' is of the form $\nu(t)$ for some basic term t . But then one of RDCS1–RDCS4 is applicable, so s' is not a normal form.

In any case that can occur it follows that s' is not a normal form. Since s' is a subterm of s , we conclude that s is not a normal form. This contradicts the assumption that s is a normal form. From this contradiction we conclude that s is a basic term, which completes the proof. ■

Remark 4.3.2.2 (Elimination for BPA_{drt}^- -ID)

Elimination for BPA_{drt}^- -ID is also claimed (without proof) in Theorem 2.1 of BAETEN AND RENIERS [35].

Theorem 4.3.2.3 (Soundness of BPA_{drt}^- -ID)

The set of closed BPA_{drt}^- -ID terms modulo bisimulation equivalence is a model of BPA_{drt}^- -ID.

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. In Theorem 4.3.1.4 we have already proven the soundness of Axioms A1–A5 and DRT1–DRT2 with respect to the term-deduction system $T(BPA_{\text{drt}}^- - \delta)$. Since the term-deduction system $T(BPA_{\text{drt}}^-$ -ID) uses the same underlying model as the term-deduction system $T(BPA_{\text{drt}}^- - \delta)$, these proofs remain valid in the setting of BPA_{drt}^- -ID. Therefore, we only have to prove soundness of the additional Axioms DRT3–DRT5 and DCS1–DCS4.

Axiom DRT3 Take the relation:

$$R = \{(\underline{\delta} \cdot s, \underline{\delta}) \mid s \in C(\text{BPA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that neither the left-hand side nor the right-hand side of the axiom can perform any transition: $\underline{\delta} \cdot s \xrightarrow{a}$, $\underline{\delta} \cdot s \xrightarrow{\sigma}$ and $\underline{\delta} \xrightarrow{a}$, $\underline{\delta} \xrightarrow{\sigma}$.

Axiom DRT4 Take the relation:

$$R = \{(s, s), (s + \underline{\delta}, s) \mid s \in C(\text{BPA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. First, $s + \underline{\delta} \xrightarrow{a} p$ iff $s \xrightarrow{a} p$, and note that $(p, p) \in R$. Secondly, $s + \underline{\delta} \xrightarrow{a} \surd$ iff $s \xrightarrow{a} \surd$. Thirdly, $s + \underline{\delta} \xrightarrow{\sigma} p$ iff $s \xrightarrow{\sigma} p$, and note that $(p, p) \in R$.

Axiom DRT5 Take the relation:

$$R = \{(s, s), (\sigma(s) + \underline{\delta}, \sigma(s)) \mid s \in C(\text{BPA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that neither the left-hand side nor the right-hand side of the axiom can perform an a -transition: $\sigma(s) + \underline{\delta} \xrightarrow{a}$ and $\sigma(s) \xrightarrow{a}$. Furthermore, the only σ -transitions are $\sigma(s) + \underline{\delta} \xrightarrow{\sigma} s$ and $\sigma(s) \xrightarrow{\sigma} s$, and note that $(s, s) \in R$.

Axiom DCS1 Take the relation:

$$R = \{(\nu(\underline{a}), \underline{a})\}$$

We look at the transitions of both sides at the same time. Observe that either side of the axiom can only do an a -transition to \surd : $\nu(\underline{a}) \xrightarrow{a} \surd$ and $\underline{a} \xrightarrow{a} \surd$. No other transitions are possible.

Axiom DCS2 Take the relation:

$$R = \{(s, s), (\nu(s + t), \nu(s) + \nu(t)) \mid s, t \in C(\text{BPA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that neither side of the axiom can do a σ -transition: $\nu(s + t) \xrightarrow{\sigma}$ and $\nu(s) + \nu(t) \xrightarrow{\sigma}$. Furthermore, $\nu(s + t) \xrightarrow{a} p$ iff $s + t \xrightarrow{a} p$ iff $s \xrightarrow{a} p$ or $t \xrightarrow{a} p$ iff $\nu(s) \xrightarrow{a} p$ or $\nu(t) \xrightarrow{a} p$ iff $\nu(s) + \nu(t) \xrightarrow{a} p$, and note that $(p, p) \in R$. Finally, $\nu(s + t) \xrightarrow{a} \surd$ iff $s + t \xrightarrow{a} \surd$ iff $s \xrightarrow{a} \surd$ or $t \xrightarrow{a} \surd$ iff $\nu(s) \xrightarrow{a} \surd$ or $\nu(t) \xrightarrow{a} \surd$ iff $\nu(s) + \nu(t) \xrightarrow{a} \surd$.

Axiom DCS3 Take the relation:

$$R = \{(s, s), (\nu(s \cdot t), \nu(s) \cdot t) \mid s, t \in C(\text{BPA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that neither side of the axiom can do a σ -transition: $\nu(s \cdot t) \xrightarrow{\sigma}$ and $\nu(s) \cdot t \xrightarrow{\sigma}$. Furthermore, $\nu(s \cdot t) \xrightarrow{a} p$ iff $s \cdot t \xrightarrow{a} p$ iff $s \xrightarrow{a} \surd$ and $p \equiv t$ or $s \xrightarrow{a} s'$ and $p \equiv s' \cdot t$ iff $\nu(s) \xrightarrow{a} \surd$ and $p \equiv t$ or $\nu(s) \xrightarrow{a} s'$ and $p \equiv s' \cdot t$ iff $\nu(s) \cdot t \xrightarrow{a} p$, and note that $(p, p) \in R$. Finally, $\nu(s \cdot t) \xrightarrow{a} \surd$ and $\nu(s) \cdot t \xrightarrow{a} \surd$.

Axiom DCS4 Take the relation:

$$R = \{(\nu(\sigma(s)), \underline{\underline{\delta}}) \mid s \in C(\text{BPA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that neither the left-hand side nor the right-hand side of the axiom can perform any a -transition or σ -transition: $\nu(\sigma(s)) \xrightarrow{a}$, $\nu(\sigma(s)) \xrightarrow{\sigma}$, and $\underline{\underline{\delta}} \xrightarrow{a}$, $\underline{\underline{\delta}} \xrightarrow{\sigma}$.

■

Remark 4.3.2.4 (Soundness of $\text{BPA}_{\text{drt}}^- \text{-ID}$)

Soundness of $\text{BPA}_{\text{drt}}^- \text{-ID}$ is also claimed (without proof) in Section 2.12.1 of BAETEN AND VERHOEF [37] (where $\text{BPA}_{\text{drt}}^- \text{-ID}$ is called $\text{BPA}_{\delta \text{dt}}$), and in Theorem 2.2 of BAETEN AND REINIERS [35].

Lemma 4.3.2.5 (Towards Completeness of $\text{BPA}_{\text{drt}}^- \text{-ID}$)

Let x be a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term and let $a \in A$. Then we have:

- (i). $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} \surd \implies \text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \underline{\underline{a}} + x$,
- (ii). $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} y \implies \text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \underline{\underline{a}} \cdot y + x$,
- (iii). $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{\sigma} \implies \text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \nu(x)$,
- (iv). $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + \underline{\underline{\delta}} = x$,
- (v). $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{\sigma} y \implies \text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \sigma(y) + \nu(x)$,
- (vi). $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} y \implies n(x) > n(y)$,
- (vii). $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{\sigma} y \implies n(x) > n(y)$.

Proof For part (i)–(v) we assume, by Theorem 4.3.2.1 and Theorem 4.3.2.3, without loss of generality, that x is a basic term, and then apply induction on the structure of basic terms. For part (vi) and (vii) we again have to use induction on the general structure of terms.

- (i). Suppose that $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} \surd$. Case 1: $x \equiv \underline{\underline{b}}$, where $b \in A_\delta$. Because $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} \surd$, it must be the case that $b \equiv a$. So we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \underline{\underline{b}} = \underline{\underline{b}} + \underline{\underline{b}} = \underline{\underline{a}} + \underline{\underline{b}} = \underline{\underline{a}} + x$. Case 2: $x \equiv \underline{\underline{b}} \cdot x'$, where $b \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} \surd$, so this case does not occur. Case 3: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} \surd$, necessarily $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x' \xrightarrow{a} \surd$ or $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x'' \xrightarrow{a} \surd$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x' = \underline{\underline{a}} + x'$ or $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x'' = \underline{\underline{a}} + x''$. But then in both cases $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = x' + x'' = \underline{\underline{a}} + x' + x'' = \underline{\underline{a}} + x$. Case 4: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \models x \xrightarrow{a} \surd$, so this case does not occur.

- (ii). Suppose that $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$. Case 1: $x \equiv \underline{b}$, where $b \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, so this case does not occur. Case 2: $x \equiv \underline{b} \cdot x'$, where $b \in A_\delta$ and x' is a basic term. Then, because $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, it must be that $b \equiv a$ and $x' \equiv y$. So, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = x + x = \underline{b} \cdot x' + x = \underline{a} \cdot y + x$. Case 3: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{a} y$ or $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x'' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x' = \underline{a} \cdot y + x'$ or $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x'' = \underline{a} \cdot y + x''$. But then in both cases $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = x' + x'' = \underline{a} \cdot y + x' + x'' = \underline{a} \cdot y + x$. Case 4: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, so this case does not occur.
- (iii). Suppose that $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$. Case 1: $x \equiv \underline{a}$, where $a \in A_\delta$. We have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \underline{a} = \nu(\underline{a}) = \nu(x)$. Case 2: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. We have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \underline{a} \cdot x' = \nu(\underline{a}) \cdot x' = \nu(\underline{a} \cdot x') = \nu(x)$. Case 3: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, necessarily $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{\sigma} y$ and $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x'' \xrightarrow{\sigma} y$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x' = \nu(x')$ and $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x'' = \nu(x'')$. But then also $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = x' + x'' = \nu(x') + \nu(x'') = \nu(x' + x'') = \nu(x)$. Case 4: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, so this case does not occur.
- (iv). Case 1: $x \equiv \underline{a}$, where $a \in A_\delta$. Then we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + \underline{\delta} = \underline{a} + \underline{\delta} = \underline{a} = x$. Case 2: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. Then $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + \underline{\delta} = \underline{a} \cdot x' + \underline{\delta} = \underline{a} \cdot \underline{x'} + \underline{\delta} \cdot x' = (\underline{a} + \underline{\delta}) \cdot x' = \underline{a} \cdot x' = x$. Case 3: $x \equiv x' + x''$, where x' and x'' are basic terms. Then, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x' + \underline{\delta} = x'$, $x'' + \underline{\delta} = x''$. So, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + \underline{\delta} = x' + x'' + \underline{\delta} = x' + x'' = x$. Case 4: $x \equiv \sigma(x')$, where x' is a basic term. Then $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + \underline{\delta} = \sigma(x') + \underline{\delta} = \sigma(x') = x$.
- (v). Suppose that $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$. Case 1: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, so this case does not occur. Case 3: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, necessarily (1) $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma} y$, or, (2) $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma} y'$, or, (3) $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{\sigma} y', x'' \xrightarrow{\sigma} y''$ where $y \equiv y' + y''$. In the first case, by the induction hypothesis, we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x' = \sigma(y) + \nu(x')$, and, by (iii), $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x'' = \nu(x'')$. Therefore, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = x' + x'' = \sigma(y) + \nu(x') + \nu(x'') = \sigma(y) + \nu(x' + x'') = \sigma(y) + \nu(x)$. The second case is treated analogously. In the third case we have, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x' = \sigma(y') + \nu(x'), x'' = \sigma(y'') + \nu(x'')$. Therefore we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = x' + x'' = \sigma(y') + \nu(x') + \sigma(y'') + \nu(x'') = \sigma(y' + y'') + \nu(x' + x'') = \sigma(y) + \nu(x)$. Case 4: $x \equiv \sigma(x')$, where x' is a basic term. Because $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, it must be the case that $x' \equiv y$. So we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \sigma(x') = \sigma(y) = \sigma(y) + \underline{\delta} = \sigma(y) + \nu(\sigma(x')) = \sigma(y) + \nu(x)$.
- (vi). Suppose that $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$. Case 1: $x \equiv \underline{b}$, where $b \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, so this case does not occur. Case 2: $x \equiv x' \cdot x''$, for certain terms x' and x'' . Then, because $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, we either have $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{a} y$ and $y \equiv x''$, or we have $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{a} x'''$

and $y \equiv x''' \cdot x''$ for some term x''' . In the first case, we have $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x'') = n(y)$, and in the second we can apply the induction hypothesis to arrive at $n(x') > n(x''')$, so we get $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x''') + n(x'') + 1 = n(x''' \cdot x'') = n(y)$. Case 3: $x \equiv x' + x''$, for certain terms x' and x'' . Since $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{a} y$ or $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x'' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(x') > n(y)$ or $n(x'') > n(y)$. In both cases $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y)$. Case 4: $x \equiv \sigma(x')$, for a certain term x' . This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, so this case does not occur. Case 5: $x \equiv \nu(x')$, for a certain term x' . Since $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(x') > n(y)$. So, $n(x) = n(\nu(x')) = n(x') + 1 > n(y)$.

- (vii). Suppose that $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$. Case 1: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: $x \equiv x' \cdot x''$, for certain terms x' and x'' . Then necessarily, $x' \xrightarrow{\sigma} x'''$ and $y \equiv x''' \cdot x''$ for some term x''' . We now can apply the induction hypothesis to arrive at $n(x') > n(x''')$, so we get $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x''') + n(x'') + 1 = n(x''' \cdot x'') = n(y)$. Case 3: $x \equiv x' + x''$, for certain terms x' and x'' . As $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, necessarily (1) $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma} y$, or, (2) $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma} y$, or, (3) $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x' \xrightarrow{\sigma} y', x'' \xrightarrow{\sigma} y''$ where $y \equiv y' + y''$. In the first case, by the induction hypothesis, $n(x') > n(y)$. So $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y)$. The second case is treated analogously. In the third case, by the induction hypothesis, $n(x') > n(y')$ and $n(x'') > n(y'')$. So $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y') + n(y'') + 1 = n(y)$. Case 4: $x \equiv \sigma(x')$, for a certain term x' . Because $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, it must be the case that $x' \equiv y$. Then we have $n(x) = n(\sigma(x')) = n(x') + 1 = n(y) + 1 > n(y)$. Case 5: $x \equiv \nu(x')$, for a certain term x' . This is in contradiction with $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{\sigma} y$, so this case does not occur. ■

Theorem 4.3.2.6 (Completeness of $\text{BPA}_{\text{drt}}^- \text{-ID}$)

The axiom system $\text{BPA}_{\text{drt}}^- \text{-ID}$ is a complete axiomatization of the set of closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ terms modulo bisimulation equivalence.

Proof We use the direct method described in Proof Outline 4.2.3.1 on page 70. Suppose $x + y \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} y$. We then prove that $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + y = y$. By Theorem 4.3.2.1 we can restrict ourselves to basic terms without loss of generality. The proof is done with induction on $n(x)$, using Lemma 4.3.2.5(vi)–(vii) and case distinction on the form of basic term x .

- (i). $x \equiv \underline{\delta}$. Then, using Lemma 4.3.2.5(iv) we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + y = \underline{\delta} + y = y + \underline{\delta} = y$.
- (ii). $x \equiv \underline{a}$, where $a \in A$. From the deduction rules we have $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} \surd$ and $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x + y \xrightarrow{a} \surd$. Since $x + y \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} y$ we also have $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash y \xrightarrow{a} \surd$. By Lemma 4.3.2.5(i) we obtain $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash y = \underline{a} + y$. So, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + y = \underline{a} + y = y$.
- (iii). $x \equiv \underline{\delta} \cdot s$, where s is a basic term. Then we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x = \underline{\delta} \cdot s = \underline{\delta}$ and, using (i), $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + y = y$.

- (iv). $x \equiv \underline{a} \cdot s$, where $a \in A$ and s is a basic term. From the deduction rules we obtain $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x \xrightarrow{a} s$ and $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash x + y \xrightarrow{a} s$. Since $x + y \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} y$, we then also have $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash y \xrightarrow{a} t$ for some t such that $s \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} t$. By the induction hypothesis we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s = t$. From Lemma 4.3.2.5(ii) we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash y = \underline{a} \cdot t + y$. So, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + y = \underline{a} \cdot s + y = \underline{a} \cdot t + y = y$.
- (v). $x \equiv s + t$, where s and t are basic terms. Since $s + t + y \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} y$, we also have $s + y \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} y$ and $t + y \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} y$. By the induction hypothesis we then have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s + y = y, t + y = y$. So, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + y = s + t + y = s + y = y$.
- (vi). $x \equiv \sigma(s)$, where s is a basic term. From the deduction rules we now have that $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash \sigma(s) \xrightarrow{\sigma} s$ and since $x + y \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} y$ we also have $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \vDash y \xrightarrow{\sigma} t, x + y \xrightarrow{\sigma} s + t$ for some t such that $s + t \sim_{\text{BPA}_{\text{drt}}^- \text{-ID}} t$. By Lemma 4.3.2.5(v) we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash y = \sigma(t) + \nu(y)$. By the induction hypothesis we have $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s + t = t$. So, $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash x + y = \sigma(s) + y = \sigma(s) + \sigma(t) + \nu(y) = \sigma(s + t) + \nu(y) = \sigma(t) + \nu(y) = y$.

■

Remark 4.3.2.7 (Completeness of $\text{BPA}_{\text{drt}}^- \text{-ID}$)

Completeness of $\text{BPA}_{\text{drt}}^- \text{-ID}$ is also claimed (without proof) in Section 2.12.1 of BAETEN AND VERHOEF [37] (where $\text{BPA}_{\text{drt}}^- \text{-ID}$ is called $\text{BPA}_{\delta\text{drt}}$), and in Theorem 2.2 of BAETEN AND RENIERS [35].

4.3.3 $\text{BPA}_{\text{drt}}^-$

We prove elimination, soundness, and completeness of $\text{BPA}_{\text{drt}}^-$. The respective proofs are like the corresponding proofs for $\text{BPA}_{\text{drt}}^- \text{-ID}$ in Section 4.3.2, albeit that the extension of the signature with the immediate deadlock makes the completeness proof more complex.

Theorem 4.3.3.1 (Elimination for $\text{BPA}_{\text{drt}}^-$)

Let t be a closed $\text{BPA}_{\text{drt}}^-$ term. Then there is a basic term s such that $\text{BPA}_{\text{drt}}^- \vdash t = s$.

Proof We use the lexicographical path ordering method we described in Proof Outline 4.2.1.1 on page 68. The term-rewriting system of Table 4.3 on the next page is associated to $\text{BPA}_{\text{drt}}^-$ by assigning a direction to some of the axioms. Give \cdot the lexicographical status for the first argument and define the following well-founded partial ordering on constant and function symbols:

$$\underline{a} < \sigma < + < \cdot < \nu$$

We give the following reductions for the rewriting rules RA7ID and RDCSID:

$$\begin{aligned} \delta \cdot x &\succ_{\text{lpo}} \delta \cdot *x \\ &\succ_{\text{lpo}} \delta \\ \nu(\delta) &\succ_{\text{lpo}} \nu^*(\delta) \\ &\succ_{\text{lpo}} \delta \end{aligned}$$

$(x + y) \cdot z \rightarrow x \cdot z + y \cdot z$	RA4
$(x \cdot y) \cdot z \rightarrow x \cdot (y \cdot z)$	RA5
$\sigma(x) \cdot y \rightarrow \sigma(x \cdot y)$	RDRT2
$\dot{\delta} \cdot x \rightarrow \dot{\delta}$	RA7ID
$\nu(\underline{a}) \rightarrow \underline{a}$	RDCS1
$\nu(x + y) \rightarrow \nu(x) + \nu(y)$	RDCS2
$\nu(x \cdot y) \rightarrow \nu(x) \cdot y$	RDCS3
$\nu(\sigma(x)) \rightarrow \underline{\underline{\delta}}$	RDCS4
$\nu(\dot{\delta}) \rightarrow \dot{\delta}$	RDCSID

Table 4.3: Term-Rewriting System for BPA_{drt}^- .

Note that the reductions for the other rewriting rules have already been given in the proofs of previous elimination theorems.

Next, we will prove that the normal forms of the closed BPA_{drt}^- terms are basic terms. Thereto, suppose that s is a normal form of some closed BPA_{drt}^- term. Furthermore, suppose that s is not a basic term. Let s' denote the smallest subterm of s which is not a basic term. Then we can prove that s' is not a normal form by case analysis. We distinguish all possible cases:

- (i). s' is an undelayable action, $\underline{\underline{\delta}}$, or $\dot{\delta}$. But then s' is a basic term. This is in contradiction with the assumption that s' is not a basic term, so this case does not occur.
- (ii). s' is of the form $s'_1 \cdot s'_2$ for basic terms s'_1 and s'_2 . With case analysis on the structure of basic term s'_1 :
 - (a) If s'_1 is $\dot{\delta}$ then rewriting rule RA7ID can be applied, and hence s' is not a normal form.
 - (b) If s'_1 is of the form \underline{a} for some $a \in A_\delta$, then $s'_1 \cdot s'_2$ is a basic term, and so s' is a basic term which again contradicts the assumption that s' is not a basic term. This case can therefore not occur.
 - (c) If s'_1 is of the form $\underline{a} \cdot t$ for some $a \in A_\delta$ and some basic term t , then rewriting rule RA5 can be applied. So, s' is not a normal form.
 - (d) If s'_1 is of the form $t_1 + t_2$ for t_1 and t_2 basic terms. Then rewriting rule RA4 is applicable. Therefore, s' is not a normal form.
 - (e) If s'_1 is of the form $\sigma(t)$ for some basic term t . Then rewriting rule RDRT2 is applicable. So, s' is not a normal form.
- (iii). s' is of the form $s'_1 + s'_2$ for basic terms s'_1 and s'_2 . In this case s' would be a basic term, which contradicts the assumption that s' is not a basic term. Therefore, this case cannot occur.

- (iv). s' is of the form $\sigma(t)$ for some basic term t . But then s' is basic term too, so the case does not occur.
- (v). s' is of the form $\nu(t)$ for some basic term t . But then one of RDCS1–RDCS4 or RDCSID is applicable, so s' is not a normal form.

In any case that can occur it follows that s' is not a normal form. Since s' is a subterm of s , we conclude that s is not a normal form. This contradicts the assumption that s is a normal form. From this contradiction we conclude that s is a basic term, which completes the proof. ■

Remark 4.3.3.2 (Elimination for $\text{BPA}_{\text{drt}}^-$)

Elimination for a slightly different version of $\text{BPA}_{\text{drt}}^-$ is also claimed (without proof) in Section 3.4 of BAETEN AND BERGSTRA [24].

Theorem 4.3.3.3 (Soundness of $\text{BPA}_{\text{drt}}^-$)

The set of closed $\text{BPA}_{\text{drt}}^-$ terms modulo bisimulation equivalence is a model of $\text{BPA}_{\text{drt}}^-$.

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. For soundness of Axioms A1–A5, DRT1–DRT4, and DCS1–DCS4 we refer to the proof of soundness of $\text{BPA}_{\text{drt}}^-$ -ID. To extend these proofs from $\text{BPA}_{\text{drt}}^-$ -ID to $\text{BPA}_{\text{drt}}^-$, we have to check that the bisimulations given in previous soundness proofs respect the immediate-deadlock predicate (as required by transfer condition (iv.) in 3.2.3.5 on page 53). However, as the fact that they do can be easily checked, we will not give details.

It remains to prove soundness of the axioms from Table 3.7 on page 52. For all axioms, we look at the transitions of both sides at the same time.

Axiom DRTSID Take the relation:

$$R = \{(\sigma(\dot{\delta}), \underline{\delta})\}$$

We look at the transitions of both sides at the same time. We have $\sigma(\dot{\delta}) \rightarrow$ and $\underline{\delta} \rightarrow$. Also, $\neg\text{ID}(\sigma(\dot{\delta}))$ and $\neg\text{ID}(\underline{\delta})$.

Axiom A6ID Take the relation:

$$R = \{(s, s), (s + \dot{\delta}, s) \mid s \in C(\text{BPA}_{\text{drt}}^-)\}$$

We look at the transitions of both sides at the same time. We have $s + \dot{\delta} \xrightarrow{a} p$ iff $s \xrightarrow{a} p$ and $s + \dot{\delta} \xrightarrow{\sigma} p$ iff $s \xrightarrow{\sigma} p$, and note that $(p, p) \in R$. Also, $s + \dot{\delta} \xrightarrow{a} \surd$ iff $s \xrightarrow{a} \surd$, and $\text{ID}(s + \dot{\delta})$ iff $\text{ID}(s) \wedge \text{ID}(\dot{\delta})$ iff $\text{ID}(s)$.

Axiom A7ID Take the relation:

$$R = \{(\dot{\delta} \cdot s, \dot{\delta}) \mid s \in C(\text{BPA}_{\text{drt}}^-)\}$$

We look at the transitions of both sides at the same time. We have $\dot{\delta} \cdot s \rightarrow$ and $\dot{\delta} \rightarrow$. Also, $\text{ID}(\dot{\delta} \cdot s)$ and $\text{ID}(\dot{\delta})$.

Axiom DCSID Take the relation:

$$R = \{(\nu(\dot{\delta}), \dot{\delta})\}$$

We look at the transitions of both sides at the same time. We have $\nu(\dot{\delta}) \rightarrow$ and $\dot{\delta} \rightarrow$. Also, $ID(\nu(\dot{\delta}))$ and $ID(\dot{\delta})$.

■

Remark 4.3.3.4 (Soundness of BPA_{drt}^-)

Soundness of a slightly different version of BPA_{drt}^- is also claimed (without proof) in Section 3.5 of BAETEN AND BERGSTRÅ [24].

Lemma 4.3.3.5 (Towards Completeness of BPA_{drt}^-)

Let x be a closed BPA_{drt}^- term and let $a \in A$. Then we have:

- (i). $T(BPA_{drt}^-) \models x \xrightarrow{a} \surd \implies BPA_{drt}^- \vdash x = \underline{a} + x$,
- (ii). $T(BPA_{drt}^-) \models x \xrightarrow{a} y \implies BPA_{drt}^- \vdash x = \underline{a} \cdot y + x$,
- (iii). $T(BPA_{drt}^-) \models ID(x) \implies BPA_{drt}^- \vdash x = \dot{\delta}$,
- (iv). $T(BPA_{drt}^-) \models \neg ID(x) \implies BPA_{drt}^- \vdash x + \underline{\dot{\delta}} = x$,
- (v). $T(BPA_{drt}^-) \models x \xrightarrow{\sigma} \implies BPA_{drt}^- \vdash x = \nu(x)$,
- (vi). $T(BPA_{drt}^-) \models x \xrightarrow{\sigma} y \implies BPA_{drt}^- \vdash x = \sigma(y) + \nu(x)$,
- (vii). $T(BPA_{drt}^-) \models x \xrightarrow{a} y \implies n(x) > n(y)$,
- (viii). $T(BPA_{drt}^-) \models x \xrightarrow{\sigma} y \implies n(x) > n(y)$.

Proof For part (i)–(vi) we assume, by Theorem 4.3.3.1 and Theorem 4.3.3.3, without loss of generality, that x is a basic term, and apply induction on the structure of basic terms. For part (vii) and (viii) we again have to use induction on the general structure of terms.

- (i). Suppose that $T(BPA_{drt}^-) \models x \xrightarrow{a} \surd$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(BPA_{drt}^-) \models x \xrightarrow{a} \surd$, so this case does not occur. Case 2: $x \equiv \underline{b}$, where $b \in A_\delta$. Because $T(BPA_{drt}^-) \models x \xrightarrow{a} \surd$, it must be the case that $b \equiv a$. So we have $BPA_{drt}^- \vdash x = \underline{b} = \underline{b} + \underline{b} = \underline{a} + \underline{b} = \underline{a} + x$. Case 3: $x \equiv \underline{b} \cdot x'$, where $b \in A_\delta$ and x' is a basic term. This is in contradiction with $T(BPA_{drt}^-) \models x \xrightarrow{a} \surd$, so this case does not occur. Case 4: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(BPA_{drt}^-) \models x \xrightarrow{a} \surd$, necessarily $T(BPA_{drt}^-) \models x' \xrightarrow{a} \surd$ or $T(BPA_{drt}^-) \models x'' \xrightarrow{a} \surd$. Therefore, by the induction hypothesis, $BPA_{drt}^- \vdash x' = \underline{a} + x'$ or $BPA_{drt}^- \vdash x'' = \underline{a} + x''$. But then in both cases $BPA_{drt}^- \vdash x = x' + x'' = \underline{a} + x' + x'' = \underline{a} + x$. Case 5: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(BPA_{drt}^-) \models x \xrightarrow{a} \surd$, so this case does not occur.
- (ii). Suppose that $T(BPA_{drt}^-) \models x \xrightarrow{a} y$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(BPA_{drt}^-) \models x \xrightarrow{a} y$, so this case does not occur. Case 2: $x \equiv \underline{b}$, where $b \in A_\delta$. This is in contradiction with $T(BPA_{drt}^-) \models x \xrightarrow{a} y$, so this case does not occur. Case 3: $x \equiv \underline{b} \cdot x'$, where $b \in A_\delta$ and x' is a basic term. Then, because $T(BPA_{drt}^-) \models x \xrightarrow{a} y$, it

must be that $b \equiv a$ and $x' \equiv y$. So, $\text{BPA}_{\text{drt}}^- \vdash x = x + x = \underline{b} \cdot x' + x = \underline{a} \cdot y + x$. Case 4: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}^-) \vDash x' \xrightarrow{a} y$ or $T(\text{BPA}_{\text{drt}}^-) \vDash x'' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \vdash x' = \underline{a} \cdot y + x'$ or $\text{BPA}_{\text{drt}}^- \vdash x'' = \underline{a} \cdot y + x''$. But then in both cases $\text{BPA}_{\text{drt}}^- \vdash x = x' + x'' = \underline{a} \cdot y + x' + x'' = \underline{a} \cdot y + x$. Case 5: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{a} y$, so this case does not occur.

(iii). Suppose that $T(\text{BPA}_{\text{drt}}^-) \vDash \text{ID}(x)$. Case 1: $x \equiv \dot{\delta}$. Then we have $\text{BPA}_{\text{drt}}^- \vdash x = \dot{\delta}$ is trivially fulfilled. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash \text{ID}(x)$, so this case does not occur. Case 3: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash \text{ID}(x)$, so this case does not occur. Case 4: $x \equiv x' + x''$, where x' and x'' are basic terms. Then, because $T(\text{BPA}_{\text{drt}}^-) \vDash \text{ID}(x)$, it must be the case that $T(\text{BPA}_{\text{drt}}^-) \vDash \text{ID}(x'), \text{ID}(x'')$. So, by the induction hypothesis, we have that $\text{BPA}_{\text{drt}}^- \vdash x' = \dot{\delta}, x'' = \dot{\delta}$. But then also $\text{BPA}_{\text{drt}}^- \vdash x = x' + x'' = \dot{\delta} + \dot{\delta} = \dot{\delta}$. Case 5: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash \text{ID}(x)$, so this case does not occur.

(iv). Suppose that $T(\text{BPA}_{\text{drt}}^-) \vDash \neg \text{ID}(x)$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash \neg \text{ID}(x)$, so this case does not occur. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. Then we have $\text{BPA}_{\text{drt}}^- \vdash x + \underline{\delta} = \underline{a} + \underline{\delta} = \underline{a} = x$. Case 3: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. Then $\text{BPA}_{\text{drt}}^- \vdash x + \underline{\delta} = \underline{a} \cdot x' + \underline{\delta} = \underline{a} \cdot x' + \underline{\delta} \cdot x' = (\underline{a} + \underline{\delta}) \cdot x' = \underline{a} \cdot x' = x$. Case 4: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^-) \vDash \neg \text{ID}(x)$, necessarily $T(\text{BPA}_{\text{drt}}^-) \vDash \neg \text{ID}(x')$ or $T(\text{BPA}_{\text{drt}}^-) \vDash \neg \text{ID}(x'')$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \vdash x' + \underline{\delta} = x'$ or $\text{BPA}_{\text{drt}}^- \vdash x'' + \underline{\delta} = x''$. So, in both cases, $\text{BPA}_{\text{drt}}^- \vdash x + \underline{\delta} = x' + x'' + \underline{\delta} = x' + x'' = x$. Case 5: $x \equiv \sigma(x')$, where x' is a basic term. Then $\text{BPA}_{\text{drt}}^- \vdash x + \underline{\delta} = \sigma(x') + \underline{\delta} = \sigma(x') = x$.

(v). Suppose that $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma}$. Case 1: $x \equiv \dot{\delta}$. By Axiom DCSID we have $\text{BPA}_{\text{drt}}^- \vdash x = \dot{\delta} = \nu(\dot{\delta}) = \nu(x)$. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. We have $\text{BPA}_{\text{drt}}^- \vdash x = \underline{a} = \nu(\underline{a}) = \nu(x)$. Case 3: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. We have $\text{BPA}_{\text{drt}}^- \vdash x = \underline{a} \cdot x' = \nu(\underline{a}) \cdot x' = \nu(\underline{a} \cdot x') = \nu(x)$. Case 4: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma}$, necessarily $T(\text{BPA}_{\text{drt}}^-) \vDash x' \xrightarrow{\sigma}$ and $T(\text{BPA}_{\text{drt}}^-) \vDash x'' \xrightarrow{\sigma}$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^- \vdash x' = \nu(x')$ and $\text{BPA}_{\text{drt}}^- \vdash x'' = \nu(x'')$. But then also $\text{BPA}_{\text{drt}}^- \vdash x = x' + x'' = \nu(x') + \nu(x'') = \nu(x' + x'') = \nu(x)$. Case 5: $x \equiv \sigma(x')$, where x' is a basic term with $\neg \text{ID}(x)$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma}$, so this case does not occur. Case 6: $x \equiv \sigma(x')$, where x' is a basic term. Then, by $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma}$, it must be the case that $\text{ID}(x')$. So, by (iii), we have that $\text{BPA}_{\text{drt}}^- \vdash x' = \dot{\delta}$. Therefore, $\text{BPA}_{\text{drt}}^- \vdash x = \sigma(x') = \sigma(\dot{\delta}) = \underline{\delta} = \nu(\sigma(x')) = \nu(x)$.

(vi). Suppose that $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma} y$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma} y$, so this case does not occur. Case 3: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma} y$, so this case does not occur. Case 4: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}^-) \vDash x \xrightarrow{\sigma} y$, necessarily (1) $T(\text{BPA}_{\text{drt}}^-) \vDash x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma}$, or, (2) $T(\text{BPA}_{\text{drt}}^-) \vDash x' \xrightarrow{\sigma}, x'' \xrightarrow{\sigma} y$, or, (3) $T(\text{BPA}_{\text{drt}}^-) \vDash x' \xrightarrow{\sigma} y', x'' \xrightarrow{\sigma} y'$ where $y \equiv y' + y''$. In the first case, by the induction hypothesis, we have $\text{BPA}_{\text{drt}}^- \vdash x' = \sigma(y) + \nu(x')$,

and, by (v), $BPA_{\text{drt}}^- \vdash x'' = \nu(x'')$. Therefore, $BPA_{\text{drt}}^- \vdash x = x' + x'' = \sigma(y) + \nu(x') + \nu(x'') = \sigma(y) + \nu(x' + x'') = \sigma(y) + \nu(x)$. The second case is treated analogously. In the third case we have, by the induction hypothesis, $BPA_{\text{drt}}^- \vdash x' = \sigma(y') + \nu(x')$, $x'' = \sigma(y'') + \nu(x'')$. Therefore we have $BPA_{\text{drt}}^- \vdash x = x' + x'' = \sigma(y') + \nu(x') + \sigma(y'') + \nu(x'') = \sigma(y' + y'') + \nu(x' + x'') = \sigma(y) + \nu(x)$. Case 5: $x \equiv \sigma(x')$, where x' is a basic term. Because $T(BPA_{\text{drt}}^-) \models x \xrightarrow{\sigma} y$, it must be the case that $x' \equiv y$. So we have $BPA_{\text{drt}}^- \vdash x = \sigma(x') = \sigma(y) = \sigma(y) + \underline{\delta} = \sigma(y) + \nu(\sigma(x')) = \sigma(y) + \nu(x)$.

(vii). Suppose that $T(BPA_{\text{drt}}^-) \models x \xrightarrow{a} y$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(BPA_{\text{drt}}^-) \models x \xrightarrow{a} y$, so this case does not occur. Case 2: $x \equiv \underline{b}$, where $b \in A_\delta$. This is in contradiction with $T(BPA_{\text{drt}}^-) \models x \xrightarrow{a} y$, so this case does not occur. Case 3: $x \equiv x' \cdot x''$, for certain terms x' and x'' . Then, because $T(BPA_{\text{drt}}^-) \models x \xrightarrow{a} y$, we either have $T(BPA_{\text{drt}}^-) \models x' \xrightarrow{a} \surd$ and $y \equiv x''$, or we have $T(BPA_{\text{drt}}^-) \models x' \xrightarrow{a} x'''$ and $y \equiv x''' \cdot x''$ for some term x''' . In the first case, we have $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x'') = n(y)$, and in the second we can apply the induction hypothesis to arrive at $n(x') > n(x''')$, so we get $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x''') + n(x'') + 1 = n(x''' \cdot x'') = n(y)$. Case 4: $x \equiv x' + x''$, for certain terms x' and x'' . Since $T(BPA_{\text{drt}}^-) \models x \xrightarrow{a} y$, necessarily $T(BPA_{\text{drt}}^-) \models x' \xrightarrow{a} y$ or $T(BPA_{\text{drt}}^-) \models x'' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(x') > n(y)$ or $n(x'') > n(y)$. In both cases $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y)$. Case 5: $x \equiv \sigma(x')$, for a certain term x' . This is in contradiction with $T(BPA_{\text{drt}}^-) \models x \xrightarrow{a} y$, so this case does not occur. Case 6: $x \equiv \nu(x')$, for a certain term x' . Since $T(BPA_{\text{drt}}^-) \models x \xrightarrow{a} y$, necessarily $T(BPA_{\text{drt}}^-) \models x' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(x') > n(y)$. So, $n(x) = n(\nu(x')) = n(x') + 1 > n(y)$.

(viii). Suppose that $T(BPA_{\text{drt}}^-) \models x \xrightarrow{\sigma} y$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(BPA_{\text{drt}}^-) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(BPA_{\text{drt}}^-) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 3: $x \equiv x' \cdot x''$, for certain terms x' and x'' . Then necessarily, $x' \xrightarrow{\sigma} x'''$ and $y \equiv x''' \cdot x''$ for some term x''' . We now can apply the induction hypothesis to arrive at $n(x') > n(x''')$, so we get $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x''') + n(x'') + 1 = n(x''' \cdot x'') = n(y)$. Case 4: $x \equiv x' + x''$, for certain terms x' and x'' . As $T(BPA_{\text{drt}}^-) \models x \xrightarrow{\sigma} y$, necessarily (1) $T(BPA_{\text{drt}}^-) \models x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma} \surd$, or, (2) $T(BPA_{\text{drt}}^-) \models x' \xrightarrow{\sigma} \surd, x'' \xrightarrow{\sigma} y$, or, (3) $T(BPA_{\text{drt}}^-) \models x' \xrightarrow{\sigma} y', x'' \xrightarrow{\sigma} y''$ where $y \equiv y' + y''$. In the first case, by the induction hypothesis, $n(x') > n(y)$. So $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y)$. The second case is treated analogously. In the third case, by the induction hypothesis, $n(x') > n(y')$ and $n(x'') > n(y'')$. So $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y') + n(y'') + 1 = n(y)$. Case 5: $x \equiv \sigma(x')$, for a certain term x' . Because $T(BPA_{\text{drt}}^-) \models x \xrightarrow{\sigma} y$, it must be the case that $x' \equiv y$. Then we have $n(x) = n(\sigma(x')) = n(x') + 1 = n(y) + 1 > n(y)$. Case 6: $x \equiv \nu(x')$, for a certain term x' . This is in contradiction with $T(BPA_{\text{drt}}^-) \models x \xrightarrow{\sigma} y$, so this case does not occur.

■

Theorem 4.3.3.6 (Completeness of BPA_{drt}^-)

The axiom system BPA_{drt}^- is a complete axiomatization of the set of closed BPA_{drt}^- terms modulo bisimulation equivalence.

Proof We use the direct method described in Proof Outline 4.2.3.1 on page 70. Suppose $s + t \sim_{\text{BPA}_{\text{drt}}^-} t$. We then prove that $\text{BPA}_{\text{drt}}^- \vdash s + t = t$. By Theorem 4.3.3.1 we can restrict ourselves to basic terms without loss of generality. The proof is done with induction on $n(s)$, using Lemma 4.3.3.5(vii)–(viii) and case distinction on the form of basic term s .

- (i). $s \equiv \delta$. Using Axiom A6ID we have $\text{BPA}_{\text{drt}}^- \vdash s + t = \delta + t = t + \delta = t$.
- (ii). $s \equiv \underline{\delta}$. Then we have $\neg\text{ID}(s + t)$, because $T(\text{BPA}_{\text{drt}}^-) \models \neg\text{ID}(s)$. Since $s + t \sim_{\text{BPA}_{\text{drt}}^-} t$, we also have $T(\text{BPA}_{\text{drt}}^-) \models \neg\text{ID}(t)$. Using Lemma 4.3.3.5(iv) we have $\text{BPA}_{\text{drt}}^- \vdash s + t = \underline{\delta} + t = t + \underline{\delta} = t$.
- (iii). $s \equiv \underline{a}$, where $a \in A$. From the deduction rules we have $T(\text{BPA}_{\text{drt}}^-) \models s \xrightarrow{a} \surd$ and $T(\text{BPA}_{\text{drt}}^-) \models s + t \xrightarrow{a} \surd$. Since $s + t \sim_{\text{BPA}_{\text{drt}}^-} t$ we also have $T(\text{BPA}_{\text{drt}}^-) \models t \xrightarrow{a} \surd$. By Lemma 4.3.3.5(i) we obtain $\text{BPA}_{\text{drt}}^- \vdash t = \underline{a} + t$. So, $\text{BPA}_{\text{drt}}^- \vdash s + t = \underline{a} + t = t$.
- (iv). $s \equiv \underline{\delta} \cdot s'$, where s' is a basic term. Then we have $\text{BPA}_{\text{drt}}^- \vdash s = \underline{\delta} \cdot s' = \underline{\delta}$ and, using (ii), $\text{BPA}_{\text{drt}}^- \vdash s + t = t$.
- (v). $s \equiv \underline{a} \cdot s'$, where $a \in A$ and s' is a basic term. From the deduction rules we obtain $T(\text{BPA}_{\text{drt}}^-) \models s \xrightarrow{a} s'$ and $T(\text{BPA}_{\text{drt}}^-) \models s + t \xrightarrow{a} s'$. Since $s + t \sim_{\text{BPA}_{\text{drt}}^-} t$, we then also have $T(\text{BPA}_{\text{drt}}^-) \models t \xrightarrow{a} t'$ for some t' such that $s' \sim_{\text{BPA}_{\text{drt}}^-} t'$. By the induction hypothesis we have $\text{BPA}_{\text{drt}}^- \vdash s' = t'$. From Lemma 4.3.3.5(ii) we have $\text{BPA}_{\text{drt}}^- \vdash t = \underline{a} \cdot t' + t$. So, $\text{BPA}_{\text{drt}}^- \vdash s + t = \underline{a} \cdot s' + t = \underline{a} \cdot t' + t = t$.
- (vi). $s \equiv s' + s''$, where s' and s'' are basic terms. Since $s' + s'' + t \sim_{\text{BPA}_{\text{drt}}^-} t$, we also have $s' + t \sim_{\text{BPA}_{\text{drt}}^-} t$ and $s'' + t \sim_{\text{BPA}_{\text{drt}}^-} t$. By the induction hypothesis we then have $\text{BPA}_{\text{drt}}^- \vdash s' + t = t, s'' + t = t$. So, $\text{BPA}_{\text{drt}}^- \vdash s + t = s' + s'' + t = s' + t = t$.
- (vii). $s \equiv \sigma(s')$, where s' is a basic term. From the deduction rules we have $T(\text{BPA}_{\text{drt}}^-) \models \sigma(s') \xrightarrow{\sigma} s'$ and since $s + t \sim_{\text{BPA}_{\text{drt}}^-} t$ we also have $T(\text{BPA}_{\text{drt}}^-) \models t \xrightarrow{\sigma} t', s + t \xrightarrow{\sigma} s' + t'$ for some t' such that $s' + t' \sim_{\text{BPA}_{\text{drt}}^-} t'$. By Lemma 4.3.3.5(vi) we have $\text{BPA}_{\text{drt}}^- \vdash t = \sigma(t') + \nu(t)$. By the induction hypothesis we have $\text{BPA}_{\text{drt}}^- \vdash s' + t' = t'$. So, $\text{BPA}_{\text{drt}}^- \vdash s + t = \sigma(s') + t = \sigma(s') + \sigma(t') + \nu(t) = \sigma(s' + t') + \nu(t) = \sigma(t') + \nu(t) = t$.

■

Remark 4.3.3.7 (Completeness of $\text{BPA}_{\text{drt}}^-$)

Completeness of a slightly different version of $\text{BPA}_{\text{drt}}^-$ is also claimed (without proof) in Section 3.5 of BAETEN AND BERGSTRA [24].

4.3.4 $\text{BPA}_{\text{drt}}^+$

We prove elimination, soundness, and completeness of $\text{BPA}_{\text{drt}}^+$. The respective proofs are like the corresponding proofs for $\text{BPA}_{\text{drt}}^-$ in Section 4.3.3, although the addition to the signature of delayable actions and the unbounded start delay complicates matters.

Theorem 4.3.4.1 (Elimination for $\text{BPA}_{\text{drt}}^+$)

Let t be a closed $\text{BPA}_{\text{drt}}^+$ term. Then there is a basic term s such that $\text{BPA}_{\text{drt}}^+ \vdash t = s$.

Proof We use the lexicographical path ordering method we described in Proof Outline 4.2.1.1 on page 68. The term-rewriting system is shown in Table 4.4. The rewriting rules RA4, RA5, RDRT2, RATS, RA7ID, RDCS1–RDCS4, and RDCSID are obtained directly from the axioms. The rewriting rules RUSD1–RUSD5 and RDCS5 are added to deal properly with the recursive definition of unbounded start delay. The corresponding equalities are for closed terms derivable from the axioms, as is shown in Proposition 3.2.4.14 on page 58. The operator \cdot is assigned the lexicographical status for the first argument and

$(x + y) \cdot z \rightarrow x \cdot z + y \cdot z$	RA4
$(x \cdot y) \cdot z \rightarrow x \cdot (y \cdot z)$	RA5
$\sigma(x) \cdot y \rightarrow \sigma(x \cdot y)$	RDRT2
$[\underline{a}]^\omega \rightarrow a$	RATS
$[a]^\omega \rightarrow a$	RUSD1
$[x \cdot y]^\omega \rightarrow [x]^\omega \cdot y$	RUSD2
$[x + y]^\omega \rightarrow [x]^\omega + [y]^\omega$	RUSD3
$[\sigma(x)]^\omega \rightarrow \delta$	RUSD4
$[\dot{\delta}]^\omega \rightarrow \delta$	RUSD5
$\dot{\delta} \cdot x \rightarrow \dot{\delta}$	RA7ID
$v(\underline{a}) \rightarrow \underline{a}$	RDCS1
$v(x + y) \rightarrow v(x) + v(y)$	RDCS2
$v(x \cdot y) \rightarrow v(x) \cdot y$	RDCS3
$v(\sigma(x)) \rightarrow \underline{\delta}$	RDCS4
$v(a) \rightarrow \underline{a}$	RDCS5
$v(\dot{\delta}) \rightarrow \delta$	RDCSID

Table 4.4: Term-Rewriting System for BPA_{drt} .

the following well-founded partial ordering on the signature of BPA_{drt} is defined:

$$\underline{a} < a < \sigma < + < \cdot < []^\omega < v$$

We give the following reductions for rewriting rules RATS, RUSD1–RUSD5, and RDCS5:

$$\begin{aligned} [\underline{a}]^\omega &\succ_{lpo} [\underline{a}]^{\omega^*} \\ &\succ_{lpo} a \\ [a]^\omega &\succ_{lpo} [a]^{\omega^*} \\ &\succ_{lpo} a \\ [x \cdot y]^\omega &\succ_{lpo} [x \cdot y]^{\omega^*} \succ_{lpo} [x \cdot y]^{\omega^*} \cdot [x \cdot y]^{\omega^*} \succ_{lpo} [x \cdot * y]^\omega \cdot (x \cdot y) \\ &\succ_{lpo} [x]^\omega \cdot (x \cdot y) \succ_{lpo} [x]^\omega \cdot (x \cdot * y) \succ_{lpo} [x]^\omega \cdot y \\ [x + y]^\omega &\succ_{lpo} [x + y]^{\omega^*} \succ_{lpo} [x + y]^{\omega^*} + [x + y]^{\omega^*} \succ_{lpo} [x + * y]^\omega + [x + * y]^\omega \\ &\succ_{lpo} [x]^\omega + [y]^\omega \end{aligned}$$

$$\begin{aligned}
& [\sigma(x)]^\omega \succ_{\text{lpo}} [\sigma(x)]^{\omega^*} \\
& \succ_{\text{lpo}} \delta \\
& [\dot{\delta}]^\omega \succ_{\text{lpo}} [\dot{\delta}]^{\omega^*} \\
& \succ_{\text{lpo}} \delta \\
& \nu(a) \succ_{\text{lpo}} \nu^*(a) \\
& \succ_{\text{lpo}} \underline{a}
\end{aligned}$$

Note that the reductions for the other rewriting rules have already been given in the proofs of previous elimination theorems.

It remains to prove that every normal form of a closed BPA_{drt} term is a basic term. Suppose that s is the normal form of a closed BPA_{drt} term. Furthermore, suppose that s is not a basic term and that s' is the smallest subterm of s which is not a basic term. We distinguish all possible cases:

- (i). s' is of the form \underline{a} or a for some $a \in A_\delta$, or of the form $\dot{\delta}$. Then s' is clearly a basic term, so this case does not occur.
- (ii). s' is of the form $s_1 \cdot s_2$ for basic terms s_1 and s_2 . With respect to basic term s_1 the following cases can be distinguished:
 - (a) $s_1 \equiv \dot{\delta}$; then RA7ID is applicable, so s' is not a normal form.
 - (b) $s_1 \equiv \underline{a}$ for some $a \in A_\delta$. Then s' is a basic term. This contradicts the assumption that s' is not a basic term.
 - (c) $s_1 \equiv a$ for some $a \in A_\delta$. Then s' is a basic term, and we have again a contradiction.
 - (d) $s_1 \equiv \underline{a} \cdot s'_1$ for some $a \in A_\delta$ and basic term s'_1 . Then rewriting rule RA5 is applicable, so s' is not a normal form.
 - (e) $s_1 \equiv a \cdot s'_1$ for some $a \in A_\delta$ and some basic term s'_1 . Then rewriting rule RA5 is applicable, so s' is not a normal form.
 - (f) $s_1 \equiv s'_1 + s''_1$ for some basic terms s'_1 and s''_1 . Then rewriting rule RA4 is applicable, so s' is not a normal form.
 - (g) $s_1 \equiv \sigma(s'_1)$ for some basic term s'_1 . Then rewriting rule RDRT2 is applicable, so s' is not a normal form.
- (iii). s' is of the form $s'_1 + s'_2$ for basic terms s'_1 and s'_2 . Then s' is a basic term itself, so this case cannot happen.
- (iv). s' is of the form $\sigma(s'')$ for some basic term s'' . Then again s' is a basic term itself, so this case cannot happen either.
- (v). s' is of the form $\nu(s'')$, where s'' is a basic term. Then one of RDCS1–RDCS5 or RDCSID can be applied, so s' is not a normal form.
- (vi). s' is of the form $[s'']^\omega$ for some basic term s'' . Then one of RATS or RUSD1–RUSD5 can be applied, so s' is not a normal form.

In every case s' is a basic term or a rewriting rule is applicable. If s' is a basic term this contradicts the assumption that it is not. If a rewriting rule is applicable then s' and s are not a normal form. This contradicts the assumption that s is a normal form. From this contradiction we conclude that s is a basic term. ■

Remark 4.3.4.2 (Elimination for BPA_{drt})

Elimination for a somewhat different version of BPA_{drt} is also claimed (without proof) in Section 3.4 of BAETEN AND BERGSTRA [24].

Definition 4.3.4.3 (Symbol for a Chain of σ 's)

We will write $x \xrightarrow{\sigma} y$ to indicate that x can reach y by doing zero or more σ -transitions. Formally, $\xrightarrow{\sigma}$ denotes the transitive, reflexive closure of $\xrightarrow{\sigma}$.

Lemma 4.3.4.4 (Towards Soundness of BPA_{drt}^+)

- (i). Let p and q be closed BPA_{drt} terms and R a bisimulation relation such that $R(p, q)$. Furthermore, suppose that $p \xrightarrow{\sigma} p'$ and $q \xrightarrow{\sigma} q'$. Then we have $R(p', q')$.
- (ii). Let p and q be closed BPA_{drt} terms such that $q \sim_{BPA_{drt}} \nu(p) + \sigma(q)$, and let R be a reflexive, symmetric, transitive bisimulation relation such that $R(q, \nu(p) + \sigma(q))$. Then we have, for all closed terms s such that $q \xrightarrow{\sigma} s$, that $R(q, s)$.

Proof

- (i). By the deduction rules, we have that any closed BPA_{drt} term has at most one outgoing σ -transition. Then, by $p \xrightarrow{\sigma} p'$ and $q \xrightarrow{\sigma} q'$, and the definition of bisimulation, Definition 3.2.1.8, we have that $R(p', q')$.
- (ii). We use induction on the number of σ -transitions in $q \xrightarrow{\sigma} s$. First, the base case where there are zero σ -transitions. Then, $q \equiv s$, and $R(q, s)$ is trivially fulfilled by the reflexivity of R . Secondly, the induction step. Assume that $q \xrightarrow{\sigma} q' \xrightarrow{\sigma} s$, for some closed BPA_{drt} term q' , and that by the induction hypothesis $R(q, q')$. Then, because $q \xrightarrow{\sigma} s'$ for some closed BPA_{drt} term s' , and $q' \xrightarrow{\sigma} s$, by (i) we have $R(s', s)$. Furthermore, because $q \xrightarrow{\sigma} s'$, and $\nu(p) + \sigma(q) \xrightarrow{\sigma} q$, by (i) we have $R(s', q)$. Then, by the symmetry of R we have $R(q, s')$, and by transitivity $R(q, s)$. ■

Theorem 4.3.4.5 (Soundness of BPA_{drt}^+)

The set of closed BPA_{drt} terms modulo bisimulation equivalence is a model of BPA_{drt}^+ .

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. The only new axioms are Axioms ATS, USD, and the conditional axioms RSP(USD). Note that the soundness proofs for Axioms A1–A5, DRT1–DRT5, and DCS1–DCS4 given in the previous sections also remain valid in the setting with delayable actions. This is due to the fact that the underlying model (namely: finite transition systems with σ 's) has not changed. Or, stated more concretely: all σ -transitions were considered without regard for whether they resulted from a σ_{rel} operator or otherwise, so the extra σ -transitions introduced by the delayable actions do not matter.

Axiom ATS Take the relation:

$$R = \{(a, \lfloor \underline{a} \rfloor^\omega)\}$$

There are only two transitions possible on the left-hand side of the axiom: $a \xrightarrow{a} \surd$ and $a \xrightarrow{\sigma} a$. The right-hand side can also perform two transitions: $\lfloor \underline{a} \rfloor^\omega \xrightarrow{a} \surd$ and $\lfloor \underline{a} \rfloor^\omega \xrightarrow{\sigma} \lfloor \underline{a} \rfloor^\omega$, and note that $(a, \lfloor \underline{a} \rfloor^\omega) \in R$. Finally, neither side satisfies the ID predicate: $\neg \text{ID}(a)$ and $\neg \text{ID}(\lfloor \underline{a} \rfloor^\omega)$.

Axiom USD Take the relation:

$$R = \{(s, s), (\lfloor s \rfloor^\omega, \nu(s) + \sigma(\lfloor s \rfloor^\omega)) \mid s \in C(\text{BPA}_{\text{drt}})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose $\lfloor s \rfloor^\omega \xrightarrow{a} p$. Then this must be due to $s \xrightarrow{a} p$. But then also $\nu(s) \xrightarrow{a} p$ and $\nu(s) + \sigma(\lfloor s \rfloor^\omega) \xrightarrow{a} p$, and note that $(p, p) \in R$.
- (ii). Suppose $\lfloor s \rfloor^\omega \xrightarrow{a} \surd$. Then this must be due to $s \xrightarrow{a} \surd$. But then also $\nu(s) \xrightarrow{a} \surd$ and $\nu(s) + \sigma(\lfloor s \rfloor^\omega) \xrightarrow{a} \surd$.
- (iii). Suppose $\lfloor s \rfloor^\omega \xrightarrow{\sigma} p$. Then necessarily $p \equiv \lfloor s \rfloor^\omega$. We also have $\sigma(\lfloor s \rfloor^\omega) \xrightarrow{\sigma} \lfloor s \rfloor^\omega$, hence $\nu(s) + \sigma(\lfloor s \rfloor^\omega) \xrightarrow{\sigma} \lfloor s \rfloor^\omega$, and note that $(\lfloor s \rfloor^\omega, \lfloor s \rfloor^\omega) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose $\nu(s) + \sigma(\lfloor s \rfloor^\omega) \xrightarrow{a} p$. Then this must be due to $s \xrightarrow{a} p$. But then also $\lfloor s \rfloor^\omega \xrightarrow{a} p$, and note that $(p, p) \in R$.
- (ii). Suppose $\nu(s) + \sigma(\lfloor s \rfloor^\omega) \xrightarrow{a} \surd$. Then this must be due to $s \xrightarrow{a} \surd$. But then also $\lfloor s \rfloor^\omega \xrightarrow{a} \surd$.
- (iii). Suppose $\nu(s) + \sigma(\lfloor s \rfloor^\omega) \xrightarrow{\sigma} p$. Then this must be due to $\sigma(\lfloor s \rfloor^\omega) \xrightarrow{\sigma} p$ with $p \equiv \lfloor s \rfloor^\omega$. Clearly, then also $\lfloor s \rfloor^\omega \xrightarrow{\sigma} \lfloor s \rfloor^\omega$, and note that $(\lfloor s \rfloor^\omega, \lfloor s \rfloor^\omega) \in R$.

Finally, we look at the immediate-deadlock predicate. Neither side has immediate deadlock: $\neg \text{ID}(\lfloor s \rfloor^\omega)$ and $\neg \text{ID}(\nu(s) + \sigma(\lfloor s \rfloor^\omega))$ (note that unbounded start delay removes immediate deadlock, see the comment on page 57).

RSP(USD) Suppose that R' is a reflexive, symmetric, transitive bisimulation relation between y and $\nu(x) + \sigma(y)$ (which exists by Lemma 3.2.1.10 on page 45). We must prove that $y \sim_{\text{BPA}_{\text{drt}}} \lfloor x \rfloor^\omega$. To do this, take R' and extend it to a bisimulation relation R between y and $\lfloor x \rfloor^\omega$ as follows:

$$R = R' \cup \{(s, \lfloor x \rfloor^\omega) \mid s \in C(\text{BPA}_{\text{drt}}) \wedge y \xrightarrow{\sigma} s\}$$

Now, let s be any closed term such that $y \xrightarrow{\sigma} s$. Then, by Lemma 4.3.4.4 on the preceding page, we have that $(y, s) \in R'$.

We now prove that R is indeed a bisimulation. Because R' already was a bisimulation, we only have to look at the pairs $(s, \lfloor x \rfloor^\omega)$. First, we look at the transitions of the left-hand side, i.e., the transitions of s :

- (i). Suppose $s \xrightarrow{a} p$. Then, as $(y, s) \in R'$, $y \xrightarrow{a} q$ such that $(p, q) \in R'$. As $(y, \nu(x) + \sigma(y)) \in R'$, we have $\nu(x) + \sigma(y) \xrightarrow{a} r$ for some r such that $(q, r) \in R'$. Hence, $\nu(x) \xrightarrow{a} r$, hence $x \xrightarrow{a} r$, hence $\lfloor x \rfloor^\omega \xrightarrow{a} r$, and note that by transitivity $(p, r) \in R'$, so $(p, r) \in R$.
- (ii). Suppose $s \xrightarrow{a} \surd$. Then, as $(y, s) \in R'$, $y \xrightarrow{a} \surd$. As $(y, \nu(x) + \sigma(y)) \in R'$, we have $\nu(x) + \sigma(y) \xrightarrow{a} \surd$. Hence, $\nu(x) \xrightarrow{a} \surd$, hence $x \xrightarrow{a} \surd$, hence $\lfloor x \rfloor^\omega \xrightarrow{a} \surd$.
- (iii). Suppose $s \xrightarrow{\sigma} p$. Since $y \xrightarrow{\sigma} s$, we have $y \xrightarrow{\sigma} p$. We also have $\lfloor x \rfloor^\omega \xrightarrow{\sigma} \lfloor x \rfloor^\omega$, and note that $(p, \lfloor x \rfloor^\omega) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose $\lfloor x \rfloor^\omega \xrightarrow{a} q$. Then $x \xrightarrow{a} q$, hence $\nu(x) \xrightarrow{a} q$, hence $\nu(x) + \sigma(y) \xrightarrow{a} q$. As $(y, \nu(x) + \sigma(y)) \in R'$, we know that $y \xrightarrow{a} p$ for some p such that $(p, q) \in R'$. Since $(y, s) \in R'$, we also have $s \xrightarrow{a} r$ for some r such that $(p, r) \in R'$, and note that by symmetry $(r, p) \in R'$, by transitivity $(r, q) \in R'$, and therefore $(r, q) \in R$.
- (ii). Suppose $\lfloor x \rfloor^\omega \xrightarrow{a} \surd$. Then $x \xrightarrow{a} \surd$, hence $\nu(x) \xrightarrow{a} \surd$, hence $\nu(x) + \sigma(y) \xrightarrow{a} \surd$. As $(y, \nu(x) + \sigma(y)) \in R'$, we know that $y \xrightarrow{a} \surd$. Since $(y, s) \in R'$, we also have $s \xrightarrow{a} \surd$.
- (iii). Suppose $\lfloor x \rfloor^\omega \xrightarrow{\sigma} q$. Then, it must be the case that $q \equiv \lfloor x \rfloor^\omega$. We also have $\sigma(y) \xrightarrow{\sigma} y$, hence $\nu(x) + \sigma(y) \xrightarrow{\sigma} y$, and as $(y, \nu(x) + \sigma(y)) \in R'$, we know that $y \xrightarrow{\sigma} p$ such that $(p, y) \in R'$. As $(y, s) \in R'$, necessarily $s \xrightarrow{\sigma} r$ for some r such that $(p, r) \in R'$. Since $y \xrightarrow{\sigma} s$, we have $y \xrightarrow{\sigma} r$, and note that $(r, \lfloor x \rfloor^\omega) \in R$.

Finally, we look at the immediate-deadlock predicate. Neither side has immediate deadlock: $\neg \text{ID}(s)$ (because R' is a bisimulation between y and $\nu(x) + \sigma(y)$, and $y \xrightarrow{\sigma} s$) and $\neg \text{ID}(\lfloor x \rfloor^\omega)$.

■

Remark 4.3.4.6 (Soundness of BPA_{drt})

Soundness of a somewhat different version of BPA_{drt} is also claimed (without proof) in Section 3.5 of BAETEN AND BERGSTRA [24].

Lemma 4.3.4.7 (Towards Completeness of $\text{BPA}_{\text{drt}}^+$)

Let x be a closed BPA_{drt} term and let $a \in A$. Then we have:

- (i). $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd \implies \text{BPA}_{\text{drt}}^+ \vdash x = \underline{a} + x$,
- (ii). $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y \implies \text{BPA}_{\text{drt}}^+ \vdash x = \underline{a} \cdot y + x$,
- (iii). $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x) \implies \text{BPA}_{\text{drt}}^+ \vdash x = \underline{\delta}$,
- (iv). $T(\text{BPA}_{\text{drt}}) \models \neg \text{ID}(x) \implies \text{BPA}_{\text{drt}}^+ \vdash x + \underline{\delta} = x$,
- (v). $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} \implies \text{BPA}_{\text{drt}}^+ \vdash x = \nu(x)$,
- (vi). $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y \implies \text{BPA}_{\text{drt}}^+ \vdash x = \sigma(y) + \nu(x)$,

(vii). $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} x \implies \text{BPA}_{\text{drt}}^+ \vdash x = \lfloor x \rfloor^\omega$,

(viii). $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y \implies n(x) > n(y)$,

(ix). $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y \implies x \equiv y \vee n(x) > n(y)$.

Proof For part (i)–(vii) we assume, by Theorem 4.3.4.1 and Theorem 4.3.4.5, without loss of generality, that x is a basic term, and then apply induction on the structure of basic terms. For part (viii) and (ix) we again have to use induction on the general structure of terms.

- (i). Suppose that $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$. Case 1: $x \equiv \delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$, so this case does not occur. Case 2: $x \equiv \underline{b}$, where $b \in A_\delta$. Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$, it must be the case that $b \equiv a$. So we have $\text{BPA}_{\text{drt}}^+ \vdash x = \underline{b} = \underline{b} + \underline{b} = \underline{a} + \underline{b} = \underline{a} + x$. Case 3: $x \equiv \underline{b}$, where $b \in A_\delta$. Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$, it must be the case that $b \equiv a$. So we have $\text{BPA}_{\text{drt}}^+ \vdash x = b = \lfloor \underline{b} \rfloor^\omega = \nu(\underline{b}) + \sigma(\lfloor \underline{b} \rfloor^\omega) = \nu(\underline{b}) + \nu(\underline{b}) + \sigma(\lfloor \underline{b} \rfloor^\omega) = \underline{b} + \lfloor \underline{b} \rfloor^\omega = \underline{a} + b = \underline{a} + x$. Case 4: $x \equiv \underline{b} \cdot x'$, where $b \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$, so this case does not occur. Case 5: $x \equiv b \cdot x'$, where $b \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$, so this case does not occur. Case 6: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$, necessarily $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{a} \surd$ or $T(\text{BPA}_{\text{drt}}) \models x'' \xrightarrow{a} \surd$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^+ \vdash x' = \underline{a} + x'$ or $\text{BPA}_{\text{drt}}^+ \vdash x'' = \underline{a} + x''$. But then in both cases $\text{BPA}_{\text{drt}}^+ \vdash x = x' + x'' = \underline{a} + x' + x'' = \underline{a} + x$. Case 7: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} \surd$, so this case does not occur.
- (ii). Suppose that $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$. Case 1: $x \equiv \delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur. Case 2: $x \equiv \underline{b}$, where $b \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur. Case 3: $x \equiv b$, where $b \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur. Case 4: $x \equiv \underline{b} \cdot x'$, where $b \in A_\delta$ and x' is a basic term. Then, because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, it must be that $b \equiv a$ and $x' \equiv y$. So, $\text{BPA}_{\text{drt}}^+ \vdash x = x + x = \underline{b} \cdot x' + x = \underline{a} \cdot y + x$. Case 5: $x \equiv b \cdot x'$, where $b \in A_\delta$ and x' is a basic term. Then, because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, it must be that $b \equiv a$ and $x' \equiv y$. So, $\text{BPA}_{\text{drt}}^+ \vdash x = b \cdot x' = a \cdot y = (\underline{a} + a) \cdot y = \underline{a} \cdot y + a \cdot y = \underline{a} \cdot y + x$. Case 6: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{a} y$ or $T(\text{BPA}_{\text{drt}}) \models x'' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^+ \vdash x' = \underline{a} \cdot y + x'$ or $\text{BPA}_{\text{drt}}^+ \vdash x'' = \underline{a} \cdot y + x''$. But then in both cases $\text{BPA}_{\text{drt}}^+ \vdash x = x' + x'' = \underline{a} \cdot y + x' + x'' = \underline{a} \cdot y + x$. Case 7: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur.
- (iii). Suppose that $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x)$. Case 1: $x \equiv \delta$. Then $\text{BPA}_{\text{drt}}^+ \vdash x = \delta$ is trivially fulfilled. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x)$, so this case does not occur. Case 3: $x \equiv a$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x)$, so this case does not occur. Case 4: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x)$, so this case does not occur. Case 5: $x \equiv a \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x)$, so this case does not occur. Case 6: $x \equiv x' + x''$, where x' and x'' are basic terms. Then, because $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x)$, it must be the

case that $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x'), \text{ID}(x'')$. So, by the induction hypothesis, we have that $\text{BPA}_{\text{drt}}^+ \vdash x' = \dot{\delta}, x'' = \dot{\delta}$. But then also $\text{BPA}_{\text{drt}}^+ \vdash x = x' + x'' = \dot{\delta} + \dot{\delta} = \dot{\delta}$. Case 7: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models \text{ID}(x)$, so this case does not occur.

(iv). Suppose that $T(\text{BPA}_{\text{drt}}) \models \neg \text{ID}(x)$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models \neg \text{ID}(x)$, so this case does not occur. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. $\text{BPA}_{\text{drt}}^+ \vdash x + \underline{\delta} = \underline{a} + \underline{\delta} = \underline{a} = x$. Case 3: $x \equiv a$, where $a \in A_\delta$. Then we have $\text{BPA}_{\text{drt}}^+ \vdash x + \underline{\delta} = a + \underline{\delta} = \lfloor \underline{a} \rfloor^\omega + \underline{\delta} = \nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega) + \underline{\delta} = \underline{a} + \underline{\delta} + \sigma(\lfloor \underline{a} \rfloor^\omega) = \underline{a} + \sigma(\lfloor \underline{a} \rfloor^\omega) = \nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega) = \lfloor \underline{a} \rfloor^\omega = a = x$. Case 4: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. $\text{BPA}_{\text{drt}}^+ \vdash x + \underline{\delta} = \underline{a} \cdot x' + \underline{\delta} = \underline{a} \cdot x' + \underline{\delta} \cdot x' = (\underline{a} + \underline{\delta}) \cdot x' = \underline{a} \cdot x' = x$. Case 5: $x \equiv a \cdot x'$, where $a \in A_\delta$ and x' is a basic term. Then, using Case 3, $\text{BPA}_{\text{drt}}^+ \vdash x + \underline{\delta} = a \cdot x' + \underline{\delta} = a \cdot x' + \underline{\delta} \cdot x' = (a + \underline{\delta}) \cdot x' = a \cdot x' = x$. Case 6: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\overline{\text{BPA}}_{\text{drt}}) \models \neg \text{ID}(x)$, necessarily $T(\text{BPA}_{\text{drt}}) \models \neg \text{ID}(x')$ or $T(\text{BPA}_{\text{drt}}) \models \neg \text{ID}(x'')$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^+ \vdash x' + \underline{\delta} = x'$ or $\text{BPA}_{\text{drt}}^+ \vdash x'' + \underline{\delta} = x''$. So, in both cases, $\text{BPA}_{\text{drt}}^+ \vdash x + \underline{\delta} = x' + x'' + \underline{\delta} = x' + x'' = x$. Case 7: $x \equiv \sigma(x')$, where x' is a basic term. Then we have $\text{BPA}_{\text{drt}}^+ \vdash x + \underline{\delta} = \sigma(x') + \underline{\delta} = \sigma(x') = x$.

(v). Suppose that $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma}$. Case 1: $x \equiv \dot{\delta}$. By Axiom DCSID we have $\text{BPA}_{\text{drt}}^+ \vdash x = \dot{\delta} = \nu(\dot{\delta}) = \nu(x)$. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. We have $\text{BPA}_{\text{drt}}^+ \vdash x = \underline{a} = \nu(\underline{a}) = \nu(x)$. Case 3: $x \equiv a$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma}$, so this case does not occur. Case 4: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. We have $\text{BPA}_{\text{drt}}^+ \vdash x = \underline{a} \cdot x' = \nu(\underline{a}) \cdot x' = \nu(\underline{a} \cdot x') = \nu(x)$. Case 5: $x \equiv a \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma}$, so this case does not occur. Case 6: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma}$, necessarily $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{\sigma}$ and $T(\text{BPA}_{\text{drt}}) \models x'' \xrightarrow{\sigma}$. Therefore, by the induction hypothesis, $\text{BPA}_{\text{drt}}^+ \vdash x' = \nu(x')$ and $\text{BPA}_{\text{drt}}^+ \vdash x'' = \nu(x'')$. But then also $\text{BPA}_{\text{drt}}^+ \vdash x = x' + x'' = \nu(x') + \nu(x'') = \nu(x' + x'') = \nu(x)$. Case 7: $x \equiv \sigma(x')$, where x' is a basic term. Then, by $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma}$, it must be the case that $\text{ID}(x')$. So, by (iii), we have that $\text{BPA}_{\text{drt}}^+ \vdash x' = \dot{\delta}$. Therefore, $\text{BPA}_{\text{drt}}^+ \vdash x = \sigma(x') = \sigma(\dot{\delta}) = \underline{\delta} = \nu(\sigma(x')) = \nu(x)$.

(vi). Suppose that $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$. Case 1: $x \equiv \dot{\delta}$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 3: $x \equiv a$, where $a \in A_\delta$. Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, it must be the case that $y \equiv a$. So we have $\text{BPA}_{\text{drt}}^+ \vdash x = a = \lfloor \underline{a} \rfloor^\omega = \nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega) = \sigma(a) + \nu(\underline{a}) + \underline{\delta} = \sigma(a) + \nu(\underline{a}) + \nu(\sigma(\lfloor \underline{a} \rfloor^\omega)) = \sigma(a) + \nu(\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) = \sigma(a) + \nu(\lfloor \underline{a} \rfloor^\omega) = \sigma(a) + \nu(a) = \sigma(y) + \nu(x)$. Case 4: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 5: $x \equiv a \cdot x'$, where $a \in A_\delta$ and x' is a basic term. Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, it must be the case that $y \equiv a \cdot x'$. So we have $\text{BPA}_{\text{drt}}^+ \vdash x = a \cdot x' = \lfloor \underline{a} \rfloor^\omega \cdot x' = (\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \cdot x' = \nu(\underline{a}) \cdot x' + \sigma(\lfloor \underline{a} \rfloor^\omega) \cdot x' = \sigma(\lfloor \underline{a} \rfloor^\omega) \cdot x' + \nu(\underline{a}) \cdot x' + \underline{\delta} = \sigma(a \cdot x') + \nu(\underline{a}) \cdot x' + \underline{\delta} \cdot x' = \sigma(y) + (\nu(\underline{a}) + \underline{\delta}) \cdot x' = \sigma(y) + (\nu(\underline{a}) + \nu(\sigma(\lfloor \underline{a} \rfloor^\omega))) \cdot x' = \sigma(y) + \nu(\underline{a} + \sigma(\lfloor \underline{a} \rfloor^\omega)) \cdot x' = \sigma(y) + \nu(\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \cdot x' = \sigma(y) + \nu(\lfloor \underline{a} \rfloor^\omega) \cdot x' = \sigma(y) + \nu(a) \cdot x' = \sigma(y) + \nu(a \cdot x') = \sigma(y) + \nu(x)$. Case 6: $x \equiv x' + x''$, where x' and x'' are basic terms. As $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, necessarily (1) $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma}$, or, (2) $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{\sigma}, x'' \xrightarrow{\sigma} y$, or, (3) $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{\sigma} y', x'' \xrightarrow{\sigma} y'$ where $y \equiv y' + y''$.

In the first case, by the induction hypothesis, we have $\text{BPA}_{\text{drt}}^+ \vdash x' = \sigma(y) + \nu(x')$, and, by (v), $\text{BPA}_{\text{drt}}^+ \vdash x'' = \nu(x'')$. Therefore, $\text{BPA}_{\text{drt}}^+ \vdash x = x' + x'' = \sigma(y) + \nu(x') + \nu(x'') = \sigma(y) + \nu(x' + x'') = \sigma(y) + \nu(x)$. The second case is treated analogously. In the third case we have, by the induction hypothesis, $\text{BPA}_{\text{drt}} \vdash x' = \sigma(y') + \nu(x')$, $x'' = \sigma(y'') + \nu(x'')$. Therefore we have $\text{BPA}_{\text{drt}}^+ \vdash x = x' + x'' = \sigma(y') + \nu(x') + \sigma(y'') + \nu(x'') = \sigma(y' + y'') + \nu(x' + x'') = \sigma(y) + \nu(x)$. Case 7: $x \equiv \sigma(x')$, where x' is a basic term. Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, it must be the case that $x' \equiv y$. So we have $\text{BPA}_{\text{drt}}^+ \vdash x = \sigma(x') = \sigma(y) = \sigma(y) + \underline{\delta} = \sigma(y) + \nu(\sigma(x')) = \sigma(y) + \nu(x)$.

(vii). Suppose that $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} x$. Case 1: $x \equiv \underline{\delta}$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} x$, so this case does not occur. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} x$, so this case does not occur. Case 3: $x \equiv a$, where $a \in A_\delta$. Then we have, using Proposition 3.2.4.14(i), $\text{BPA}_{\text{drt}}^+ \vdash x = a = [a]^\omega = [x]^\omega$. Case 4: $x \equiv \underline{a} \cdot x'$, where $a \in A_\delta$ and x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} x$, so this case does not occur. Case 5: $x \equiv a \cdot x'$, where $a \in A_\delta$ and x' is a basic term. Then we can derive, using Proposition 3.2.4.14(i) and (ii), $\text{BPA}_{\text{drt}}^+ \vdash x = a \cdot x' = [a]^\omega \cdot x' = [a \cdot x']^\omega = [x]^\omega$. Case 6: $x \equiv x' + x''$, where x' and x'' are basic terms. Then we can derive, using Proposition 3.2.4.14(iii) and the induction hypothesis, $\text{BPA}_{\text{drt}}^+ \vdash x = x' + x'' = [x']^\omega + [x'']^\omega = [x' + x'']^\omega = [x]^\omega$. Case 7: $x \equiv \sigma(x')$, where x' is a basic term. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} x$, so this case does not occur.

(viii). Suppose that $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$. Case 1: $x \equiv \underline{\delta}$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur. Case 2: $x \equiv \underline{b}$, where $b \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur. Case 3: $x \equiv b$, where $b \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur. Case 4: $x \equiv x' \cdot x''$, for certain terms x' and x'' . Then, because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, we either have $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{a} \surd$ and $y \equiv x''$, or we have $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{a} x'''$ and $y \equiv x''' \cdot x''$ for some term x''' . In the first case, we have $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x'') = n(y)$, and in the second we can apply the induction hypothesis to arrive at $n(x') > n(x''')$, so we get $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x''') + n(x'') + 1 = n(x''' \cdot x'') = n(y)$. Case 5: $x \equiv x' + x''$, for certain terms x' and x'' . Since $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{a} y$ or $T(\text{BPA}_{\text{drt}}) \models x'' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(x') > n(y)$ or $n(x'') > n(y)$. In both cases $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y)$. Case 6: $x \equiv \sigma(x')$, for a certain term x' . This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, so this case does not occur. Case 7: $x \equiv \nu(x')$, for a certain term x' . Since $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(x') > n(y)$. So, $n(x) = n(\nu(x')) = n(x') + 1 > n(y)$. Case 8: $x \equiv [x']^\omega$, for a certain term x' . Since $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{a} y$, necessarily $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{a} y$. Therefore, by the induction hypothesis, $n(x') > n(y)$. So, $n(x) = n([x']^\omega) = n(x') + 1 > n(y)$.

(ix). Suppose that $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$. Case 1: $x \equiv \underline{\delta}$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 2: $x \equiv \underline{a}$, where $a \in A_\delta$. This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 3: $x \equiv a$, where $a \in A_\delta$. Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, it must be the case that $x \equiv y$, and we are done. Case 4: $x \equiv x' \cdot x''$, for certain terms x' and x'' . Then necessarily, $x' \xrightarrow{\sigma} x'''$ and

$y \equiv x''' \cdot x''$ for some term x''' . We now can apply the induction hypothesis to arrive at $x' \equiv x'''$ or $n(x') > n(x''')$, so we get $x \equiv y$ or $n(x) = n(x' \cdot x'') = n(x') + n(x'') + 1 > n(x''') + n(x'') + 1 = n(x''' \cdot x'') = n(y)$. Case 5: $x \equiv x' + x''$, for certain terms x' and x'' . As $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, necessarily (1) $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{\sigma} y, x'' \xrightarrow{\sigma}$, or, (2) $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{\sigma}, x'' \xrightarrow{\sigma} y$, or, (3) $T(\text{BPA}_{\text{drt}}) \models x' \xrightarrow{\sigma} y', x'' \xrightarrow{\sigma} y''$ where $y \equiv y' + y''$. In the first case, by the induction hypothesis, $n(x') > n(y)$. So $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y)$. The second case is treated analogously. In the third case, by the induction hypothesis, $n(x') > n(y')$ and $n(x'') > n(y'')$. So $n(x) = n(x' + x'') = n(x') + n(x'') + 1 > n(y') + n(y'') + 1 = n(y)$. Case 6: $x \equiv \sigma(x')$, for a certain term x' . Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, it must be the case that $x' \equiv y$. Then we have $n(x) = n(\sigma(x')) = n(x') + 1 = n(y) + 1 > n(y)$. Case 7: $x \equiv \nu(x')$, for a certain term x' . This is in contradiction with $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, so this case does not occur. Case 8: $x \equiv [x']^\omega$, for a certain term x' . Because $T(\text{BPA}_{\text{drt}}) \models x \xrightarrow{\sigma} y$, it must be the case that $x \equiv y$, and we are done. ■

Remark 4.3.4.8 (Towards Completeness of $\text{BPA}_{\text{drt}}^+$)

Note that Lemma 3.2.4.19 on page 64 now also follows as a corollary from Lemma 4.3.4.7(iii) and (iv) by the law of the excluded middle.

Theorem 4.3.4.9 (Completeness of $\text{BPA}_{\text{drt}}^+$)

The axiom system $\text{BPA}_{\text{drt}}^+$ is a complete axiomatization of the set of closed BPA_{drt} terms modulo bisimulation equivalence.

Proof We use the direct method described in Proof Outline 4.2.3.1 on page 70. Suppose that $s + t \sim_{\text{BPA}_{\text{drt}}} t$. We then prove that $\text{BPA}_{\text{drt}}^+ \vdash s + t = t$. By Theorem 4.3.4.1 we can restrict ourselves to basic terms s and t . The proof is done with induction on $n(s)$, using Lemma 4.3.4.7(viii)–(ix) and case distinction on the form of basic term s .

- (i). $s \equiv \dot{\delta}$. Using Axiom A6ID we have $\text{BPA}_{\text{drt}}^+ \vdash s + t = \dot{\delta} + t = t + \dot{\delta} = t$.
- (ii). $s \equiv \underline{\delta}$. Then we have $\neg \text{ID}(s + t)$, because $T(\text{BPA}_{\text{drt}}) \models \neg \text{ID}(s)$. Since $s + t \sim_{\text{BPA}_{\text{drt}}} t$, we also have $T(\text{BPA}_{\text{drt}}) \models \neg \text{ID}(t)$. Using Lemma 4.3.4.7(iv) we have $\text{BPA}_{\text{drt}}^+ \vdash s + t = \underline{\delta} + t = t + \underline{\delta} = t$.
- (iii). $s \equiv \underline{a}$, where $a \in A$. From the deduction rules we have $T(\text{BPA}_{\text{drt}}) \models s \xrightarrow{a} \surd$ and $T(\text{BPA}_{\text{drt}}) \models s + t \xrightarrow{a} \surd$. Since $s + t \sim_{\text{BPA}_{\text{drt}}} t$ we also have $T(\text{BPA}_{\text{drt}}) \models t \xrightarrow{a} \surd$. By Lemma 4.3.4.7(i) we obtain $\text{BPA}_{\text{drt}}^+ \vdash t = \underline{a} + t$. So, $\text{BPA}_{\text{drt}}^+ \vdash s + t = \underline{a} + t = t$.
- (iv). $s \equiv \delta$. Then $\delta \xrightarrow{\sigma} \delta$. Therefore $s + t \xrightarrow{\sigma} s + t'$ and $t \xrightarrow{\sigma} t'$ with $s + t' \sim_{\text{BPA}_{\text{drt}}} t'$. With Lemma 4.3.4.7(vi) we have $\text{BPA}_{\text{drt}}^+ \vdash t = \sigma(t') + \nu(t)$. Two cases need to be considered:
 - (a) $t \equiv t'$. Now, $s + t \xrightarrow{\sigma} s + t$ and $t \xrightarrow{\sigma} t$, so by Lemma 4.3.4.7(vii) we have $\text{BPA}_{\text{drt}}^+ \vdash s + t = [s + t]^\omega$ and $\text{BPA}_{\text{drt}}^+ \vdash t = [t]^\omega$. So we can derive, using Proposition 3.2.4.14(i) and (iii) and Lemma 4.3.4.7(iv): $\text{BPA}_{\text{drt}}^+ \vdash s + t = [s + t]^\omega = [s]^\omega + [t]^\omega = [\delta]^\omega + [t]^\omega = \delta + [t]^\omega = [\underline{\delta}]^\omega + [t]^\omega = [\underline{\delta} + t]^\omega = [t]^\omega = t$.

- (b) $t \neq t'$. Now, by Lemma 4.3.4.7(ix), $n(t') < n(t)$. Therefore, the induction hypothesis is applicable: $\text{BPA}_{\text{drt}}^+ \vdash \delta + t' = t'$. Consider the following computation: $\text{BPA}_{\text{drt}}^+ \vdash s + t = \delta + t = \lfloor \underline{\delta} \rfloor^\omega + t = \nu(\underline{\delta}) + \sigma(\lfloor \underline{\delta} \rfloor^\omega) + t = \underline{\delta} + \sigma(\delta) + t = \sigma(\delta) + t = \sigma(\delta) + \sigma(t') + \nu(t) = \sigma(\delta + t') + \nu(t) = \sigma(t') + \nu(t) = t$.
- (v). $s \equiv a$, where $a \in A$. Then $s \xrightarrow{a} \surd$. Therefore $s + t \xrightarrow{a} \surd$ and, since $s + t \sim_{\text{BPA}_{\text{drt}}} t$, $t \xrightarrow{a} \surd$. Using Lemma 4.3.4.7(i) we obtain $\text{BPA}_{\text{drt}}^+ \vdash t = \underline{a} + t$. We also have $s \xrightarrow{\sigma} s$. Therefore $s + t \xrightarrow{\sigma} s + t'$ and $t \xrightarrow{\sigma} t'$. From Lemma 4.3.4.7(vi) we obtain: $\text{BPA}_{\text{drt}}^+ \vdash t = \sigma(t') + \nu(t)$. Two cases can be distinguished:
- (a) $t \equiv t'$. Now, $s + t \xrightarrow{\sigma} s + t$ and $t \xrightarrow{\sigma} t$, so by Lemma 4.3.4.7(vii) we have $\text{BPA}_{\text{drt}}^+ \vdash s + t = \lfloor s + t \rfloor^\omega$ and $\text{BPA}_{\text{drt}}^+ \vdash t = \lfloor t \rfloor^\omega$. So we can derive, using Proposition 3.2.4.14(i) and (iii): $\text{BPA}_{\text{drt}}^+ \vdash s + t = \lfloor s + t \rfloor^\omega = \lfloor s \rfloor^\omega + \lfloor t \rfloor^\omega = \lfloor a \rfloor^\omega + \lfloor t \rfloor^\omega = a + \lfloor t \rfloor^\omega = \lfloor \underline{a} \rfloor^\omega + \lfloor t \rfloor^\omega = \lfloor \underline{a} + t \rfloor^\omega = \lfloor t \rfloor^\omega = t$.
- (b) $t \neq t'$. Now, by Lemma 4.3.4.7(ix), $n(t') < n(t)$. Therefore the induction hypothesis is applicable: $\text{BPA}_{\text{drt}}^+ \vdash a + t' = t'$. Consider the following computation: $\text{BPA}_{\text{drt}}^+ \vdash s + t = a + t = \lfloor \underline{a} \rfloor^\omega + t = \nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega) + t = \underline{a} + \sigma(a) + t = \sigma(a) + t = \sigma(a) + \sigma(t') + \nu(t) = \sigma(a + t') + \nu(t) = \sigma(t') + \nu(t) = t$.
- (vi). $s \equiv \underline{\delta} \cdot s'$, where s' is a basic term. Then we have $\text{BPA}_{\text{drt}}^+ \vdash s = \underline{\delta} \cdot s' = \underline{\delta}$ and, using (ii), $\text{BPA}_{\text{drt}}^+ \vdash s + t = t$.
- (vii). $s \equiv \underline{a} \cdot s'$, where $a \in A$ and s' is a basic term. From the deduction rules we obtain $T(\text{BPA}_{\text{drt}}) \vDash s \xrightarrow{a} s'$ and $T(\text{BPA}_{\text{drt}}) \vDash s + t \xrightarrow{a} s'$. Since $s + t \sim_{\text{BPA}_{\text{drt}}} t$, we then also have $T(\text{BPA}_{\text{drt}}) \vDash t \xrightarrow{a} t'$ for some t' such that $s' \sim_{\text{BPA}_{\text{drt}}} t'$. By the induction hypothesis we have $\text{BPA}_{\text{drt}}^+ \vdash s' = t'$. From Lemma 4.3.4.7(ii) we have $\text{BPA}_{\text{drt}}^+ \vdash t = \underline{a} \cdot t' + t$. So, $\text{BPA}_{\text{drt}}^+ \vdash s + t = \underline{a} \cdot s' + t = \underline{a} \cdot t' + t = t$.
- (viii). $s \equiv \delta \cdot s'$, where s' is a basic term. Then we have, using Proposition 3.2.4.16, $\text{BPA}_{\text{drt}}^+ \vdash s = \delta \cdot s' = \delta$ and, using (iv), $\text{BPA}_{\text{drt}}^+ \vdash s + t = t$.
- (ix). $s \equiv a \cdot s'$, where $a \in A$ and s' is a basic term. Then $s \xrightarrow{a} s'$ and $s + t \xrightarrow{a} s'$. Since $s + t \sim_{\text{BPA}_{\text{drt}}} t$ we also have $t \xrightarrow{a} t'$ for some t' such that $s' \sim_{\text{BPA}_{\text{drt}}} t'$. By induction we therefore have $\text{BPA}_{\text{drt}}^+ \vdash s' = t'$. We also have $s \xrightarrow{\sigma} s$ and $s + t \xrightarrow{\sigma} s + t''$ and $t \xrightarrow{\sigma} t''$. By Lemma 4.3.4.7(ii) we have $\text{BPA}_{\text{drt}}^+ \vdash t = \underline{a} \cdot t' + t$ and $\text{BPA}_{\text{drt}}^+ \vdash t = \sigma(t'') + \nu(t)$. Two cases can be distinguished:
- (a) $t \equiv t''$. Now, $s + t \xrightarrow{\sigma} s + t$ and $t \xrightarrow{\sigma} t$, so by Lemma 4.3.4.7(vii) we have $\text{BPA}_{\text{drt}}^+ \vdash s + t = \lfloor s + t \rfloor^\omega$ and $\text{BPA}_{\text{drt}}^+ \vdash t = \lfloor t \rfloor^\omega$. So we can derive, using Proposition 3.2.4.14(i)-(iii): $\text{BPA}_{\text{drt}}^+ \vdash s + t = \lfloor s + t \rfloor^\omega = \lfloor s \rfloor^\omega + \lfloor t \rfloor^\omega = \lfloor a \cdot s' \rfloor^\omega + \lfloor t \rfloor^\omega = \lfloor a \rfloor^\omega \cdot \lfloor s' \rfloor^\omega + \lfloor t \rfloor^\omega = a \cdot s' + \lfloor t \rfloor^\omega = \lfloor \underline{a} \rfloor^\omega \cdot s' + \lfloor t \rfloor^\omega = \lfloor \underline{a} \cdot s' \rfloor^\omega + \lfloor t \rfloor^\omega = \lfloor \underline{a} \cdot s' + t \rfloor^\omega = \lfloor \underline{a} \cdot t' + t \rfloor^\omega = \lfloor t \rfloor^\omega = t$.
- (b) $t \neq t''$. Now, by Lemma 4.3.4.7(ix), $n(t'') < n(t)$. By the induction hypothesis we then have $\text{BPA}_{\text{drt}}^+ \vdash s + t'' = t''$. Consider the following computation: $\text{BPA}_{\text{drt}}^+ \vdash s + t = a \cdot s' + t = \lfloor \underline{a} \rfloor^\omega \cdot s' + t = (\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \cdot s' + t = (\underline{a} + \sigma(a)) \cdot s' + t = \underline{a} \cdot t' + \sigma(a) \cdot s' + t = \sigma(a) \cdot s' + t = \sigma(a \cdot s') + t = \sigma(s) + t = \sigma(s) + \sigma(t'') + \nu(t) = \sigma(s + t'') + \nu(t) = \sigma(t'') + \nu(t) = t$.

- (x). $s \equiv s' + s''$, where s' and s'' are basic terms. Since $s' + s'' + t \sim_{\text{BPA}_{\text{drt}}} t$, we also have $s' + t \sim_{\text{BPA}_{\text{drt}}} t$ and $s'' + t \sim_{\text{BPA}_{\text{drt}}} t$. By the induction hypothesis we then have $\text{BPA}_{\text{drt}}^+ \vdash s' + t = t, s'' + t = t$. So, $\text{BPA}_{\text{drt}}^+ \vdash s + t = s' + s'' + t = s' + t = t$.
- (xi). $s \equiv \sigma(s')$, where s' is a basic term. From the deduction rules we have $T(\text{BPA}_{\text{drt}}) \models \sigma(s') \xrightarrow{\sigma} s'$ and since $s + t \sim_{\text{BPA}_{\text{drt}}} t$ we also have $T(\text{BPA}_{\text{drt}}) \models t \xrightarrow{\sigma} t', s + t \xrightarrow{\sigma} s' + t'$ for some t' such that $s' + t' \sim_{\text{BPA}_{\text{drt}}} t'$. By Lemma 4.3.4.7(vi) we have $\text{BPA}_{\text{drt}}^+ \vdash t = \sigma(t') + \nu(t)$. By the induction hypothesis we have $\text{BPA}_{\text{drt}}^+ \vdash s' + t' = t'$. So, $\text{BPA}_{\text{drt}}^+ \vdash s + t = \sigma(s') + t = \sigma(s') + \sigma(t') + \nu(t) = \sigma(s' + t') + \nu(t) = \sigma(t') + \nu(t) = t$.

■

Remark 4.3.4.10 (Completeness of BPA_{drt})

Completeness of a somewhat different version of BPA_{drt} is also claimed (without proof) in Section 3.5 of BAETEN AND BERGSTRA [24].

4.3.5 BPA'_{drt}

In this section, we define a process algebra named BPA'_{drt} , that is almost identical to BPA_{drt} , except that it has five additional axioms. These are chosen such that elimination, soundness, and completeness results for BPA'_{drt} follow as corollaries from the corresponding results for $\text{BPA}_{\text{drt}}^+$.

Definition 4.3.5.1 (Axioms of BPA'_{drt})

The process algebra BPA'_{drt} is axiomatized by the axioms of BPA_{drt} given in Definition 3.2.4.5 on page 55, and Axioms USD1–USD5 shown in Table 4.5: $\text{BPA}'_{\text{drt}} = \text{A1–A5} + \text{A6ID–A7ID} + \text{DRT1–DRT5} + \text{DRTSID} + \text{DCS1–DCS4} + \text{DCSID} + \text{ATS} + \text{USD} + \text{USD1–USD5}$.

$[a]^\omega = a$	USD1
$[x \cdot y]^\omega = [x]^\omega \cdot y$	USD2
$[x + y]^\omega = [x]^\omega + [y]^\omega$	USD3
$[\sigma_{\text{rel}}(x)]^\omega = \delta$	USD4
$[\dot{\delta}]^\omega = \delta$	USD5

Table 4.5: Additional axioms for unbounded start delay.

☞ Axioms USD1–USD5 precisely correspond to equalities (i)–(v) of Proposition 3.2.4.14 on page 58. In this way, we obtain an axiomatization that is in many ways (elimination, soundness, completeness) like $\text{BPA}_{\text{drt}}^+$, but is also purely equational, i.e., does not contain conditional axioms or recursion principles.

Definition 4.3.5.2 (Signature, Semantics, and Basic Terms of BPA'_{drt})

The signature, semantics, bisimulation, bisimulation model, and basic terms of BPA'_{drt} are the same as those of BPA_{drt} .

Corollary 4.3.5.3 (Elimination for BPA'_{drt})

Let t be a closed BPA'_{drt} term. Then there is a basic term s such that $BPA'_{drt} \vdash t = s$.

Proof In the same way as Theorem 4.3.4.1. Note that all rewriting rules of Table 4.4 correspond to derivable equalities in BPA'_{drt} . ■

Corollary 4.3.5.4 (Soundness of BPA'_{drt})

The set of closed BPA'_{drt} terms modulo bisimulation equivalence is a model of BPA'_{drt} .

Proof We use the indirect method of Proof Outline 4.2.2.2 on page 70. The result follows directly from the soundness of BPA^+_{drt} (see Theorem 4.3.4.5 on page 94) combined with the fact that Axioms USD1–USD5 are, for closed terms, derivable in BPA^+_{drt} (see Proposition 3.2.4.14 on page 58). ■

Corollary 4.3.5.5 (Completeness of BPA'_{drt})

The axiom system BPA'_{drt} is a complete axiomatization of the set of closed BPA'_{drt} terms modulo bisimulation equivalence.

Proof We use the indirect method of Proof Outline 4.2.3.2 on page 71. Careful inspection of the dependencies between the proofs in this section reveals that the proof of Theorem 4.3.4.9 only relies upon RSP(USD) to ensure Proposition 3.2.4.14(i)–(v). So, we obviously do not need RSP(USD) anymore if we add the corresponding Axioms USD1–USD5, and the result follows. ■

4.4 Conclusions

Since this chapter is closely related with the following chapter, we do not give conclusions for this chapter separately. Instead, we will give conclusions for both chapters at the end of Chapter 5, in Section 5.5.

5

Axioms for Concurrency

5.1 Introduction

In this chapter we will examine the various possibilities to extend the discrete-time basic process algebras of Chapter 3 with merge operators. In Section 5.2 we will do this for the process algebra BPA_{drt}^- -ID of Section 3.2.2, extending it with the free merge operator to get PA_{drt}^- -ID, and with the merge operator to get ACP_{drt}^- -ID. Then we do this again, in a different way, leading to the alternative process algebras PA_{drt}^- -ID' and ACP_{drt}^- -ID'.

In Section 5.3 we will extend the process algebra BPA_{drt}^+ , that contains delayable actions and the immediate deadlock, to define process algebras PA_{drt}^+ and ACP_{drt}^+ . We show how we can replace the recursion principle of PA_{drt}^+ by a set of unconditional axioms, like we did for BPA_{drt}^+ in Section 4.3.5, leading to the process algebra PA'_{drt} . For ACP_{drt}^+ we do the same in even two ways, leading to the process algebras ACP'_{drt} and ACP''_{drt} .

For all process algebras in this chapter, we give motivated axioms, a bisimulation model, and elimination, soundness, and completeness results.

5.2 Process Algebras with Undelayable Actions

We introduce the process algebras PA_{drt}^- -ID, PA_{drt}^- -ID', ACP_{drt}^- -ID, and ACP_{drt}^- -ID'. These are all based on BPA_{drt}^- -ID, and hence do not contain delayable actions or the immediate deadlock.

5.2.1 PA_{drt}^- -ID

In this section, we extend BPA_{drt}^- -ID to PA_{drt}^- -ID by introducing axioms for the left merge that are based on the representation of BPA_{drt}^- -ID terms shown in Lemma 3.2.2.10 on page 50. This is special, as normally new operators are axiomatized directly on the signature (like the encapsulation operator, see Axioms D1-D4 shown in Table 2.18 on page 29), or on the basis of an inductive definition of basic terms (like the untimed left merge, see Axioms M2-M4 shown in Table 2.12 on page 21).

Note that the approach we take in this section has intriguing correspondences with the approach taken by NICOLLIN AND SIFAKIS [154]; see Section 8.6.1 for a discussion about the similarities and differences between the two approaches.

Definition 5.2.1.1 (Signature of PA_{drt}^- -ID)

The signature of PA_{drt}^- -ID consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , the “now” operator ν_{rel} , the *free merge operator* \parallel , and the *left merge operator* \ll .

Definition 5.2.1.2 (Axioms of PA_{drt}^- -ID)

The process algebra PA_{drt}^- -ID is axiomatized by the axioms of $\text{BPA}_{\text{drt}}^-$ -ID given in Definition 3.2.2.2 on page 47, Axioms DRTM1–DRTM4 shown in Table 5.1, and Axioms DRTM5–DRTM6 shown in Table 5.2: PA_{drt}^- -ID = A1–A5 + DRT1–DRT5 + DCS1–DCS4 + DRTM1–DRTM6.

$x \parallel y = x \ll y + y \ll x$	DRTM1
$\underline{a} \ll x = \underline{a} \cdot x$	DRTM2
$\underline{a} \cdot x \ll y = \underline{a} \cdot (x \parallel y)$	DRTM3
$(x + y) \ll z = x \ll z + y \ll z$	DRTM4

Table 5.1: Axioms for free merge.

$\sigma_{\text{rel}}(x) \ll \nu_{\text{rel}}(y) = \underline{\delta}$	DRTM5
$\sigma_{\text{rel}}(x) \ll (\nu_{\text{rel}}(y) + \sigma_{\text{rel}}(z)) = \sigma_{\text{rel}}(x \ll z)$	DRTM6

Table 5.2: Additional axioms for PA_{drt}^- -ID.

☞ Axioms DRTM1–DRTM4 are straightforward reformulations of their untimed counterparts M1–M4, where the untimed action a has been replaced by the undelayable action \underline{a} . So, within time-slices, i.e., between time steps, the left merge behaves as in the untimed case.

Axioms DRTM5 and DRTM6 are more interesting. They handle the time-step behavior of the left merge. DRTM5 expresses that when the left argument of the left merge can only do a time step, and the right hand cannot do a time step, then the left merge ends in undelayable deadlock, as the left argument cannot do its initial action before the right argument does. DRTM6 expresses that when the left argument of the left merge can only do a time step, and the right argument can do at least a time step, then the whole will do a time step, and continue as the left merge of the respective arguments after they have done a time step. So, the “now” part of the right argument is immaterial, as it cannot delay its initial action until the next time slice.

Definition 5.2.1.3 (Semantics of PA_{drt}^- -ID)

The semantics of PA_{drt}^- -ID are given by the term-deduction system $T(PA_{\text{drt}}^-$ -ID) induced by the deduction rules for BPA_{drt}^- -ID given in Definition 3.2.2.4 on page 48 and the deduction rules for the free merge shown in Table 5.3.

$$\begin{array}{ccc}
 \frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} & \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'} & \frac{x \xrightarrow{a} x'}{x \perp\!\!\!\perp y \xrightarrow{a} x' \perp\!\!\!\perp y} \\
 \\
 \frac{x \xrightarrow{a} \surd}{x \parallel y \xrightarrow{a} y} & \frac{y \xrightarrow{a} \surd}{x \parallel y \xrightarrow{a} x} & \frac{x \xrightarrow{a} \surd}{x \perp\!\!\!\perp y \xrightarrow{a} y} \\
 \\
 \frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \parallel y \xrightarrow{\sigma} x' \parallel y'} & \frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \perp\!\!\!\perp y \xrightarrow{\sigma} x' \perp\!\!\!\perp y'} &
 \end{array}$$

Table 5.3: Deduction rules for free merge.

☞ The deduction rules that do not involve time steps are the same as in the untimed case (see Table 2.13 on page 23). The two extra rules express that the left merge and the free merge can do a time step if and only if both their arguments can. After having done a time step, they continue as the left merge or free merge respectively of the parts that remain of their arguments after they themselves have done a time step.

Definition 5.2.1.4 (Bisimulation and Bisimulation Model for PA_{drt}^- -ID)

Bisimulation for PA_{drt}^- -ID and the corresponding bisimulation model are defined in the same way as for BPA_{drt}^- - δ and BPA respectively. Replace “ BPA_{drt}^- - δ ” by “ PA_{drt}^- -ID” in Definition 3.2.1.8 on page 45 and “BPA” by “ PA_{drt}^- -ID” in Definition 2.3.1.16 on page 12.

Definition 5.2.1.5 (Basic Terms of PA_{drt}^- -ID)

If we speak of basic terms in the context of PA_{drt}^- -ID, we mean $(\sigma, \underline{\delta})$ -basic terms as defined in Definition 3.2.2.6 on page 49.

Definition 5.2.1.6 (Number of Symbols of a PA_{drt}^- -ID Term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). For $a \in A_\delta$, we define $n(\underline{a}) = 1$,
- (ii). for closed PA_{drt}^- -ID terms x and y , we define $n(x + y) = n(x \cdot y) = n(x \parallel y) = n(x \perp\!\!\!\perp y) = n(x) + n(y) + 1$,
- (iii). for a closed PA_{drt}^- -ID term x , we define $n(\sigma(x)) = n(\nu(x)) = n(x) + 1$.

Theorem 5.2.1.7 (Elimination for PA_{drt}^- -ID)

Let t be a closed PA_{drt}^- -ID term. Then there is a closed BPA_{drt}^- -ID term s such that PA_{drt}^- -ID $\vdash s = t$.

Proof We use the direct method we described in Proof Outline 4.2.1.2 on page 68. Let t be a closed PA_{drt}^- -ID term. The theorem is proven by induction on $n(t)$ and case distinction on the general structure of t .

- (i). $t \equiv \underline{a}$ for some $a \in A_\delta$. Then t is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
- (ii). $t \equiv t_1 + t_2$ for closed PA_{drt}^- -ID terms t_1 and t_2 . By induction there are closed $\text{BPA}_{\text{drt}}^-$ -ID terms s_1 and s_2 such that $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 = s_1$ and $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_2 = s_2$. But then also $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 + t_2 = s_1 + s_2$ and $s_1 + s_2$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
- (iii). $t \equiv t_1 \cdot t_2$ for closed PA_{drt}^- -ID terms t_1 and t_2 . This case is treated analogously to case (ii).
- (iv). $t \equiv \sigma(t_1)$ for a closed PA_{drt}^- -ID term t_1 . This case is treated analogously to case (ii).
- (v). $t \equiv \nu(t_1)$ for a closed PA_{drt}^- -ID term t_1 . This case is treated analogously to case (ii).
- (vi). $t \equiv t_1 \parallel t_2$ for closed PA_{drt}^- -ID terms t_1 and t_2 . By induction there are closed $\text{BPA}_{\text{drt}}^-$ -ID terms s_1 and s_2 such that $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 = s_1$ and $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_2 = s_2$. By Theorem 4.3.2.1, the elimination theorem for $\text{BPA}_{\text{drt}}^-$ -ID, there are basic terms r_1 and r_2 such that $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s_1 = r_1$ and $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s_2 = r_2$. But then also, $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 = r_1$, $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_2 = r_2$, and $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2$. We prove this case by induction on the structure of basic term r_1 :
 - (a) $r_1 \equiv \underline{a}$ for some $a \in A_\delta$. Then $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{a} \parallel r_2 = \underline{a} \cdot r_2$, and $\underline{a} \cdot r_2$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
 - (b) $r_1 \equiv \underline{a} \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Then $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{a} \cdot r'_1 \parallel r_2 = \underline{a} \cdot (r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed $\text{BPA}_{\text{drt}}^-$ -ID term p such that $\text{PA}_{\text{drt}}^- \text{-ID} \vdash r'_1 \parallel r_2 = p$. Then, $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) = \underline{a} \cdot p$, and $\underline{a} \cdot p$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
 - (c) $r_1 \equiv r'_1 + r''_1$ for basic terms r'_1 and r''_1 . Then $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = (r'_1 + r''_1) \parallel r_2 = r'_1 \parallel r_2 + r''_1 \parallel r_2$. By induction there exist closed $\text{BPA}_{\text{drt}}^-$ -ID terms p_1 and p_2 such that $\text{PA}_{\text{drt}}^- \text{-ID} \vdash r'_1 \parallel r_2 = p_1$ and $\text{PA}_{\text{drt}}^- \text{-ID} \vdash r''_1 \parallel r_2 = p_2$. Then also $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r'_1 \parallel r_2 + r''_1 \parallel r_2 = p_1 + p_2$, and $p_1 + p_2$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
 - (d) $r_1 \equiv \sigma(r'_1)$ for a basic term r'_1 . By Lemma 3.2.2.10 there is a basic term r'_2 such that either $\text{PA}_{\text{drt}}^- \text{-ID} \vdash r_2 = \nu(r_2)$ or $\text{PA}_{\text{drt}}^- \text{-ID} \vdash r_2 = \nu(r_2) + \sigma(r'_2)$ with $n(r'_2) < n(r_2)$. With case analysis we obtain:
 - 1. $r_2 = \nu(r_2)$. Then we have $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel r_2 = \sigma(r'_1) \parallel \nu(r_2) = \underline{\delta}$, and $\underline{\delta}$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
 - 2. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 . Then $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel r_2 = \sigma(r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) = \sigma(r'_1 \parallel r'_2)$. By the induction hypothesis there is a closed $\text{BPA}_{\text{drt}}^-$ -ID term p such that $\text{PA}_{\text{drt}}^- \text{-ID} \vdash r'_1 \parallel r'_2 = p$. But then also $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = \sigma(r'_1 \parallel r'_2) = \sigma(p)$, and $\sigma(p)$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
- (vii). $t \equiv t_1 \parallel t_2$ for closed PA_{drt}^- -ID terms t_1 and t_2 . Then $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1$. By (vi) there are closed $\text{BPA}_{\text{drt}}^-$ -ID terms p_1 and p_2 such that $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = p_1$ and $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_2 \parallel t_1 = p_2$. But then also $\text{PA}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1 = p_1 + p_2$, and $p_1 + p_2$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.

■

☞ The lexicographical path ordering method is not very suitable here, as Axioms DRTM5 and DRTM6 do not fit well onto basic terms. If we would define a term-rewriting system using these axioms, it would not have basic terms as normal forms. Would we try to fix that by introducing additional rewriting rules, then we would easily lose strong termination.

As we will see later on, it is still possible to find a term-rewriting system for PA_{drt}^- -ID that is both strongly terminating and has basic terms as normal forms. To prove that this is indeed so, however, is more difficult than proving elimination directly as done above, so in this case we have preferred the direct method for proving elimination over the lexicographical path ordering method.

Corollary 5.2.1.8 (Elimination for PA_{drt}^- -ID)

Let t be a closed PA_{drt}^- -ID term. Then there is a basic term s such that PA_{drt}^- -ID $\vdash s = t$.

Proof This follows immediately from:

- (i). The elimination theorem for PA_{drt}^- -ID (see Theorem 5.2.1.7),
- (ii). the elimination theorem for BPA_{drt}^- -ID (see Theorem 4.3.2.1),
- (iii). the fact that all axioms of BPA_{drt}^- -ID are also contained in PA_{drt}^- -ID.

■

Remark 5.2.1.9 (Elimination for PA_{drt}^- -ID)

Elimination for PA_{drt}^- -ID is also claimed (without proof) in Theorem 3.2 of BAETEN AND RENIERS [35].

Theorem 5.2.1.10 (Soundness of PA_{drt}^- -ID)

The set of closed PA_{drt}^- -ID terms modulo bisimulation equivalence is a model of PA_{drt}^- -ID.

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. We only treat the axioms that are new for PA_{drt}^- -ID.

Axiom DRTM1 Take the relation:

$$R = \{(s, s), (s \parallel t, t \parallel s), (s \parallel t, s \ll t + t \ll s) \mid s, t \in C(PA_{\text{drt}}^- \text{-ID})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose that $s \parallel t \xrightarrow{a} p$. First we look at the $(s \parallel t, t \parallel s)$ pairs. By inspection of the deduction rules we can conclude that either $s \xrightarrow{a} p_1$ and $p \equiv p_1 \parallel t$, or $t \xrightarrow{a} p_2$ and $p \equiv s \parallel p_2$, or $s \xrightarrow{a} \surd$ and $p \equiv t$, or $t \xrightarrow{a} \surd$ and $p \equiv s$. Therefore, either $t \parallel s \xrightarrow{a} t \parallel p_1$, or $t \parallel s \xrightarrow{a} p_2 \parallel s$, or $t \parallel s \xrightarrow{a} t$, or $t \parallel s \xrightarrow{a} s$ respectively, and note that $(p_1 \parallel t, t \parallel p_1) \in R$, $(s \parallel p_2, p_2 \parallel s) \in R$, $(t, t) \in R$, and $(s, s) \in R$. Continuing with the $(s \parallel t, s \ll t + t \ll s)$ pairs, we also have either $s \ll t \xrightarrow{a} p_1 \parallel t$, or $t \ll s \xrightarrow{a} p_2 \parallel s$, or $s \ll t \xrightarrow{a} t$, or $t \ll s \xrightarrow{a} s$. Therefore, either $s \ll t + t \ll s \xrightarrow{a} p_1 \parallel t$, or $s \ll t + t \ll s \xrightarrow{a} p_2 \parallel s$, or $s \ll t + t \ll s \xrightarrow{a} t$, or $s \ll t + t \ll s \xrightarrow{a} s$ respectively, and again note that $(p_1 \parallel t, p_1 \parallel t) \in R$, $(s \parallel p_2, p_2 \parallel s) \in R$, $(t, t) \in R$, and $(s, s) \in R$.

- (ii). Suppose that $s \parallel t \xrightarrow{a} \surd$. This case cannot occur.
- (iii). Suppose that $s \parallel t \xrightarrow{\sigma} p$. First we look at the $(s \parallel t, t \parallel s)$ pairs. By inspection of the deduction rules we can conclude that $s \xrightarrow{\sigma} p_1$, $t \xrightarrow{\sigma} p_2$, and $p \equiv p_1 \parallel p_2$. Therefore, $t \parallel s \xrightarrow{\sigma} p_2 \parallel p_1$, and note that $(p_1 \parallel p_2, p_2 \parallel p_1) \in R$.
Continuing with the $(s \parallel t, s \parallel\!\!\! \perp t + t \parallel\!\!\! \perp s)$ pairs, we also have $s \parallel\!\!\! \perp t \xrightarrow{\sigma} p_1 \parallel\!\!\! \perp p_2$ and $t \parallel\!\!\! \perp s \xrightarrow{\sigma} p_2 \parallel\!\!\! \perp p_1$. Therefore, $s \parallel\!\!\! \perp t + t \parallel\!\!\! \perp s \xrightarrow{\sigma} p_1 \parallel\!\!\! \perp p_2 + p_2 \parallel\!\!\! \perp p_1$, and note that $(p_1 \parallel\!\!\! \perp p_2, p_1 \parallel\!\!\! \perp p_2 + p_2 \parallel\!\!\! \perp p_1) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose that $t \parallel s \xrightarrow{a} p$. This case is handled in the same way as the corresponding (sub)case for the left-hand side shown above.
- (ii). Suppose that $t \parallel s \xrightarrow{a} \surd$. This case cannot occur.
- (iii). Suppose that $t \parallel s \xrightarrow{\sigma} p$. This case is handled in the same way as the corresponding (sub)case for the left-hand side shown above.
- (iv). Suppose that $s \parallel\!\!\! \perp t + t \parallel\!\!\! \perp s \xrightarrow{a} p$. By inspection of the deduction rules we can conclude that either $s \xrightarrow{a} p_1$ and $p \equiv p_1 \parallel t$, or $t \xrightarrow{a} p_2$ and $p \equiv p_2 \parallel s$, or $s \xrightarrow{a} \surd$ and $p \equiv t$, or $t \xrightarrow{a} \surd$ and $p \equiv s$. Therefore, either $s \parallel t \xrightarrow{a} p_1 \parallel t$, or $s \parallel t \xrightarrow{a} s \parallel p_2$, or $s \parallel t \xrightarrow{a} t$, or $s \parallel t \xrightarrow{a} s$ respectively, and note that $(p_1 \parallel t, p_1 \parallel t) \in R$, $(s \parallel p_2, p_2 \parallel s) \in R$, $(t, t) \in R$, and $(s, s) \in R$.
- (v). Suppose that $s \parallel\!\!\! \perp t + t \parallel\!\!\! \perp s \xrightarrow{a} \surd$. This case cannot occur.
- (vi). Suppose that $s \parallel\!\!\! \perp t + t \parallel\!\!\! \perp s \xrightarrow{\sigma} p$. By inspection of the deduction rules we can conclude that $s \xrightarrow{\sigma} p_1$, $t \xrightarrow{\sigma} p_2$, and $p \equiv p_1 \parallel\!\!\! \perp p_2 + p_2 \parallel\!\!\! \perp p_1$. Since both s and t can perform a σ transition, we obtain $s \parallel t \xrightarrow{\sigma} p_1 \parallel p_2$, and note that $(p_1 \parallel\!\!\! \perp p_2, p_1 \parallel\!\!\! \perp p_2 + p_2 \parallel\!\!\! \perp p_1) \in R$.

Axiom DRTM2 Take the relation:

$$R = \{(s, s), (\underline{a} \parallel\!\!\! \perp s, \underline{a} \cdot s) \mid s \in C(\text{PA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. The only possible transition of the left-hand side is $\underline{a} \parallel\!\!\! \perp s \xrightarrow{a} s$, the only possible transition of the right-hand side is $\underline{a} \cdot s \xrightarrow{a} s$, and note that $(s, s) \in R$.

Axiom DRTM3 Take the relation:

$$R = \{(s, s), (\underline{a} \cdot s \parallel\!\!\! \perp t, \underline{a} \cdot (s \parallel t)) \mid s, t \in C(\text{PA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. The only possible transition of the left-hand side is $\underline{a} \cdot s \parallel\!\!\! \perp t \xrightarrow{a} s \parallel t$, the only possible transition of the right-hand side is $\underline{a} \cdot (s \parallel t) \xrightarrow{a} s \parallel t$, and note that $(s \parallel t, s \parallel t) \in R$.

Axiom DRTM4 Take the relation:

$$R = \{(s, s), ((s + t) \parallel\!\!\! \perp u, s \parallel\!\!\! \perp u + t \parallel\!\!\! \perp u) \mid s, t, u \in C(\text{PA}_{\text{drt}}^- \text{-ID})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose $(s+t) \Downarrow u \xrightarrow{a} p$. By inspection of the deduction rules we can conclude that either $s \xrightarrow{a} p_1$ and $p \equiv p_1 \parallel u$, or $t \xrightarrow{a} p_2$ and $p \equiv p_2 \parallel u$, or $s \xrightarrow{a} \surd$ and $p \equiv u$, or $t \xrightarrow{a} \surd$ and $p \equiv u$. So, either $s \Downarrow u \xrightarrow{a} p_1 \parallel u$, or $t \Downarrow u \xrightarrow{a} p_2 \parallel u$, or $s \Downarrow u \xrightarrow{a} u$, or $t \Downarrow u \xrightarrow{a} u$ respectively. Therefore, either $s \Downarrow u+t \Downarrow u \xrightarrow{a} p_1 \parallel u$, or $s \Downarrow u+t \Downarrow u \xrightarrow{a} p_2 \parallel u$, or $s \Downarrow u+t \Downarrow u \xrightarrow{a} u$, or $s \Downarrow u+t \Downarrow u \xrightarrow{a} u$ respectively, and note that $(p_1 \parallel u, p_1 \parallel u) \in R$, $(p_2 \parallel u, p_2 \parallel u) \in R$, $(u, u) \in R$, and $(u, u) \in R$.
- (ii). Suppose $(s+t) \Downarrow u \xrightarrow{a} \surd$. This case cannot occur.
- (iii). Suppose $(s+t) \Downarrow u \xrightarrow{\sigma} p$. Then $s+t \xrightarrow{\sigma} p_1$, $u \xrightarrow{\sigma} p_2$, and $p \equiv p_1 \parallel p_2$. Then one of the following situations has occurred:
- (a) $s \xrightarrow{\sigma} p_1$ and $t \xrightarrow{\sigma} \surd$: then $s \Downarrow u \xrightarrow{\sigma} p_1 \parallel p_2$ and $t \Downarrow u \xrightarrow{\sigma} \surd$. Therefore, $s \Downarrow u+t \Downarrow u \xrightarrow{\sigma} p_1 \parallel p_2$, and note that $(p_1 \parallel p_2, p_1 \parallel p_2) \in R$.
- (b) $s \xrightarrow{\sigma} \surd$ and $t \xrightarrow{\sigma} p_1$: this case is handled in the same way as the previous one.
- (c) $s \xrightarrow{\sigma} q_1$, $t \xrightarrow{\sigma} q_2$, and $p_1 \equiv q_1 + q_2$: then $s \Downarrow u \xrightarrow{\sigma} q_1 \parallel p_2$ and $t \Downarrow u \xrightarrow{\sigma} q_2 \parallel p_2$. Therefore $s \Downarrow u+t \Downarrow u \xrightarrow{\sigma} q_1 \parallel p_2 + q_2 \parallel p_2$, and note that $((q_1 + q_2) \parallel p_2, q_1 \parallel p_2 + q_2 \parallel p_2) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose $s \Downarrow u+t \Downarrow u \xrightarrow{a} p$. By inspection of the deduction rules we can conclude that either $s \xrightarrow{a} p_1$ and $p \equiv p_1 \parallel u$, or $t \xrightarrow{a} p_2$ and $p \equiv p_2 \parallel u$, or $s \xrightarrow{a} \surd$ and $p \equiv u$, or $t \xrightarrow{a} \surd$ and $p \equiv u$. Therefore, either $(s+t) \Downarrow u \xrightarrow{a} p_1 \parallel u$, or $(s+t) \Downarrow u \xrightarrow{a} p_2 \parallel u$, or $(s+t) \Downarrow u \xrightarrow{a} u$, or $(s+t) \Downarrow u \xrightarrow{a} u$, and note that $(p_1 \parallel u, p_1 \parallel u) \in R$, $(p_2 \parallel u, p_2 \parallel u) \in R$, $(u, u) \in R$, and $(u, u) \in R$.
- (ii). Suppose $s \Downarrow u+t \Downarrow u \xrightarrow{a} \surd$. This case cannot occur.
- (iii). Suppose $s \Downarrow u+t \Downarrow u \xrightarrow{\sigma} p$. Then this must be due to one of the following:
- (a) $s \Downarrow u \xrightarrow{\sigma} p$ and $t \Downarrow u \xrightarrow{\sigma} \surd$: then $s \xrightarrow{\sigma} q_1$, $u \xrightarrow{\sigma} q_2$, and $p \equiv q_1 \parallel q_2$. Therefore, $s+t \xrightarrow{\sigma} q_1$ and $(s+t) \Downarrow u \xrightarrow{\sigma} q_1 \parallel q_2$, and note that $(q_1 \parallel q_2, q_1 \parallel q_2) \in R$.
- (b) $s \Downarrow u \xrightarrow{\sigma} \surd$ and $t \Downarrow u \xrightarrow{\sigma} p$: this case is handled in the same way as the previous one.
- (c) $s \Downarrow u \xrightarrow{\sigma} p_1$, $t \Downarrow u \xrightarrow{\sigma} p_2$, and $p \equiv p_1 + p_2$: then $s \xrightarrow{\sigma} q_1$, $t \xrightarrow{\sigma} q_2$, $u \xrightarrow{\sigma} q_3$, $p_1 \equiv q_1 \parallel q_3$, and $p_2 \equiv q_2 \parallel q_3$. Therefore, $s+t \xrightarrow{\sigma} q_1 + q_2$ and $(s+t) \Downarrow u \xrightarrow{\sigma} (q_1 + q_2) \parallel q_3$, and note that $((q_1 + q_2) \parallel q_3, q_1 \parallel q_3 + q_2 \parallel q_3) \in R$.

Axiom DRTM5 Take the relation:

$$R = \{(\sigma(s) \Downarrow v(t), \underline{\delta}) \mid s, t \in C(\text{PA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $\sigma(s) \Downarrow v(t) \nrightarrow$. Also for the right-hand side there are no transitions possible: $\underline{\delta} \nrightarrow$.

Axiom DRTM6 Take the relation:

$$R = \{(s, s), (\sigma(s) \Downarrow (v(t) + \sigma(u)), \sigma(s \Downarrow u)) \mid s, t, u \in C(\text{PA}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that the only possible transition of the left-hand side is $\sigma(s) \Downarrow (v(t) + \sigma(u)) \xrightarrow{\sigma} s \Downarrow u$, and that the only possible transition of the right-hand side is $\sigma(s \Downarrow u) \xrightarrow{\sigma} s \Downarrow u$, and note that $(s \Downarrow u, s \Downarrow u) \in R$.

■

Remark 5.2.1.11 (Soundness of PA_{drt}^- -ID)

Soundness of PA_{drt}^- -ID is also claimed (without proof) in Theorem 3.3 of BAETEN AND RENIERS [35].

Remark 5.2.1.12 (Conservativity etc.)

In Theorems 5.2.1.13 and 5.2.1.14 below we use the concepts *operationally conservative extension*, *path format*, *sum of term-deduction systems* (denoted by \oplus), *pure deduction rules*, and *well-founded deduction rules*. For a formal definition of these concepts and the intuitions behind them, see Section 2.4.1 of BAETEN AND VERHOEF [37], and the references given there.

Theorem 5.2.1.13 (Conservativity of PA_{drt}^- -ID with respect to BPA_{drt}^- -ID)

The process algebra PA_{drt}^- -ID is a conservative extension of the process algebra BPA_{drt}^- -ID.

Proof In order to prove conservativity it is sufficient to verify that the following conditions are satisfied:

- (i). Bisimulation equivalence is definable in terms of predicate and relation symbols only,
- (ii). BPA_{drt}^- -ID is a complete axiomatization with respect to the bisimulation equivalence model induced by $T(BPA_{\text{drt}}^-$ -ID) (see Theorem 4.3.2.6),
- (iii). PA_{drt}^- -ID is a sound axiomatization with respect to the bisimulation equivalence model induced by $T(PA_{\text{drt}}^-$ -ID) (see Theorem 5.2.1.10),
- (iv). $T(PA_{\text{drt}}^-$ -ID) is an operationally conservative extension of $T(BPA_{\text{drt}}^-$ -ID).

And in order for $T(PA_{\text{drt}}^-$ -ID) indeed to be an operationally conservative extension of $T(BPA_{\text{drt}}^-$ -ID) we must verify the following conditions:

- (i). $T(BPA_{\text{drt}}^-$ -ID) is a pure, well-founded term-deduction system in path format,
- (ii). $T(PA_{\text{drt}}^-$ -ID) is a term-deduction system in path format,
- (iii). $T(BPA_{\text{drt}}^-$ -ID) \oplus $T(PA_{\text{drt}}^-$ -ID) is defined.

That the above properties hold can be trivially checked from the relevant definitions. ■

Theorem 5.2.1.14 (Completeness of PA_{drt}^- -ID)

The axioms system PA_{drt}^- -ID is a complete axiomatization of the set of closed PA_{drt}^- -ID terms modulo bisimulation equivalence.

Proof We use Verhoef's method described in Proof Outline 4.2.3.4 on page 71. Completeness then follows immediately from:

- (i). PA_{drt}^- -ID has the elimination property for BPA_{drt}^- -ID (see Theorem 5.2.1.7),
- (ii). PA_{drt}^- -ID is a conservative extension of BPA_{drt}^- -ID (see Theorem 5.2.1.13).

■

Remark 5.2.1.15 (Completeness of PA_{drt}^- -ID)

Completeness of PA_{drt}^- -ID is also claimed (without proof) in Theorem 3.3 of BAETEN AND RENIERS [35].

5.2.2 PA_{drt}^- -ID'

In this section, we extend $\text{BPA}_{\text{drt}}^-$ -ID to PA_{drt}^- -ID' by introducing axioms for the free merge that are based on an inductive definition of basic terms. As we will show, for closed terms, all derivable equalities of PA_{drt}^- -ID' are also derivable in PA_{drt}^- -ID, and vice versa.

Definition 5.2.2.1 (Signature of PA_{drt}^- -ID')

The signature of PA_{drt}^- -ID' is identical to the signature of PA_{drt}^- -ID as given in Definition 5.2.1.1; it consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , the “now” operator ν_{rel} , the *free merge operator* \parallel , and the *left merge operator* \ll .

Definition 5.2.2.2 (Axioms of PA_{drt}^- -ID')

The process algebra PA_{drt}^- -ID' is axiomatized by the axioms of $\text{BPA}_{\text{drt}}^-$ -ID given in Definition 3.2.2.2 on page 47, Axioms DRTM1-DRTM4 shown in Table 5.1 on page 106, and Axioms DRTM7-DRTM11 shown in Table 5.4: PA_{drt}^- -ID' = A1-A5 + DRT1-DRT5 + DCS1-DCS4 + DRTM1-DRTM4 + DRTM7-DRTM11.

$\sigma_{\text{rel}}(x) \ll \underline{a} = \underline{\delta}$	DRTM7
$\sigma_{\text{rel}}(x) \ll \underline{a} \cdot y = \underline{\delta}$	DRTM8
$\sigma_{\text{rel}}(x) \ll (\underline{a} + y) = \sigma_{\text{rel}}(x) \ll y$	DRTM9
$\sigma_{\text{rel}}(x) \ll (\underline{a} \cdot y + z) = \sigma_{\text{rel}}(x) \ll z$	DRTM10
$\sigma_{\text{rel}}(x) \ll \sigma_{\text{rel}}(y) = \sigma_{\text{rel}}(x \ll y)$	DRTM11

Table 5.4: Additional axioms for PA_{drt}^- -ID'.

☞ The axioms of PA_{drt}^- -ID' differ from those of PA_{drt}^- -ID only in the axiomatization of the left merge with respect to time, i.e., in the axioms for $\sigma(x) \ll y$. Instead of splitting y in a “now” and a “next time slice” part, as we have did for PA_{drt}^- -ID in Table 5.2 on page 106, we here inductively cover all forms that the y might take.

To begin with, we assume that the right argument of the left merge is a basic term, since we have the elimination property for $\text{BPA}_{\text{drt}}^-$ -ID, the Basic Process Algebra that underlies PA_{drt}^- -ID'. Now, Axioms DRTM9 and DRTM10 strip away as many summands of the form \underline{a} or $\underline{a} \cdot y$ as possible from the right argument. This is motivated by the fact that since the left argument of the left merge is $\sigma(x)$, an \underline{a} or $\underline{a} \cdot y$ summand on the right side can never execute anyway. Then, if we started with one or more σ -summands, these are exactly the summands that will remain, and using time-factorization and Axiom DRTM11, we can rewrite the left merge into a left merge in the next time slice. If we started with no σ -summands at all, i.e., with only \underline{a} and $\underline{a} \cdot y$ summands, then we will end

up with exactly one such summand. Then, by Axioms DRTM7 and DRTM8, we can rewrite the left merge into $\underline{\delta}$, as apparently there is no way for the right argument to move along with the left argument into the next time slice.

In this way, $\sigma(x) \underline{\parallel} y$ is defined for all possible cases. Note that these intuitions are only intuitions: no formal claims are made. We will prove soundness and completeness results for $PA_{\text{drt}}^- \text{-ID}'$ in the remaining part of this section, without using the above intuitions at all.

Definition 5.2.2.3 (Semantics of $PA_{\text{drt}}^- \text{-ID}'$)

The semantics of $PA_{\text{drt}}^- \text{-ID}'$ are given by the term-deduction system $T(PA_{\text{drt}}^- \text{-ID}')$ which is identical to the term-deduction system $T(PA_{\text{drt}}^- \text{-ID})$ given in Definition 5.2.1.3 on page 107.

Definition 5.2.2.4 (Bisimulation and Bisimulation Model for $PA_{\text{drt}}^- \text{-ID}'$)

Bisimulation for $PA_{\text{drt}}^- \text{-ID}'$ and the corresponding bisimulation model are defined in the same way as for $BPA_{\text{drt}}^- \text{-}\delta$ and BPA respectively. Replace “ $BPA_{\text{drt}}^- \text{-}\delta$ ” by “ $PA_{\text{drt}}^- \text{-ID}'$ ” in Definition 3.2.1.8 on page 45 and “BPA” by “ $PA_{\text{drt}}^- \text{-ID}'$ ” in Definition 2.3.1.16 on page 12.

Definition 5.2.2.5 (Basic Terms of $PA_{\text{drt}}^- \text{-ID}'$)

If we speak of basic terms in the context of $PA_{\text{drt}}^- \text{-ID}'$, we mean $(\sigma, \underline{\delta})$ -basic terms as defined in Definition 3.2.2.6 on page 49.

Definition 5.2.2.6 (Number of Symbols of a $PA_{\text{drt}}^- \text{-ID}'$ Term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). For $a \in A_\delta$, we define $n(\underline{a}) = 1$,
- (ii). for closed $PA_{\text{drt}}^- \text{-ID}'$ terms x and y , we define $n(x + y) = n(x \cdot y) = n(x \parallel y) = n(x \underline{\parallel} y) = n(x) + n(y) + 1$,
- (iii). for a closed $PA_{\text{drt}}^- \text{-ID}'$ term x , we define $n(\sigma(x)) = n(\nu(x)) = n(x) + 1$.

Theorem 5.2.2.7 (Elimination for $PA_{\text{drt}}^- \text{-ID}'$)

Let t be a closed $PA_{\text{drt}}^- \text{-ID}'$ term. Then there is a closed $BPA_{\text{drt}}^- \text{-ID}$ term s such that $PA_{\text{drt}}^- \text{-ID}' \vdash s = t$.

Proof We use the lexicographical path ordering method we described in Proof Outline 4.2.1.1 on page 68. The term-rewriting system used consists of the term-rewriting system for $BPA_{\text{drt}}^- \text{-ID}$ shown in Table 4.2 on page 79, augmented with the additional term-rewriting rules shown in Table 5.5 on the facing page. Note that we have added natural number subscripts n to the left merge and free merge operators, in order to deal with the mutually recursive nature of definition of these operators. For a description of this technique, and a rigorous formal justification of its use, see Theorem 3.2.3 of BAETEN AND VERHOEF [37], and the references given there. The operator \cdot is assigned the lexicographical status of the first argument, and the following well-founded ordering on constant and function symbols is used:

$$\underline{a} < \sigma < + < \cdot < \parallel_2 < \parallel_2 < \parallel_3 < \dots < \parallel_n < \parallel_n < \parallel_{n+1} < \dots$$

$\sigma(x) + \sigma(y) \rightarrow \sigma(x + y)$	RDRT1
$x \parallel_n y \rightarrow x \parallel_n y + y \parallel_n x$	RDRTM1
$\underline{a} \parallel_n x \rightarrow \underline{a} \cdot x$	RDRTM2
$\underline{a} \cdot x \parallel_{n+1} y \rightarrow \underline{a} \cdot (x \parallel_n y)$	RDRTM3
$(x + y) \parallel_n z \rightarrow x \parallel_n z + y \parallel_n z$	RDRTM4
$\sigma(x) \parallel_n \underline{a} \rightarrow \underline{\delta}$	RDRTM7
$\sigma(x) \parallel_n \underline{a} \cdot y \rightarrow \underline{\delta}$	RDRTM8
$\sigma(x) \parallel_n (\underline{a} + y) \rightarrow \sigma(x) \parallel_n y$	RDRTM9
$\sigma(x) \parallel_n (\underline{a} \cdot y + z) \rightarrow \sigma(x) \parallel_n z$	RDRTM10
$\sigma(x) \parallel_n \sigma(y) \rightarrow \sigma(x \parallel_n y)$	RDRTM11

Table 5.5: Additional term-rewriting rules for $\text{PA}_{\text{drt}}^- \text{-ID}'$.

That the left-hand side of every rewriting rule is bigger than the right-hand side with respect to the ordering \succ_{lpo} , is shown by the following reductions:

$$\begin{aligned}
& \sigma(x) + \sigma(y) \succ_{\text{lpo}} \sigma(x) + \sigma(y) \succ_{\text{lpo}} \sigma(\sigma(x) + \sigma(y)) \succ_{\text{lpo}} \sigma(\sigma^*(x) + \sigma^*(y)) \\
& \succ_{\text{lpo}} \sigma(x + y) \\
& x \parallel_n y \succ_{\text{lpo}} x \parallel_n^* y \succ_{\text{lpo}} x \parallel_n^* y + x \parallel_n^* y \\
& \succ_{\text{lpo}} (x \parallel_n^* y) \parallel_n (x \parallel_n^* y) + (x \parallel_n^* y) \parallel_n (x \parallel_n^* y) \\
& \succ_{\text{lpo}} x \parallel_n y + y \parallel_n x \\
& \underline{a} \parallel_n x \succ_{\text{lpo}} \underline{a} \parallel_n^* x \succ_{\text{lpo}} (\underline{a} \parallel_n^* x) \cdot (\underline{a} \parallel_n^* x) \\
& \succ_{\text{lpo}} \underline{a} \cdot x \\
& \underline{a} \cdot x \parallel_{n+1} y \succ_{\text{lpo}} \underline{a} \cdot x \parallel_{n+1}^* y \succ_{\text{lpo}} (\underline{a} \cdot x \parallel_{n+1}^* y) \cdot (\underline{a} \cdot x \parallel_{n+1}^* y) \\
& \succ_{\text{lpo}} (\underline{a} \cdot x) \cdot ((\underline{a} \cdot x \parallel_{n+1}^* y) \parallel_n (\underline{a} \cdot x \parallel_{n+1}^* y)) \\
& \succ_{\text{lpo}} (\underline{a} \cdot x) \cdot (\underline{a} \cdot x \parallel_n y) \succ_{\text{lpo}} (\underline{a} \cdot^* x) \cdot (\underline{a} \cdot^* x \parallel_n y) \\
& \succ_{\text{lpo}} \underline{a} \cdot (x \parallel_n y) \\
& (x + y) \parallel_n z \succ_{\text{lpo}} (x + y) \parallel_n^* z \succ_{\text{lpo}} (x + y) \parallel_n^* z + (x + y) \parallel_n^* z \\
& \succ_{\text{lpo}} (x +^* y) \parallel_n z + (x +^* y) \parallel_n z \succ_{\text{lpo}} (x \parallel_n z) + (y \parallel_n z) \\
& \sigma(x) \parallel_n \underline{a} \succ_{\text{lpo}} \sigma(x) \parallel_n^* \underline{a} \\
& \succ_{\text{lpo}} \underline{\delta} \\
& \sigma(x) \parallel_n \underline{a} \cdot y \succ_{\text{lpo}} \sigma(x) \parallel_n^* \underline{a} \cdot y \\
& \succ_{\text{lpo}} \underline{\delta} \\
& \sigma(x) \parallel_n (\underline{a} + y) \succ_{\text{lpo}} \sigma(x) \parallel_n^* (\underline{a} + y) \succ_{\text{lpo}} \sigma(x) \parallel_n (\underline{a} +^* y) \\
& \succ_{\text{lpo}} \sigma(x) \parallel_n y \\
& \sigma(x) \parallel_n (\underline{a} \cdot y + z) \succ_{\text{lpo}} \sigma(x) \parallel_n^* (\underline{a} \cdot y + z) \succ_{\text{lpo}} \sigma(x) \parallel_n (\underline{a} \cdot y +^* z) \\
& \succ_{\text{lpo}} \sigma(x) \parallel_n z
\end{aligned}$$

$$\begin{aligned} \sigma(x) \ll_n \sigma(y) >_{\text{lpo}} \sigma(x) \ll_n^* \sigma(y) >_{\text{lpo}} \sigma(\sigma(x) \ll_n^* \sigma(y)) \\ >_{\text{lpo}} \sigma(\sigma^*(x) \ll_n \sigma^*(y)) >_{\text{lpo}} \sigma(x \ll_n y) \end{aligned}$$

It remains to prove that every normal form of a closed $\text{PA}_{\text{drt}}^- \text{-ID}'$ term is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term. We prove this as follows: suppose that s is a normal form of a closed $\text{PA}_{\text{drt}}^- \text{-ID}'$ term, and furthermore suppose that s is not a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term. Now consider the smallest subterm s' of s that is not a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term. Then, s' must be of the form $s' \equiv s_1 \parallel s_2$ or of the form $s' \equiv s_1 \ll s_2$, for closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ terms s_1 and s_2 . By the elimination theorem for $\text{BPA}_{\text{drt}}^- \text{-ID}$, Theorem 4.3.2.1 on page 79, we may assume that s_1 and s_2 are basic terms. Now in the first case, $s' \equiv s_1 \parallel s_2$, clearly rewriting rule RDRTM1 is applicable. This contradicts the assumption that s is a normal form, so this case cannot occur. In the second case, $s' \equiv s_1 \ll s_2$, it is not clear at first sight that a contradiction can be derived. So, the following cases have to be considered for basic term s_1 :

- (i). $s_1 \equiv \underline{a}$ for some $a \in A_\delta$. Then rewriting rule RDRTM2 is applicable.
- (ii). $s_1 \equiv \underline{a} \cdot s'_1$ for some $a \in A_\delta$ and closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term s'_1 . Clearly, rewriting rule RDRTM3 is applicable.
- (iii). $s_1 \equiv s'_1 + s''_1$ for some closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ terms s'_1 and s''_1 . This time rewriting rule RDRTM4 is applicable.
- (iv). $s_1 \equiv \sigma(s'_1)$ for some closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term s'_1 . The following cases can be considered for the general form of basic term s_2 :
 - (a) $s_2 \equiv \underline{a}$ for some $a \in A_\delta$. In this case rewriting rule RDRTM7 is applicable.
 - (b) $s_2 \equiv \underline{a} \cdot s'_2$ for some $a \in A$ and basic term s'_2 . In this case rewriting rule RDRTM8 is applicable.
 - (c) $s_2 \equiv \sum_{i < m} \underline{a_i} \cdot s_{2,i} + \sum_{j < n} \underline{b_j} + \sum_{k < p} \sigma(s'_{2,k})$ for $m, n, p \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and $s_{2,i}$ and $s'_{2,k}$ basic terms. In this case at least one of the rewriting rules RDRT1 , RDRTM9 , RDRTM10 , or RDRTM11 is applicable.

In every case a rewriting rule is applicable. Therefore s' is not a normal form.

We see that in both cases s' is not a normal form. But this contradicts the assumption that s is a normal form. From this contradiction we conclude that s does not contain a merge operator. Therefore s must be closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term, which had to be proven. ■

Corollary 5.2.2.8 (Elimination for $\text{PA}_{\text{drt}}^- \text{-ID}'$)

Let t be a closed $\text{PA}_{\text{drt}}^- \text{-ID}'$ term. Then there is a basic term s such that $\text{PA}_{\text{drt}}^- \text{-ID}' \vdash s = t$.

Proof This follows immediately from:

- (i). The elimination theorem for $\text{PA}_{\text{drt}}^- \text{-ID}'$ (see Theorem 5.2.2.7),
- (ii). the elimination theorem for $\text{BPA}_{\text{drt}}^- \text{-ID}$ (see Theorem 4.3.2.1),
- (iii). the fact that all axioms of $\text{BPA}_{\text{drt}}^- \text{-ID}$ are also contained in $\text{PA}_{\text{drt}}^- \text{-ID}'$.

■

Remark 5.2.2.9 (Elimination for $PA_{\text{drt}}^- \text{-ID}'$)

Elimination for a slightly different version of $PA_{\text{drt}}^- \text{-ID}'$ is also claimed (without proof) in Theorem 3.4.2 of BAETEN AND VERHOEF [37] (where $PA_{\text{drt}}^- \text{-ID}'$ is called $PA_{\delta \text{dt}}$).

Definition 5.2.2.10 (Axiom Ground Equivalence of Process Algebras)

Two process algebras P and P' are called *axiom ground equivalent* if they have identical signatures, and the same equalities over closed terms hold in both systems, i.e., for all closed terms s and t over the common signature we have $P \vdash s = t$ iff $P' \vdash s = t$.

Definition 5.2.2.11 (Deduction-System Ground Equivalence of Process Algebras)

Two process algebras P and P' , for which term-deduction systems $T(P)$ and $T(P')$ and corresponding bisimulation models M and M' are given, are called *deduction-system ground equivalent* if they have identical signatures, and the same equalities over closed terms hold in both bisimulation models, i.e., for all closed terms s and t over the common signature we have $M \models s \sim_p t$ iff $M' \models s \sim_{p'} t$.

Definition 5.2.2.12 (Ground Equivalence of Process Algebras)

Two process algebras P and P' that are both *axiom ground equivalent* and *deduction-system ground equivalent* are simply called *ground equivalent*.

Theorem 5.2.2.13 (Axiom Ground Equivalence of $PA_{\text{drt}}^- \text{-ID}$ and $PA_{\text{drt}}^- \text{-ID}'$)

For all closed $PA_{\text{drt}}^- \text{-ID}$ terms s and t we have $PA_{\text{drt}}^- \text{-ID} \vdash s = t$ if and only if $PA_{\text{drt}}^- \text{-ID}' \vdash s = t$.

Proof It suffices to show that, for closed terms, every axiom of $PA_{\text{drt}}^- \text{-ID}'$ is derivable from the axioms of $PA_{\text{drt}}^- \text{-ID}$, and vice versa, that, for closed terms, every axiom of $PA_{\text{drt}}^- \text{-ID}$ is derivable from the axioms of $PA_{\text{drt}}^- \text{-ID}'$. We can restrict ourselves to the axioms that are not shared by both process algebras.

Part I

First, we show that Axioms DRTM7–DRTM11 of $PA_{\text{drt}}^- \text{-ID}'$ are derivable in $PA_{\text{drt}}^- \text{-ID}$:

Axiom DRTM7

$$PA_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll \underline{\underline{a}} = \underline{\underline{\delta}}$$

Consider the following derivation:

$$PA_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll \underline{\underline{a}} = \sigma(x) \ll \nu(\underline{\underline{a}}) = \underline{\underline{\delta}}$$

Axiom DRTM8

$$PA_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll \underline{\underline{a}} \cdot y = \underline{\underline{\delta}}$$

Consider the following derivation:

$$PA_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll \underline{\underline{a}} \cdot y = \sigma(x) \ll \nu(\underline{\underline{a}}) \cdot y = \sigma(x) \ll \nu(\underline{\underline{a}} \cdot y) = \underline{\underline{\delta}}$$

Axiom DRTM9

$$\text{PA}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll (\underline{\underline{a}} + y) = \sigma(x) \ll y$$

By the Lemma 3.2.2.10, there is a basic terms s such that either $\text{PA}_{\text{drt}}^- \text{-ID} \vdash y = \nu(y)$ or $\text{PA}_{\text{drt}}^- \text{-ID} \vdash y = \nu(y) + \sigma(s)$. So, there are two cases to be distinguished.

(i). $\text{PA}_{\text{drt}}^- \text{-ID} \vdash y = \nu(y)$. Then we have:

$$\begin{aligned} \text{PA}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll (\underline{\underline{a}} + y) &= \\ \sigma(x) \ll (\underline{\underline{a}} + \nu(y)) &= \\ \sigma(x) \ll (\nu(\underline{\underline{a}}) + \nu(y)) &= \\ \sigma(x) \ll \nu(\underline{\underline{a}} + y) &= \\ \underline{\underline{\delta}} &= \\ \sigma(x) \ll \nu(y) &= \\ \sigma(x) \ll y & \end{aligned}$$

(ii). $\text{PA}_{\text{drt}}^- \text{-ID} \vdash y = \nu(y) + \sigma(s)$. Then we have:

$$\begin{aligned} \text{PA}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll (\underline{\underline{a}} + y) &= \\ \sigma(x) \ll (\underline{\underline{a}} + \nu(y) + \sigma(s)) &= \\ \sigma(x) \ll (\nu(\underline{\underline{a}}) + \nu(y) + \sigma(s)) &= \\ \sigma(x) \ll (\nu(\underline{\underline{a}} + y) + \sigma(s)) &= \\ \sigma(x) \ll \sigma(s) &= \\ \sigma(x) \ll (\nu(y) + \sigma(s)) &= \\ \sigma(x) \ll y & \end{aligned}$$

Axiom DRTM10

$$\text{PA}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll (\underline{\underline{a}} \cdot y + z) = \sigma(x) \ll z$$

Handled in the same way as the previous case.

Axiom DRTM11

$$\text{PA}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll \sigma(y) = \sigma(x \ll y)$$

Consider the following derivation: $\text{PA}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \ll \sigma(y) = \sigma(x) \ll (\underline{\underline{\delta}} + \sigma(y)) = \sigma(x) \ll (\nu(\underline{\underline{\delta}}) + \sigma(y)) = \sigma(x \ll y)$.

Part II

Secondly, we show that Axioms DRTM5-DRTM6 of $\text{PA}_{\text{drt}}^- \text{-ID}$ are derivable in $\text{PA}_{\text{drt}}^- \text{-ID}'$:

Axiom DRTM5

$$\text{PA}_{\text{drt}}^- \text{-ID}' \vdash \sigma(x) \ll \nu(y) = \underline{\underline{\delta}}$$

Use the general form of basic term y . Take:

$$y \equiv \sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \sigma(u_k)$$

for $m, n, p \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and basic terms t_i and u_k . Then we have:

$$\begin{aligned} \text{PA}_{\text{drt}}^- \text{-ID}' \vdash \\ \sigma(x) \ll \nu(y) &= \\ \sigma(x) \ll \nu \left(\sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \sigma(u_k) \right) &= \\ \sigma(x) \ll \left(\sum_{i < m} \nu(\underline{\underline{a_i}} \cdot t_i) + \sum_{j < n} \nu(\underline{\underline{b_j}}) + \sum_{k < p} \nu(\sigma(u_k)) \right) &= \\ \sigma(x) \ll \left(\sum_{i < m} \nu(\underline{\underline{a_i}}) \cdot t_i + \sum_{j < n} \nu(\underline{\underline{b_j}}) + \sum_{k < p} \nu(\sigma(u_k)) \right) &= \\ \sigma(x) \ll \left(\sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \underline{\underline{\delta}} \right) &= \\ \sigma(x) \ll \left(\sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \underline{\underline{\delta}} \right) &= \\ \sigma(x) \ll \underline{\underline{\delta}} &= \\ \underline{\underline{\delta}} & \end{aligned}$$

Axiom DRTM6

$$\text{PA}_{\text{drt}}^- \text{-ID}' \vdash \sigma(x) \ll (\nu(y) + \sigma(z)) = \sigma(x \ll z)$$

Use the general form of basic term y . Take:

$$y \equiv \sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \sigma(u_k)$$

for $m, n, p \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and basic terms t_i and u_k . Then we have:

$$\begin{aligned} \text{PA}_{\text{drt}}^- \text{-ID}' \vdash \\ \sigma(x) \ll (\nu(y) + \sigma(z)) &= \\ \sigma(x) \ll \left(\nu \left(\sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \sigma(u_k) \right) + \sigma(z) \right) &= \\ \sigma(x) \ll \left(\sum_{i < m} \nu(\underline{\underline{a_i}} \cdot t_i) + \sum_{j < n} \nu(\underline{\underline{b_j}}) + \sum_{k < p} \nu(\sigma(u_k)) + \sigma(z) \right) &= \\ \sigma(x) \ll \left(\sum_{i < m} \nu(\underline{\underline{a_i}}) \cdot t_i + \sum_{j < n} \nu(\underline{\underline{b_j}}) + \sum_{k < p} \nu(\sigma(u_k)) + \sigma(z) \right) &= \end{aligned}$$

$$\begin{aligned}
\sigma(x) &\Downarrow \left(\sum_{i < m} \underline{a_i} \cdot t_i + \sum_{j < n} \underline{b_j} + \sum_{k < p} \underline{\delta} + \sigma(z) \right) = \\
\sigma(x) &\Downarrow \left(\sum_{i < m} \underline{a_i} \cdot t_i + \sum_{j < n} \underline{b_j} + \sigma(z) \right) = \\
\sigma(x) &\Downarrow \sigma(z) = \\
\sigma(x \Downarrow z) &
\end{aligned}$$

■

Theorem 5.2.2.14 (Deduction-System Ground Equivalence of $\text{PA}_{\text{drt}}^- \text{-ID}$ and $\text{PA}_{\text{drt}}^- \text{-ID}'$)
 $\text{PA}_{\text{drt}}^- \text{-ID}$ and $\text{PA}_{\text{drt}}^- \text{-ID}'$ are deduction-system ground equivalent.

Proof Both $T(\text{PA}_{\text{drt}}^- \text{-ID})$ and $T(\text{PA}_{\text{drt}}^- \text{-ID}')$ have the same signature and the same set of deduction rules, so trivially the same equalities hold between closed terms in the respective bisimulation models. ■

Theorem 5.2.2.15 (Soundness and Completeness)

Let P and P' be two process algebras, with corresponding bisimulation structures M and M' , that are ground equivalent. Then P is a sound axiomatization of M iff P' is a sound axiomatization of M' . Furthermore, P is a complete axiomatization of M iff P' is a complete axiomatization of M' .

Proof First we prove the claim regarding soundness. Suppose that P is a sound axiomatization of M . Let s and t be closed terms of P' , and suppose that $P' \vdash s = t$. By the axiom ground equivalence of P and P' we obtain $P \vdash s = t$. Using that P is a sound axiomatization of M we have that $M \models s \sim_p t$. By the deduction-system ground equivalence of P and P' it follows that $M' \models s \sim_{p'} t$. The proof in the other direction is analogous.

The second part of the theorem, regarding completeness, is proven in the same way. ■

Remark 5.2.2.16 (Soundness and Completeness versus Elimination)

Note that it is useless to extend Theorem 5.2.2.15 to include an elimination result, as in general elimination is used to prove axiom ground equivalence between P and P' . In the proof of Theorem 5.2.2.13 on page 117, for example, we use elimination to allow ourselves to restrict the proof to basic terms.

Corollary 5.2.2.17 (Soundness of $\text{PA}_{\text{drt}}^- \text{-ID}'$)

The set of closed $\text{PA}_{\text{drt}}^- \text{-ID}'$ terms modulo bisimulation equivalence is a model of $\text{PA}_{\text{drt}}^- \text{-ID}'$.

Proof We use the ground equivalence method described in Proof Outline 4.2.2.3 on page 70. The proof then follows immediately from the following observations and Theorem 5.2.2.15:

- (i). The process algebras $\text{PA}_{\text{drt}}^- \text{-ID}$ and $\text{PA}_{\text{drt}}^- \text{-ID}'$ are ground equivalent (see Theorems 5.2.2.13 and 5.2.2.14),
- (ii). Soundness of $\text{PA}_{\text{drt}}^- \text{-ID}$ (see Theorem 5.2.1.10).

■

Remark 5.2.2.18 (Soundness of $PA_{\text{drt}}^- \text{-ID}'$)

Soundness of a slightly different version of $PA_{\text{drt}}^- \text{-ID}'$ is also claimed (without proof) in Theorem 3.4.3 of BAETEN AND VERHOEF [37] (where $PA_{\text{drt}}^- \text{-ID}'$ is called $PA_{\delta \text{drt}}$).

Corollary 5.2.2.19 (Completeness of $PA_{\text{drt}}^- \text{-ID}'$)

The axiom system $PA_{\text{drt}}^- \text{-ID}'$ is a complete axiomatization of the set of closed $PA_{\text{drt}}^- \text{-ID}'$ terms modulo bisimulation equivalence.

Proof We use the ground equivalence method described in Proof Outline 4.2.3.3 on page 71. The proof then follows immediately from the following observations and Theorem 5.2.2.15:

- (i). The process algebras $PA_{\text{drt}}^- \text{-ID}$ and $PA_{\text{drt}}^- \text{-ID}'$ are ground equivalent (see Theorems 5.2.2.13 and 5.2.2.14),
- (ii). Completeness of $PA_{\text{drt}}^- \text{-ID}$ (see Theorem 5.2.1.14).

■

Remark 5.2.2.20 (Completeness of $PA_{\text{drt}}^- \text{-ID}'$)

Completeness of a slightly different version of $PA_{\text{drt}}^- \text{-ID}'$ is also claimed (without proof) in Theorem 3.4.5 of BAETEN AND VERHOEF [37] (where $PA_{\text{drt}}^- \text{-ID}'$ is called $PA_{\delta \text{drt}}$).

5.2.3 $ACP_{\text{drt}}^- \text{-ID}$

In this section, we modify $PA_{\text{drt}}^- \text{-ID}$ by extending the free merge to a merge, where the axioms for the communication merge are based on the σ/ν -representation of $BPA_{\text{drt}}^- \text{-ID}$ terms shown in Lemma 3.2.2.10 on page 50. The resulting process algebra is called $ACP_{\text{drt}}^- \text{-ID}$.

Definition 5.2.3.1 (Signature of $ACP_{\text{drt}}^- \text{-ID}$)

The signature of $ACP_{\text{drt}}^- \text{-ID}$ consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , the “now” operator ν_{rel} , the *merge operator* \parallel , the *left merge operator* \ll , the *communication merge operator* $|$, and the *encapsulation operator* ∂_H .

Definition 5.2.3.2 (Axioms of $ACP_{\text{drt}}^- \text{-ID}$)

The process algebra $ACP_{\text{drt}}^- \text{-ID}$ is axiomatized by the axioms of $PA_{\text{drt}}^- \text{-ID}$ given in Definition 5.2.1.2 on page 106 *minus* Axiom DRTM1, *plus* the Axioms DRTCM1–DRTCM5, DRTCM12–DRTCM13, DRTCF and DRTD1–DRTD5 shown in Table 5.6 on the next page, and Axioms DRTCM6–DRTCM7 shown in Table 5.7 on the following page: $ACP_{\text{drt}}^- \text{-ID} = A1\text{--}A5 + \text{DRT1--DRT5} + \text{DCS1--DCS4} + \text{DRTM2--DRTM6} + \text{DRTCM1--DRTCM7} + \text{DRTCM12--DRTCM13} + \text{DRTCF} + \text{DRTD1--DRTD5}$.

☞ Axioms DRTCM1–DRTCM4, DRTCM12, DRTCM13, DRTCF, and DRTD1–DRTD4 are straightforward reformulations of their untimed counterparts, Axioms CM1–CM6, CF, and D1–D4 from Table 2.18 on page 29.

$x \parallel y = x \parallel\!\! \parallel y + y \parallel\!\! \parallel x + x \mid y$	DRTCM1
$\underline{a} \mid \underline{b} \cdot x = (\underline{a} \mid \underline{b}) \cdot x$	DRTCM2
$\underline{a} \cdot x \mid \underline{b} = (\underline{a} \mid \underline{b}) \cdot x$	DRTCM3
$\underline{a} \cdot x \mid \underline{b} \cdot y = (\underline{a} \mid \underline{b}) \cdot (x \parallel y)$	DRTCM4
$\sigma_{\text{rel}}(x) \mid \sigma_{\text{rel}}(y) = \sigma_{\text{rel}}(x \mid y)$	DRTCM5
$(x + y) \mid z = x \mid z + y \mid z$	DRTCM12
$x \mid (y + z) = x \mid y + x \mid z$	DRTCM13
$\underline{a} \mid \underline{b} = \underline{c} \quad \text{if } \gamma(a, b) = c$	DRTCF
$\partial_H(\underline{a}) = \underline{a} \quad \text{if } a \notin H$	DRTD1
$\partial_H(\underline{a}) = \underline{\delta} \quad \text{if } a \in H$	DRTD2
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	DRTD3
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	DRTD4
$\partial_H(\sigma_{\text{rel}}(x)) = \sigma_{\text{rel}}(\partial_H(x))$	DRTD5

Table 5.6: Axioms for merge and encapsulation.

$\sigma_{\text{rel}}(x) \mid \nu_{\text{rel}}(y) = \underline{\delta}$	DRTCM6
$\nu_{\text{rel}}(x) \mid \sigma_{\text{rel}}(y) = \underline{\delta}$	DRTCM7

Table 5.7: Additional axioms for $\text{ACP}_{\text{drt}}^-$ -ID.

Axiom DRTCM5 expresses that the time-unit delay operator distributes over the communication merge (due to the fact that time steps do not communicate and time factorization). Axioms DRTCM6 and DRTCM7 express that if one side of a communication merge *must* do its first action in the current time slice and the other side *cannot* do an action in the current time slice, the communication merge collapses to an undelayable deadlock, as no communication between the initial actions of both sides is possible. Axiom DRTD5 expresses that the encapsulation commutes with the time-unit delay operator: time cannot be encapsulated.

Definition 5.2.3.3 (Semantics of $\text{ACP}_{\text{drt}}^-$ -ID)

The semantics of $\text{ACP}_{\text{drt}}^-$ -ID are given by the term-deduction system $T(\text{ACP}_{\text{drt}}^- \text{-ID})$ induced by the deduction rules for PA_{drt}^- -ID given in Definition 5.2.1.3 on page 107, the deduction rules for the merge shown in Table 5.8 on the next page, and the deduction rules for the encapsulation shown in Table 5.9 on the facing page.

☞ These deduction rules are identical to their untimed counterparts, save the additional two rules that express that a communication merge and an encapsulation can only do a time step if their arguments can.

$\frac{x \xrightarrow{a} x', y \xrightarrow{b} y', y(a, b) = c}{x \parallel y \xrightarrow{c} x' \parallel y'}$	$\frac{x \xrightarrow{a} x', y \xrightarrow{b} y', y(a, b) = c}{x \mid y \xrightarrow{c} x' \mid y'}$
$\frac{x \xrightarrow{a} x', y \xrightarrow{b} \surd, y(a, b) = c}{x \parallel y \xrightarrow{c} x'}$	$\frac{x \xrightarrow{a} x', y \xrightarrow{b} \surd, y(a, b) = c}{x \mid y \xrightarrow{c} x'}$
$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} y', y(a, b) = c}{x \parallel y \xrightarrow{c} y'}$	$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} y', y(a, b) = c}{x \mid y \xrightarrow{c} y'}$
$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} \surd, y(a, b) = c}{x \parallel y \xrightarrow{c} \surd}$	$\frac{x \xrightarrow{a} \surd, y \xrightarrow{b} \surd, y(a, b) = c}{x \mid y \xrightarrow{c} \surd}$
$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \mid y \xrightarrow{\sigma} x' \mid y'}$	

Table 5.8: Additional deduction rules for merge.

$\frac{x \xrightarrow{a} x', a \notin H}{\partial_H(x) \xrightarrow{a} \partial_H(x')}$	$\frac{x \xrightarrow{a} \surd, a \notin H}{\partial_H(x) \xrightarrow{a} \surd}$	$\frac{x \xrightarrow{\sigma} x'}{\partial_H(x) \xrightarrow{\sigma} \partial_H(x')}$
---	---	---

Table 5.9: Deduction rules for encapsulation.

Definition 5.2.3.4 (Bisimulation and Bisimulation Model for $\text{ACP}_{\text{drt}}^-$ -ID)

Bisimulation for $\text{ACP}_{\text{drt}}^-$ -ID and the corresponding bisimulation model are defined in the same way as for $\text{BPA}_{\text{drt}}^-$ - δ and BPA respectively. Replace “ $\text{BPA}_{\text{drt}}^-$ - δ ” by “ $\text{ACP}_{\text{drt}}^-$ -ID” in Definition 3.2.1.8 on page 45 and “BPA” by “ $\text{ACP}_{\text{drt}}^-$ -ID” in Definition 2.3.1.16 on page 12.

Definition 5.2.3.5 (Basic Terms of $\text{ACP}_{\text{drt}}^-$ -ID)

If we speak of basic terms in the context of $\text{ACP}_{\text{drt}}^-$ -ID, we mean $(\sigma, \underline{\delta})$ -basic terms as defined in Definition 3.2.2.6 on page 49.

Definition 5.2.3.6 (Number of Symbols of an $\text{ACP}_{\text{drt}}^-$ -ID Term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). For $a \in A_\delta$, we define $n(\underline{a}) = 1$,
- (ii). for closed $\text{ACP}_{\text{drt}}^-$ -ID terms x and y , we define $n(x + y) = n(x \cdot y) = n(x \parallel y) = n(x \parallel\!\!\! \parallel y) = n(x \mid y) = n(x) + n(y) + 1$,
- (iii). for a closed $\text{ACP}_{\text{drt}}^-$ -ID term x , we define $n(\sigma(x)) = n(\nu(x)) = n(\partial_H(x)) = n(x) + 1$.

Theorem 5.2.3.7 (Elimination for ACP_{drt}^- -ID)

Let t be a closed ACP_{drt}^- -ID term. Then there is a closed BPA_{drt}^- -ID term s such that ACP_{drt}^- -ID $\vdash s = t$.

Proof We use the direct method described in Proof Outline 4.2.1.2 on page 68. Let t be a closed ACP_{drt}^- -ID term. The theorem is proven by induction on $n(t)$ and case distinction on the general structure of t .

- (i). $t \equiv \underline{a}$ for some $a \in A_\delta$. Then t is a closed BPA_{drt}^- -ID term.
- (ii). $t \equiv t_1 + t_2$ for closed ACP_{drt}^- -ID terms t_1 and t_2 . By induction there are closed BPA_{drt}^- -ID terms s_1 and s_2 such that ACP_{drt}^- -ID $\vdash t_1 = s_1$ and ACP_{drt}^- -ID $\vdash t_2 = s_2$. But then also ACP_{drt}^- -ID $\vdash t_1 + t_2 = s_1 + s_2$ and $s_1 + s_2$ is a closed BPA_{drt}^- -ID term.
- (iii). $t \equiv t_1 \cdot t_2$ for closed ACP_{drt}^- -ID terms t_1 and t_2 . This case is treated analogously to case (ii).
- (iv). $t \equiv \sigma(t_1)$ for a closed ACP_{drt}^- -ID term t_1 . This case is treated analogously to case (ii).
- (v). $t \equiv \nu(t_1)$ for a closed ACP_{drt}^- -ID term t_1 . This case is treated analogously to case (ii).
- (vi). $t \equiv t_1 \parallel t_2$ for closed ACP_{drt}^- -ID terms t_1 and t_2 . By induction there are closed BPA_{drt}^- -ID terms s_1 and s_2 such that ACP_{drt}^- -ID $\vdash t_1 = s_1$ and ACP_{drt}^- -ID $\vdash t_2 = s_2$. By Theorem 4.3.2.1, the elimination theorem for BPA_{drt}^- -ID, there are basic terms r_1 and r_2 such that BPA_{drt}^- -ID $\vdash s_1 = r_1$ and BPA_{drt}^- -ID $\vdash s_2 = r_2$. But then also, ACP_{drt}^- -ID $\vdash t_1 = r_1$, ACP_{drt}^- -ID $\vdash t_2 = r_2$, and ACP_{drt}^- -ID $\vdash t_1 \parallel t_2 = r_1 \parallel r_2$. We prove this case by induction on the structure of basic term r_1 :
 - (a) $r_1 \equiv \underline{a}$ for some $a \in A_\delta$. Then ACP_{drt}^- -ID $\vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{a} \parallel r_2 = \underline{a} \cdot r_2$, and $\underline{a} \cdot r_2$ is a closed BPA_{drt}^- -ID term.
 - (b) $r_1 \equiv \underline{a} \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Then ACP_{drt}^- -ID $\vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{a} \cdot r'_1 \parallel r_2 = \underline{a} \cdot (r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed BPA_{drt}^- -ID term p such that ACP_{drt}^- -ID $\vdash r'_1 \parallel r_2 = p$. Then, ACP_{drt}^- -ID $\vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) = \underline{a} \cdot p$, and $\underline{a} \cdot p$ is a closed BPA_{drt}^- -ID term.
 - (c) $r_1 \equiv r'_1 + r''_1$ for basic terms r'_1 and r''_1 . Then ACP_{drt}^- -ID $\vdash t_1 \parallel t_2 = r_1 \parallel r_2 = (r'_1 + r''_1) \parallel r_2 = r'_1 \parallel r_2 + r''_1 \parallel r_2$. By induction there exist closed BPA_{drt}^- -ID terms p_1 and p_2 such that ACP_{drt}^- -ID $\vdash r'_1 \parallel r_2 = p_1$ and ACP_{drt}^- -ID $\vdash r''_1 \parallel r_2 = p_2$. Then also ACP_{drt}^- -ID $\vdash t_1 \parallel t_2 = r'_1 \parallel r_2 + r''_1 \parallel r_2 = p_1 + p_2$, and $p_1 + p_2$ is a closed BPA_{drt}^- -ID term.
 - (d) $r_1 \equiv \sigma(r'_1)$ for a basic term r'_1 . By Lemma 3.2.2.10 there is a basic term r'_2 such that either ACP_{drt}^- -ID $\vdash r_2 = \nu(r_2)$ or ACP_{drt}^- -ID $\vdash r_2 = \nu(r_2) + \sigma(r'_2)$ with $n(r'_2) < n(r_2)$. With case analysis we obtain:
 1. $r_2 = \nu(r_2)$. Then ACP_{drt}^- -ID $\vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel r_2 = \sigma(r'_1) \parallel \nu(r_2) = \underline{\delta}$, and $\underline{\delta}$ is a closed BPA_{drt}^- -ID term.

2. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 . Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel t_2 = \sigma(r'_1) \parallel r_2 = \sigma(r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) = \sigma(r'_1 \parallel r'_2)$. By the induction hypothesis there is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term p such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash r'_1 \parallel r'_2 = p$. But then also $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = \sigma(r'_1 \parallel r'_2) = \sigma(p)$, and $\sigma(p)$ is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term.
- (vii). $t \equiv t_1 \mid t_2$ for closed $\text{ACP}_{\text{drt}}^- \text{-ID}$ terms t_1 and t_2 . By induction there are closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ terms s_1 and s_2 such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 = s_1$ and $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_2 = s_2$. By Theorem 4.3.2.1, the elimination theorem for $\text{BPA}_{\text{drt}}^- \text{-ID}$, there are basic terms r_1 and r_2 such that $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s_1 = r_1$ and $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s_2 = r_2$. But then also, $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 = r_1$, $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_2 = r_2$, and $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r_1 \mid r_2$. We prove this case by simultaneous induction on the structure of basic terms r_1 and r_2 . We examine all possible cases:
- $r_1 \equiv \underline{a}$ and $r_2 \equiv \underline{b}$ for some $a, b \in A_\delta$. Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \mid \underline{b} = \underline{c}$, and \underline{c} is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term.
 - $r_1 \equiv \underline{a}$ and $r_2 \equiv \underline{b} \cdot r'_2$ for some $a, b \in A_\delta$ and some basic term r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \mid \underline{b} \cdot r'_2 = \underline{c} \cdot r'_2$, and $\underline{c} \cdot r'_2$ is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term.
 - $r_1 \equiv \underline{a} \cdot r'_1$ and $r_2 \equiv \underline{b}$ for some $a, b \in A_\delta$ and some basic term r'_1 . This case is treated symmetrically to the previous case.
 - $r_1 \equiv \underline{a} \cdot r'_1$ and $r_2 \equiv \underline{b} \cdot r'_2$ for some $a, b \in A_\delta$ and some basic terms r'_1 and r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \cdot r'_1 \mid \underline{b} \cdot r'_2 = \underline{c} \cdot (r'_1 \parallel r'_2)$. By the induction hypothesis there exists a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term s' such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash r'_1 \parallel r'_2 = s'$. So $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = \underline{c} \cdot (r'_1 \parallel r'_2) = \underline{c} \cdot s'$, and $\underline{c} \cdot s'$ is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term.
 - $r_1 \equiv r'_1 + r''_1$ for some basic terms r'_1 and r''_1 , and r_2 is of arbitrary form. Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r_1 \mid r_2 = (r'_1 + r''_1) \mid r_2 = r'_1 \mid r_2 + r''_1 \mid r_2$. By the induction hypothesis there exist closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash r'_1 \mid r_2 = p_1$ and $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash r''_1 \mid r_2 = p_2$. So, we have $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r'_1 \mid r_2 + r''_1 \mid r_2 = p_1 + p_2$, and $p_1 + p_2$ is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term.
 - r_1 is of arbitrary form and $r_2 \equiv r'_2 + r''_2$ for some basic terms r'_2 and r''_2 . This case is treated symmetrically to the previous case.
 - $r_1 \equiv \sigma(r'_1)$ for some basic term r'_1 , and r_2 is of arbitrary form. By Lemma 3.2.2.10 there is a basic term r'_2 such that either $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash r_2 = \nu(r_2)$ or $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash r_2 = \nu(r_2) + \sigma(r'_2)$ with $n(r'_2) < n(r_2)$. With case analysis we obtain:
 - $r_2 = \nu(r_2)$. Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r_1 \mid r_2 = \sigma(r'_1) \mid \nu(r_2) = \underline{\delta}$, and $\underline{\delta}$ is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term.
 - $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 . Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = r_1 \mid r_2 = \sigma(r'_1) \mid (\nu(r_2) + \sigma(r'_2)) = \sigma(r'_1) \mid \sigma(r'_2) = \sigma(r'_1 \mid r'_2)$. By the induction hypothesis there is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term p such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash r'_1 \mid r'_2 = p$. But then also $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \mid t_2 = \sigma(r'_1 \mid r'_2) = \sigma(p)$, and $\sigma(p)$ is a closed $\text{BPA}_{\text{drt}}^- \text{-ID}$ term.
 - r_1 is of arbitrary form and $r_2 \equiv \sigma(r'_2)$ for some basic term r'_2 . This case is treated symmetrically to the previous case.

- (viii). $t \equiv t_1 \parallel t_2$ for closed $\text{ACP}_{\text{drt}}^-$ -ID terms t_1 and t_2 . Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1 + t_1 | t_2$. By (vi) and (vii) there are closed $\text{BPA}_{\text{drt}}^-$ -ID terms p_1, p_2 , and p_3 , such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = p_1$, $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_2 \parallel t_1 = p_2$, and $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 | t_2 = p_3$. But then also $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1 + t_1 | t_2 = p_1 + p_2 + p_3$, and $p_1 + p_2 + p_3$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
- (ix). $t \equiv \partial_H(t_1)$ for a closed $\text{ACP}_{\text{drt}}^-$ -ID term t_1 . By induction there is a closed $\text{BPA}_{\text{drt}}^-$ -ID term s_1 such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 = s_1$. By Theorem 4.3.2.1, the elimination theorem for $\text{BPA}_{\text{drt}}^-$ -ID, there is a basic term r_1 such that $\text{BPA}_{\text{drt}}^- \text{-ID} \vdash s_1 = r_1$. But then also, $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash t_1 = r_1$, and $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(r_1)$. We prove this case by induction on the structure of basic term r_1 :
- (a) $r_1 \equiv \underline{a}$ for some $a \in A_\delta$. Suppose $a \notin H$. Then, $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\underline{a}) = \underline{a}$, and \underline{a} is a closed $\text{BPA}_{\text{drt}}^-$ -ID term. Otherwise, $a \in H$, and we get $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\underline{a}) = \underline{\delta}$, and $\underline{\delta}$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
 - (b) $r_1 \equiv \underline{a} \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\underline{a} \cdot r'_1) = \partial_H(\underline{a}) \cdot \partial_H(r'_1)$. By the induction hypothesis there exist closed $\text{BPA}_{\text{drt}}^-$ -ID terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(\underline{a}) = p_1$ and $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(r'_1) = p_2$. Then also $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(\underline{a}) \cdot \partial_H(r'_1) = p_1 \cdot p_2$, and $p_1 \cdot p_2$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
 - (c) $r_1 \equiv r'_1 + r''_1$ for closed $\text{BPA}_{\text{drt}}^-$ -ID terms r'_1 and r''_1 . Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(r'_1 + r''_1) = \partial_H(r'_1) + \partial_H(r''_1)$. By the induction hypothesis there exist closed $\text{BPA}_{\text{drt}}^-$ -ID terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(r'_1) = p_1$ and $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(r''_1) = p_2$. Then also $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(r'_1) + \partial_H(r''_1) = p_1 + p_2$, and $p_1 + p_2$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.
 - (d) $r_1 \equiv \sigma(r'_1)$ for some closed $\text{BPA}_{\text{drt}}^-$ -ID term r'_1 . Then $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\sigma(r'_1)) = \sigma(\partial_H(r'_1))$. By the induction hypothesis there exist a closed $\text{BPA}_{\text{drt}}^-$ -ID term p such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(r'_1) = p$. Then also $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \partial_H(t_1) = \sigma(\partial_H(r'_1)) = \sigma(p)$, and $\sigma(p)$ is a closed $\text{BPA}_{\text{drt}}^-$ -ID term.

■

Corollary 5.2.3.8 (Elimination for $\text{ACP}_{\text{drt}}^-$ -ID)

Let t be a closed $\text{ACP}_{\text{drt}}^-$ -ID term. Then there is a basic term s such that $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash s = t$.

Proof This follows immediately from:

- (i). The elimination theorem for $\text{ACP}_{\text{drt}}^-$ -ID (see Theorem 5.2.3.7),
- (ii). the elimination theorem for $\text{BPA}_{\text{drt}}^-$ -ID (see Theorem 4.3.2.1),
- (iii). the fact that all axioms of $\text{BPA}_{\text{drt}}^-$ -ID are also contained in $\text{ACP}_{\text{drt}}^-$ -ID.

■

Remark 5.2.3.9 (Elimination for ACP_{drt}^- -ID)

Elimination for ACP_{drt}^- -ID is also claimed (without proof) in Theorem 4.2 of BAETEN AND RENIERS [35].

Theorem 5.2.3.10 (Soundness of ACP_{drt}^- -ID)

The set of closed ACP_{drt}^- -ID terms modulo bisimulation equivalence is a model of ACP_{drt}^- -ID.

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. We only treat the the axioms which are added to BPA_{drt}^- -ID to obtain ACP_{drt}^- -ID.

Axioms DRTM2–DRTM6 The proofs for the soundness of these axioms with respect to PA_{drt}^- -ID that are given in Theorem 5.2.1.10 remain valid, since there are no new deduction rules dealing with \parallel .

Axiom DRTCM1 Take the relation:

$$R = \{(s, s), (s \parallel t, t \parallel s), (s \parallel t, s \parallel t + t \parallel s + s \mid t) \mid s, t \in C(ACP_{\text{drt}}^- \text{-ID})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose that $s \parallel t \xrightarrow{a} p$. First we look at the $(s \parallel t, t \parallel s)$ pairs. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} p_1$ and $p \equiv p_1 \parallel t$. Then $t \parallel s \xrightarrow{a} t \parallel p_1$, and $(p_1 \parallel t, t \parallel p_1) \in R$
 - (b) $t \xrightarrow{a} p_2$ and $p \equiv s \parallel p_2$. Then $t \parallel s \xrightarrow{a} p_2 \parallel s$, and $(s \parallel p_2, p_2 \parallel s) \in R$.
 - (c) $s \xrightarrow{a} \surd$ and $p \equiv t$. Then $t \parallel s \xrightarrow{a} t$, and $(t, t) \in R$.
 - (d) $t \xrightarrow{a} \surd$ and $p \equiv s$. Then $t \parallel s \xrightarrow{a} s$, and $(s, s) \in R$.
 - (e) $s \xrightarrow{b} p_1, t \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. Then $t \parallel s \xrightarrow{a} p_2 \parallel p_1$, and $(p_1 \parallel p_2, p_2 \parallel p_1) \in R$.
 - (f) $s \xrightarrow{b} \surd, t \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_2$. Then $t \parallel s \xrightarrow{a} p_2$, and $(p_2, p_2) \in R$.
 - (g) $s \xrightarrow{b} p_1, t \xrightarrow{c} \surd, \gamma(b, c) = a$, and $p \equiv p_1$. Then $t \parallel s \xrightarrow{a} p_1$, and $(p_1, p_1) \in R$.

We continue with the $(s \parallel t, s \parallel t + t \parallel s)$ pairs. Distinguishing the same cases as above, we derive:

- (a) $s \parallel t \xrightarrow{a} p_1 \parallel t$. Then $s \parallel t + t \parallel s \mid t \xrightarrow{a} p_1 \parallel t$, and $(p_1 \parallel t, p_1 \parallel t) \in R$.
 - (b) $t \parallel s \xrightarrow{a} p_2 \parallel s$. Then $s \parallel t + t \parallel s \mid t \xrightarrow{a} p_2 \parallel s$, and $(s \parallel p_2, p_2 \parallel s) \in R$.
 - (c) $s \parallel t \xrightarrow{a} t$. Then $s \parallel t + t \parallel s \mid t \xrightarrow{a} t$, and $(t, t) \in R$.
 - (d) $t \parallel s \xrightarrow{a} s$. Then $s \parallel t + t \parallel s \mid t \xrightarrow{a} s$, and $(s, s) \in R$.
 - (e) $s \mid t \xrightarrow{a} p_1 \parallel p_2$. Then $s \parallel t + t \parallel s \mid t \xrightarrow{a} p_1 \parallel p_2$, and $(p_1 \parallel p_2, p_1 \parallel p_2) \in R$.
 - (f) $s \mid t \xrightarrow{a} p_2$. Then $s \parallel t + t \parallel s \mid t \xrightarrow{a} p_2$, and $(p_2, p_2) \in R$.
 - (g) $s \mid t \xrightarrow{a} p_1$. Then $s \parallel t + t \parallel s \mid t \xrightarrow{a} p_1$, and $(p_1, p_1) \in R$.
- (ii). Suppose that $s \parallel t \xrightarrow{a} \surd$. By inspection of the deduction rules we can conclude that $s \xrightarrow{b} \surd, t \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. Therefore, $t \parallel s \xrightarrow{a} \surd$, and continuing, $s \mid t \xrightarrow{a} \surd$, so $s \parallel t + t \parallel s \mid t \xrightarrow{a} \surd$.
 - (iii). Suppose that $s \parallel t \xrightarrow{\sigma} p$. By inspection of the deduction rules we can conclude that $s \xrightarrow{\sigma} p_1$ and $t \xrightarrow{\sigma} p_2$ and $p \equiv p_1 \parallel p_2$. Therefore, $t \parallel s \xrightarrow{\sigma} p_2 \parallel p_1$, and note that $(p_1 \parallel p_2, p_2 \parallel p_1) \in R$. Continuing, we also have $s \parallel t \xrightarrow{\sigma} p_1 \parallel p_2$, $t \parallel s \xrightarrow{\sigma} p_2 \parallel p_1$, and $s \mid t \xrightarrow{\sigma} p_1 \parallel p_2$. Therefore, $s \parallel t + t \parallel s \mid t \xrightarrow{\sigma} p_1 \parallel p_2 + p_2 \parallel p_1 + p_1 \mid p_2$, and note that $(p_1 \parallel p_2, p_1 \parallel p_2 + p_2 \parallel p_1 + p_1 \mid p_2) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose that $t \parallel s \xrightarrow{a} p$. This case is handled in the same way as the corresponding (sub)case for the left-hand side shown above.
- (ii). Suppose that $t \parallel s \xrightarrow{a} \surd$. This case is handled in the same way as the corresponding (sub)case for the left-hand side shown above.
- (iii). Suppose that $t \parallel s \xrightarrow{\sigma} p$. This case is handled in the same way as the corresponding (sub)case for the left-hand side shown above.
- (iv). Suppose that $s \parallel t + t \parallel s + s \mid t \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} p_1$ and $p \equiv p_1 \parallel t$. Then $s \parallel t \xrightarrow{a} p_1 \parallel t$, and $(p_1 \parallel t, p_1 \parallel t) \in R$.
 - (b) $t \xrightarrow{a} p_2$ and $p \equiv p_2 \parallel s$. Then $s \parallel t \xrightarrow{a} s \parallel p_2$, and $(s \parallel p_2, p_2 \parallel s) \in R$.
 - (c) $s \xrightarrow{a} \surd$ and $p \equiv t$. Then $s \parallel t \xrightarrow{a} t$, and $(t, t) \in R$.
 - (d) $t \xrightarrow{a} \surd$ and $p \equiv s$. Then $s \parallel t \xrightarrow{a} s$, and $(s, s) \in R$.
 - (e) $s \xrightarrow{b} p_1, t \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. Then $s \parallel t \xrightarrow{a} p_1 \parallel p_2$, and $(p_1 \parallel p_2, p_1 \parallel p_2) \in R$.
 - (f) $s \xrightarrow{b} \surd, t \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_2$. Then $s \parallel t \xrightarrow{a} p_2$, and $(p_2, p_2) \in R$.
 - (g) $s \xrightarrow{b} p_1, t \xrightarrow{c} \surd, \gamma(b, c) = a$, and $p \equiv p_1$. Then $s \parallel t \xrightarrow{a} p_1$, and $(p_1, p_1) \in R$.
- (v). Suppose that $s \parallel t + t \parallel s + s \mid t \xrightarrow{a} \surd$. By inspection of the deduction rules we can conclude that $s \xrightarrow{b} \surd, t \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. Therefore, $s \parallel t \xrightarrow{a} \surd$.
- (vi). Suppose that $s \parallel t + t \parallel s + s \mid t \xrightarrow{\sigma} p$. By inspection of the deduction rules we can conclude that $s \xrightarrow{\sigma} p_1, t \xrightarrow{\sigma} p_2$, and $p \equiv p_1 \parallel p_2 + p_2 \parallel p_1 + p_1 \mid p_2$. Since both s and t can perform a σ transition, we obtain $s \parallel t \xrightarrow{\sigma} p_1 \parallel p_2$, and note that $(p_1 \parallel p_2, p_1 \parallel p_2 + p_2 \parallel p_1 + p_1 \mid p_2) \in R$.

Axiom DRTCM2 Take the relation:

$$R = \{(s, s), (\underline{a} \mid \underline{b} \cdot s, (\underline{a} \mid \underline{b}) \cdot s) \mid s \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. First, if $\gamma(a, b) = \delta$ there are no transitions possible on either side, and we are done. Otherwise, suppose $\gamma(a, b) = c$. Then the only possible transition on the left-hand side is $\underline{a} \cdot s \mid \underline{b} \xrightarrow{c} s$, and the only possible transition on the right-hand side is $(\underline{a} \mid \underline{b}) \cdot s \xrightarrow{c} s$, and note that $(s, s) \in R$.

Axiom DRTCM3 Take the relation:

$$R = \{(s, s), (\underline{a} \cdot s \mid \underline{b}, (\underline{a} \mid \underline{b}) \cdot s) \mid s \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

This axiom is treated symmetrically to the previous axiom.

Axiom DRTCM4 Take the relation:

$$R = \{(s, s), (\underline{a} \cdot s \mid \underline{b} \cdot t, (\underline{a} \mid \underline{b}) \cdot (s \parallel t)) \mid s, t \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. First, if $\gamma(a, b) = \delta$ there are no transitions possible on either side, and we are done. Otherwise, suppose $\gamma(a, b) = c$. Then the only possible transition on the left-hand side is $\underline{a} \cdot s \mid \underline{b} \cdot t \xrightarrow{c} s \parallel t$, and the only possible transition on the right-hand side is $(\underline{a} \mid \underline{b}) \cdot (s \parallel t) \xrightarrow{c} s \parallel t$, and note that $(s \parallel t, s \parallel t) \in R$.

Axiom DRTCM5 Take the relation:

$$R = \{(\sigma(s) \mid \sigma(t), \sigma(s \mid t)) \mid s, t \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. The only possible transition of the left-hand side is $\sigma(s) \mid \sigma(t) \xrightarrow{\sigma} s \mid t$, and the only possible transition of the right-hand side is $\sigma(s \mid t) \xrightarrow{\sigma} s \mid t$, and note that $(s \mid t, s \mid t) \in R$.

Axiom DRTCM6 Take the relation:

$$R = \{(\sigma(s) \mid \nu(t), \underline{\delta}) \mid s, t \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $\sigma(s) \mid \nu(t) \nrightarrow$. Also for the right-hand side there are no transitions possible: $\underline{\delta} \nrightarrow$.

Axiom DRTCM7 Take the relation:

$$R = \{(\nu(s) \mid \sigma(t), \underline{\delta}) \mid s, t \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

This axiom is treated symmetrically to the previous axiom.

Axiom DRTCM12 Take the relation:

$$R = \{(s, s), ((s+t) \mid u, s \mid u+t \mid u) \mid s, t, u \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose that $(s+t) \mid u \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{b} p_1, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. Then $s \mid u \xrightarrow{a} p_1 \parallel p_2$, so also $s \mid u+t \mid u \xrightarrow{a} p_1 \parallel p_2$, and note that $(p_1 \parallel p_2, p_1 \parallel p_2) \in R$.
 - (b) $t \xrightarrow{b} p_1, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. This case is treated symmetrically to the previous case.
 - (c) $s \xrightarrow{b} \surd, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_2$. Then $s \mid u \xrightarrow{a} p_2$, so also $s \mid u+t \mid u \xrightarrow{a} p_2$, and note that $(p_2, p_2) \in R$.
 - (d) $t \xrightarrow{b} \surd, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_2$. This case is treated symmetrically to the previous case.
 - (e) $s \xrightarrow{b} p_1, u \xrightarrow{c} \surd, \gamma(b, c) = a$, and $p \equiv p_1$. Then $s \mid u \xrightarrow{a} p_1$, so also $s \mid u+t \mid u \xrightarrow{a} p_1$, and note that $(p_1, p_1) \in R$.
 - (f) $t \xrightarrow{b} p_1, u \xrightarrow{c} \surd, \gamma(b, c) = a$, and $p \equiv p_1$. This case is treated symmetrically to the previous case.
- (ii). Suppose that $(s+t) \mid u \xrightarrow{a} \surd$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{b} \surd, u \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. Then $s \mid u \xrightarrow{a} \surd$, so also $s \mid u+t \mid u \xrightarrow{a} \surd$.
 - (b) $t \xrightarrow{b} \surd, u \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. This case is treated symmetrically to the previous case.
- (iii). Suppose that $(s+t) \mid u \xrightarrow{\sigma} p$. By inspection of the deduction rules we distinguish the following cases:

- (a) $s \xrightarrow{\sigma} p_1, t \xrightarrow{\sigma}, u \xrightarrow{\sigma} p_2$, and $p \equiv p_1 | p_2$. Then $s | u \xrightarrow{\sigma} p_1 | p_2$ and $t | u \xrightarrow{\sigma}$, so $s | u + t | u \xrightarrow{\sigma} p_1 | p_2$, and note that $(p_1 | p_2, p_1 | p_2) \in R$.
- (b) $s \xrightarrow{\sigma}, t \xrightarrow{\sigma} p_1, u \xrightarrow{\sigma} p_2$, and $p \equiv p_1 | p_2$. This case is treated symmetrically to the previous case.
- (c) $s \xrightarrow{\sigma} p_1, t \xrightarrow{\sigma} p_2, u \xrightarrow{\sigma} p_3$, and $p \equiv (p_1 + p_2) | p_3$. Then $s | u \xrightarrow{\sigma} p_1 | p_3$ and $t | u \xrightarrow{\sigma} p_2 | p_3$, so $s | u + t | u \xrightarrow{\sigma} p_1 | p_3 + p_2 | p_3$, and note that $((p_1 + p_2) | p_3, p_1 | p_3 + p_2 | p_3) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose that $s | u + t | u \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{b} p_1, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. Then $s + t \xrightarrow{b} p_1$, so $(s + t) | u \xrightarrow{a} p_1 \parallel p_2$, and note that $(p_1 \parallel p_2, p_1 \parallel p_2) \in R$.
 - (b) $t \xrightarrow{b} p_1, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. This case is treated symmetrically to the previous case.
 - (c) $s \xrightarrow{b} \surd, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_2$. Then $s + t \xrightarrow{b} \surd$, so $(s + t) | u \xrightarrow{a} p_2$, and note that $(p_2, p_2) \in R$.
 - (d) $t \xrightarrow{b} \surd, u \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_2$. This case is treated symmetrically to the previous case.
 - (e) $s \xrightarrow{b} p_1, u \xrightarrow{c} \surd, \gamma(b, c) = a$, and $p \equiv p_1$. Then $s + t \xrightarrow{b} p_1$, so $(s + t) | u \xrightarrow{a} p_1$, and note that $(p_1, p_1) \in R$.
 - (f) $t \xrightarrow{b} p_1, u \xrightarrow{c} \surd, \gamma(b, c) = a$, and $p \equiv p_1$. This case is treated symmetrically to the previous case.
- (ii). Suppose that $s | u + t | u \xrightarrow{a} \surd$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{b} \surd, u \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. Then $s + t \xrightarrow{b} \surd$, so $(s + t) | u \xrightarrow{a} \surd$.
 - (b) $t \xrightarrow{b} \surd, u \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. This case is treated symmetrically to the previous case.
- (iii). Suppose that $s | u + t | u \xrightarrow{\sigma} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{\sigma} p_1, t \xrightarrow{\sigma}, u \xrightarrow{\sigma} p_2$, and $p \equiv p_1 | p_2$. Then $s + t \xrightarrow{\sigma} p_1$, so $(s + t) | u \xrightarrow{\sigma} p_1 | p_2$, and note that $(p_1 | p_2, p_1 | p_2) \in R$.
 - (b) $s \xrightarrow{\sigma}, t \xrightarrow{\sigma} p_1, u \xrightarrow{\sigma} p_2$, and $p \equiv p_1 | p_2$. This case is treated symmetrically to the previous case.
 - (c) $s \xrightarrow{\sigma} p_1, t \xrightarrow{\sigma} p_2, u \xrightarrow{\sigma} p_3$, and $p \equiv p_1 | p_3 + p_2 | p_3$. Then $s + t \xrightarrow{\sigma} p_1 + p_2$, so $(s + t) | u \xrightarrow{\sigma} (p_1 + p_2) | p_3$, and note that $((p_1 + p_2) | p_3, p_1 | p_3 + p_2 | p_3) \in R$.

Axiom DRTCM13 Take the relation:

$$R = \{(s, s), (s | (t + u), s | t + s | u) \mid s, t, u \in C(\text{ACP}_{\text{drt}}\text{-ID})\}$$

This axiom is treated symmetrically to the previous axiom.

Axiom DRTCF Take the relation:

$$R = \{(\underline{a} \mid \underline{b}, \underline{c}) \mid a, b, c \in A_\delta \text{ and } \gamma(a, b) = c\}$$

We look at the transitions of both sides at the same time. If $c \neq \delta$, the only possible transition of the left-hand side is $\underline{a} \mid \underline{b} \xrightarrow{c} \surd$, and the only possible transition of the right-hand side is $\underline{c} \xrightarrow{c} \surd$. If $c = \delta$, there are no transitions possible on either side.

Axiom DRTD1 Take the relation:

$$R = \{(\partial_H(\underline{a}), \underline{a}) \mid a \in A_\delta \text{ and } a \notin H\}$$

We look at the transitions of both sides at the same time. The only possible transition of the left-hand side is $\partial_H(\underline{a}) \xrightarrow{a} \surd$, and the only possible transition of the right-hand side is $\underline{a} \xrightarrow{a} \surd$.

Axiom DRTD2 Take the relation:

$$R = \{(\partial_H(\underline{a}), \underline{\delta}) \mid a \in A_\delta \text{ and } a \in H\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $\partial_H(\underline{a}) \nrightarrow$. Also for the right-hand side there are no transitions possible: $\underline{\delta} \nrightarrow$.

Axiom DRTD3 Take the relation:

$$R = \{(s, s), (\partial_H(s+t), \partial_H(s) + \partial_H(t)) \mid s, t \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose that $\partial_H(s+t) \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} p_1$ and $p \equiv \partial_H(p_1)$. Then we have $\partial_H(s) \xrightarrow{a} \partial_H(p_1)$, so also $\partial_H(s) + \partial_H(t) \xrightarrow{a} \partial_H(p_1)$, and note that $(\partial_H(p_1), \partial_H(p_1)) \in R$.
 - (b) $t \xrightarrow{a} p_2$ and $p \equiv \partial_H(p_2)$. Then we have $\partial_H(t) \xrightarrow{a} \partial_H(p_2)$, so also $\partial_H(s) + \partial_H(t) \xrightarrow{a} \partial_H(p_2)$, and note that $(\partial_H(p_2), \partial_H(p_2)) \in R$.
- (ii). Suppose that $\partial_H(s+t) \xrightarrow{a} \surd$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} \surd$. Then we have $\partial_H(s) \xrightarrow{a} \surd$, so also $\partial_H(s) + \partial_H(t) \xrightarrow{a} \surd$.
 - (b) $t \xrightarrow{a} \surd$. Then we have $\partial_H(t) \xrightarrow{a} \surd$, so also $\partial_H(s) + \partial_H(t) \xrightarrow{a} \surd$.
- (iii). Suppose that $\partial_H(s+t) \xrightarrow{\sigma} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{\sigma} p_1$, $t \xrightarrow{\sigma} \surd$, and $p \equiv \partial_H(p_1)$. Then we have $\partial_H(s) \xrightarrow{\sigma} \partial_H(p_1)$ and $\partial_H(t) \xrightarrow{\sigma} \surd$, so also $\partial_H(s) + \partial_H(t) \xrightarrow{\sigma} \partial_H(p_1)$, and note that $(\partial_H(p_1), \partial_H(p_1)) \in R$.
 - (b) $s \xrightarrow{\sigma} \surd$, $t \xrightarrow{\sigma} p_2$, and $p \equiv \partial_H(p_2)$. Then we have $\partial_H(t) \xrightarrow{\sigma} \partial_H(p_2)$ and $\partial_H(s) \xrightarrow{\sigma} \surd$, so also $\partial_H(s) + \partial_H(t) \xrightarrow{\sigma} \partial_H(p_2)$, and note that $(\partial_H(p_2), \partial_H(p_2)) \in R$.
 - (c) $s \xrightarrow{\sigma} p_1$, $t \xrightarrow{\sigma} p_2$, and $p \equiv \partial_H(p_1 + p_2)$. Then we have $\partial_H(s) \xrightarrow{\sigma} \partial_H(p_1)$ and $\partial_H(t) \xrightarrow{\sigma} \partial_H(p_2)$, so also $\partial_H(s) + \partial_H(t) \xrightarrow{\sigma} \partial_H(p_1) + \partial_H(p_2)$, and note that $(\partial_H(p_1 + p_2), \partial_H(p_1) + \partial_H(p_2)) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose that $\partial_H(s) + \partial_H(t) \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} p_1$ and $p \equiv \partial_H(p_1)$. Then we have $\partial_H(s+t) \xrightarrow{a} \partial_H(p_1)$, and note that $(\partial_H(p_1), \partial_H(p_1)) \in R$.
 - (b) $t \xrightarrow{a} p_2$ and $p \equiv \partial_H(p_2)$. Then we have $\partial_H(s+t) \xrightarrow{a} \partial_H(p_2)$, and note that $(\partial_H(p_2), \partial_H(p_2)) \in R$.
- (ii). Suppose that $\partial_H(s) + \partial_H(t) \xrightarrow{a} \surd$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} \surd$. Then we have $\partial_H(s+t) \xrightarrow{a} \surd$.
 - (b) $t \xrightarrow{a} \surd$. Then we have $\partial_H(s+t) \xrightarrow{a} \surd$.
- (iii). Suppose that $\partial_H(s) + \partial_H(t) \xrightarrow{\sigma} p$.
 - (a) $s \xrightarrow{\sigma} p_1, t \xrightarrow{\sigma} \surd$, and $p \equiv \partial_H(p_1)$. Then we have $\partial_H(s+t) \xrightarrow{\sigma} \partial_H(p_1)$, and note that $(\partial_H(p_1), \partial_H(p_1)) \in R$.
 - (b) $s \xrightarrow{\sigma} \surd, t \xrightarrow{\sigma} p_2$, and $p \equiv \partial_H(p_2)$. Then we have $\partial_H(s+t) \xrightarrow{\sigma} \partial_H(p_2)$, and note that $(\partial_H(p_2), \partial_H(p_2)) \in R$.
 - (c) $s \xrightarrow{\sigma} p_1, t \xrightarrow{\sigma} p_2$, and $p \equiv \partial_H(p_1) + \partial_H(p_2)$. Then we have $\partial_H(s+t) \xrightarrow{\sigma} \partial_H(p_1 + p_2)$, and note that $(\partial_H(p_1 + p_2), \partial_H(p_1) + \partial_H(p_2)) \in R$.

Axiom DRTD4 Take the relation:

$$R = \{(s, s), (\partial_H(s \cdot t), \partial_H(s) \cdot \partial_H(t)) \mid s, t \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose that $\partial_H(s \cdot t) \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} p_1$ and $p \equiv \partial_H(p_1 \cdot t)$. Then we have $\partial_H(s) \xrightarrow{a} \partial_H(p_1)$, so also $\partial_H(s) \cdot \partial_H(t) \xrightarrow{a} \partial_H(p_1) \cdot \partial_H(t)$, and note that $(\partial_H(p_1 \cdot t), \partial_H(p_1) \cdot \partial_H(t)) \in R$.
 - (b) $s \xrightarrow{a} \surd$ and $p \equiv \partial_H(t)$. Then $\partial_H(s) \xrightarrow{a} \surd$, so also $\partial_H(s) \cdot \partial_H(t) \xrightarrow{a} \partial_H(t)$, and note that $(\partial_H(t), \partial_H(t)) \in R$.
- (ii). Suppose that $\partial_H(s \cdot t) \xrightarrow{a} \surd$. This case cannot occur.
- (iii). Suppose that $\partial_H(s \cdot t) \xrightarrow{\sigma} p$. Then necessarily $s \xrightarrow{\sigma} p_1$ and $p \equiv \partial_H(p_1 \cdot t)$. Then we have $\partial_H(s) \xrightarrow{\sigma} \partial_H(p_1)$, so also $\partial_H(s) \cdot \partial_H(t) \xrightarrow{\sigma} \partial_H(p_1) \cdot \partial_H(t)$, and note that $(\partial_H(p_1 \cdot t), \partial_H(p_1) \cdot \partial_H(t)) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose that $\partial_H(s) \cdot \partial_H(t) \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{a} p_1$ and $p \equiv \partial_H(p_1) \cdot \partial_H(t)$. Then we have $\partial_H(s \cdot t) \xrightarrow{a} \partial_H(p_1 \cdot t)$, and note that $(\partial_H(p_1 \cdot t), \partial_H(p_1) \cdot \partial_H(t)) \in R$.
 - (b) $s \xrightarrow{a} \surd$ and $p \equiv \partial_H(t)$. Then we have $\partial_H(s \cdot t) \xrightarrow{a} \partial_H(t)$, and note that $(\partial_H(t), \partial_H(t)) \in R$.
- (ii). Suppose that $\partial_H(s) \cdot \partial_H(t) \xrightarrow{a} \surd$. This case cannot occur.

- (iii). Suppose that $\partial_H(s) \cdot \partial_H(t) \xrightarrow{\sigma} p$. Then necessarily $s \xrightarrow{\sigma} p_1$ and $p \equiv \partial_H(p_1) \cdot \partial_H(t)$. Then we have $\partial_H(s \cdot t) \xrightarrow{\sigma} \partial_H(p_1 \cdot t)$, and note that $(\partial_H(p_1 \cdot t), \partial_H(p_1) \cdot \partial_H(t)) \in R$.

Axiom DRTD5 Take the relation:

$$R = \{(s, s), (\partial_H(\sigma(s)), \sigma(\partial_H(s))) \mid s \in C(\text{ACP}_{\text{drt}}^- \text{-ID})\}$$

We look at the transitions of both sides at the same time. The only possible transition of the left-hand side is $\partial_H(\sigma(s)) \xrightarrow{\sigma} \partial_H(s)$, and the only possible transition of the right-hand side is $\sigma(\partial_H(s)) \xrightarrow{\sigma} \partial_H(s)$, and note that $(\partial_H(s), \partial_H(s)) \in R$.

■

Remark 5.2.3.11 (Soundness of $\text{ACP}_{\text{drt}}^- \text{-ID}$)

Soundness of $\text{ACP}_{\text{drt}}^- \text{-ID}$ is also claimed (without proof) in Theorem 4.2 of BAETEN AND RENIERS [35].

Theorem 5.2.3.12 (Conservativity of $\text{ACP}_{\text{drt}}^- \text{-ID}$ with respect to $\text{BPA}_{\text{drt}}^- \text{-ID}$)

The process algebra $\text{ACP}_{\text{drt}}^- \text{-ID}$ is a conservative extension of the process algebra $\text{BPA}_{\text{drt}}^- \text{-ID}$.

Proof In order to prove conservativity it is sufficient to verify that the following conditions are satisfied:

- (i). Bisimulation equivalence is definable in terms of predicate and relation symbols only,
- (ii). $\text{BPA}_{\text{drt}}^- \text{-ID}$ is a complete axiomatization with respect to the bisimulation equivalence model induced by $T(\text{BPA}_{\text{drt}}^- \text{-ID})$ (see Theorem 4.3.2.6),
- (iii). $\text{ACP}_{\text{drt}}^- \text{-ID}$ is a sound axiomatization with respect to the bisimulation equivalence model induced by $T(\text{ACP}_{\text{drt}}^- \text{-ID})$ (see Theorem 5.2.3.10),
- (iv). $T(\text{ACP}_{\text{drt}}^- \text{-ID})$ is an operationally conservative extension of $T(\text{BPA}_{\text{drt}}^- \text{-ID})$.

And in order for $T(\text{ACP}_{\text{drt}}^- \text{-ID})$ indeed to be an operationally conservative extension of $T(\text{BPA}_{\text{drt}}^- \text{-ID})$ we must verify the following conditions:

- (i). $T(\text{BPA}_{\text{drt}}^- \text{-ID})$ is a pure, well-founded term-deduction system in path format,
- (ii). $T(\text{ACP}_{\text{drt}}^- \text{-ID})$ is a term-deduction system in path format,
- (iii). $T(\text{BPA}_{\text{drt}}^- \text{-ID}) \oplus T(\text{ACP}_{\text{drt}}^- \text{-ID})$ is defined.

That the above properties hold can be trivially checked from the relevant definitions. ■

Theorem 5.2.3.13 (Completeness of $\text{ACP}_{\text{drt}}^- \text{-ID}$)

The axiom system $\text{ACP}_{\text{drt}}^- \text{-ID}$ is a complete axiomatization of the set of closed $\text{ACP}_{\text{drt}}^- \text{-ID}$ terms modulo bisimulation equivalence.

Proof We use Verhoef's method described in Proof Outline 4.2.3.4 on page 71. Completeness then follows immediately from:

- (i). $\text{ACP}_{\text{drt}}^- \text{-ID}$ has the elimination property for $\text{BPA}_{\text{drt}}^- \text{-ID}$ (see Theorem 5.2.3.7),
- (ii). $\text{ACP}_{\text{drt}}^- \text{-ID}$ is a conservative extension of $\text{BPA}_{\text{drt}}^- \text{-ID}$ (see Theorem 5.2.3.12).

■

Remark 5.2.3.14 (Completeness of $ACP_{\text{drt}}^- \text{-ID}$)

Completeness of $ACP_{\text{drt}}^- \text{-ID}$ is also claimed (without proof) in Theorem 4.2 of BAETEN AND RENIERS [35].

5.2.4 $ACP_{\text{drt}}^- \text{-ID}'$

In this section, we modify $PA_{\text{drt}}^- \text{-ID}'$ by extending the free merge to a merge, where the axioms for the communication merge are based on the inductive definition of basic terms. The resulting process algebra is called $ACP_{\text{drt}}^- \text{-ID}'$.

Definition 5.2.4.1 (Signature of $ACP_{\text{drt}}^- \text{-ID}'$)

The signature of $ACP_{\text{drt}}^- \text{-ID}'$ is identical to the signature of $ACP_{\text{drt}}^- \text{-ID}$ as given in Definition 5.2.3.1; it consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , the *“now” operator* ν_{rel} , the *merge operator* \parallel , the *left merge operator* \llcorner , the *communication merge operator* \mid , and the *encapsulation operator* ∂_H .

Definition 5.2.4.2 (Axioms of $ACP_{\text{drt}}^- \text{-ID}'$)

The process algebra $ACP_{\text{drt}}^- \text{-ID}'$ is axiomatized by the axioms of $PA_{\text{drt}}^- \text{-ID}'$ given in Definition 5.2.2.2 on page 113 *minus* Axiom DRTM1, *plus* the Axioms DRTCM1–DRTCM5, DRTCM12–DRTCM13, DRTCF, and DRTD1–DRTD5 shown in Table 5.6 on page 122, and the Axioms DRTCM8–DRTCM11 that are shown in Table 5.10: $ACP_{\text{drt}}^- \text{-ID}' = A1\text{-}A5 + \text{DRT1}\text{-}\text{DRT5} + \text{DCS1}\text{-}\text{DCS4} + \text{DRTM2}\text{-}\text{DRTM4} + \text{DRTM7}\text{-}\text{DRTM11} + \text{DRTCM1}\text{-}\text{DRTCM5} + \text{DRTCM8}\text{-}\text{DRTCM13} + \text{DRTCF} + \text{DRTD1}\text{-}\text{DRTD5}$.

$\underline{a} \mid \sigma_{\text{rel}}(x) = \underline{\delta}$	DRTCM8
$\sigma_{\text{rel}}(x) \mid \underline{a} = \underline{\delta}$	DRTCM9
$\underline{a} \cdot x \mid \sigma_{\text{rel}}(y) = \underline{\delta}$	DRTCM10
$\sigma_{\text{rel}}(x) \mid \underline{a} \cdot y = \underline{\delta}$	DRTCM11

Table 5.10: Additional axioms for $ACP_{\text{drt}}^- \text{-ID}'$.

☞ Axioms DRTCM8–DRTCM11 all express that if one side of a communication merge *must* do its first action in the current time slice and the other side *cannot* do an action in the current time slice, the communication merge collapses to an undelayable deadlock, as no communication between the initial actions of both sides is possible. Note that this is exactly the same motivation we provided for Axioms DRTCM6 and DRTCM7. Intuitively, both groups of axioms are interchangeable.

Definition 5.2.4.3 (Semantics of $ACP_{\text{drt}}^- \text{-ID}'$)

The semantics of $ACP_{\text{drt}}^- \text{-ID}'$ are given by the term-deduction system $T(ACP_{\text{drt}}^- \text{-ID}')$ which is identical to the term-deduction system $T(ACP_{\text{drt}}^- \text{-ID})$ given in Definition 5.2.3.3 on page 122.

Definition 5.2.4.4 (Bisimulation and Bisimulation Model for $ACP_{\text{drt}}^- \text{-ID}'$)

Bisimulation for $ACP_{\text{drt}}^- \text{-ID}'$ and the corresponding bisimulation model are defined in the same way as for $BPA_{\text{drt}}^- \text{-}\delta$ and BPA respectively. Replace “ $BPA_{\text{drt}}^- \text{-}\delta$ ” by “ $ACP_{\text{drt}}^- \text{-ID}'$ ” in Definition 3.2.1.8 on page 45 and “BPA” by “ $ACP_{\text{drt}}^- \text{-ID}'$ ” in Definition 2.3.1.16 on page 12.

Definition 5.2.4.5 (Basic Terms of $ACP_{\text{drt}}^- \text{-ID}'$)

If we speak of basic terms in the context of $ACP_{\text{drt}}^- \text{-ID}'$, we mean $(\sigma, \underline{\delta})$ -basic terms as defined in Definition 3.2.2.6 on page 49.

Definition 5.2.4.6 (Number of Symbols of an $ACP_{\text{drt}}^- \text{-ID}'$ Term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). For $a \in A_\delta$, we define $n(\underline{a}) = 1$,
- (ii). for closed $ACP_{\text{drt}}^- \text{-ID}'$ terms x and y , we define $n(x + y) = n(x \cdot y) = n(x \parallel y) = n(x \perp y) = n(x \mid y) = n(x) + n(y) + 1$,
- (iii). for a closed $ACP_{\text{drt}}^- \text{-ID}'$ term x , we define $n(\sigma(x)) = n(\nu(x)) = n(\partial_H(x)) = n(x) + 1$.

Theorem 5.2.4.7 (Elimination for $ACP_{\text{drt}}^- \text{-ID}'$)

Let t be a closed $ACP_{\text{drt}}^- \text{-ID}'$ term. Then there is a closed $BPA_{\text{drt}}^- \text{-ID}$ term s such that $ACP_{\text{drt}}^- \text{-ID}' \vdash s = t$.

Proof We use the lexicographical path ordering method we described in Proof Outline 4.2.1.1 on page 68. The term-rewriting system used consists of the term-rewriting system for $BPA_{\text{drt}}^- \text{-ID}$ shown in Table 4.2 on page 79, augmented with the additional term-rewriting rules shown in Table 5.11, Table 5.12, and Table 5.13 on page 137. Note that we have added natural number subscripts n to the merge operators, in order to deal with the mutually recursive nature of definition of these operators. For a description of this technique, and a rigorous formal justification of its use, see Theorem 3.2.3 of BAETEN AND VERHOEF [37], and the references given there.

The operator \cdot is assigned the lexicographical status of the first argument, and the following well-founded ordering on constant and function symbols is used:

$$\underline{a} < \sigma < + < \cdot < \partial_H < \perp_n, \mid_n < \parallel_n < \perp_{n+1}, \mid_{n+1} < \dots$$

That the left-hand side of every rewriting rule is bigger than the right-hand side with respect to the ordering \succ_{lpo} , is shown by the following reductions:

$$\begin{aligned} x \parallel_n y &\succ_{\text{lpo}} x \parallel_n^* y \succ_{\text{lpo}} x \parallel_n^* y + x \parallel_n^* y \succ_{\text{lpo}} x \parallel_n^* y + x \parallel_n^* y + x \parallel_n^* y \\ &\succ_{\text{lpo}} (x \parallel_n^* y) \perp_n (x \parallel_n^* y) + (x \parallel_n^* y) \perp_n (x \parallel_n^* y) + (x \parallel_n^* y) \mid_n (x \parallel_n^* y) \\ &\succ_{\text{lpo}} x \perp_n y + y \perp_n x + x \mid_n y \\ \underline{a} \mid_n \underline{b} &\succ_{\text{lpo}} \underline{a} \mid_n^* \underline{b} \\ &\succ_{\text{lpo}} \underline{c} \\ \underline{a} \mid_n \underline{b} \cdot x &\succ_{\text{lpo}} \underline{a} \mid_n^* \underline{b} \cdot x \succ_{\text{lpo}} (\underline{a} \mid_n^* \underline{b} \cdot x) \cdot (\underline{a} \mid_n^* \underline{b} \cdot x) \succ_{\text{lpo}} (\underline{a} \mid_n \underline{b} \cdot^* x) \cdot (\underline{b} \cdot x) \\ &\succ_{\text{lpo}} (\underline{a} \mid_n \underline{b}) \cdot (\underline{b} \cdot x) \succ_{\text{lpo}} (\underline{a} \mid_n \underline{b}) \cdot^* (\underline{b} \cdot x) \succ_{\text{lpo}} (\underline{a} \mid_n \underline{b}) \cdot (\underline{b} \cdot^* x) \\ &\succ_{\text{lpo}} (\underline{a} \mid_n \underline{b}) \cdot x \end{aligned}$$

$\sigma(x) + \sigma(y) \rightarrow \sigma(x + y)$	RDRT1
$x \parallel_n y \rightarrow x \parallel_n y + y \parallel_n x + x \parallel_n y$	RDRTCM1
$\underline{a} \parallel_n x \rightarrow \underline{a} \cdot x$	RDRTM2
$\underline{a} \cdot x \parallel_{n+1} y \rightarrow \underline{a} \cdot (x \parallel_n y)$	RDRTM3
$(x + y) \parallel_n z \rightarrow x \parallel_n z + y \parallel_n z$	RDRTM4
$\sigma(x) \parallel_n \underline{a} \rightarrow \underline{\delta}$	RDRTM7
$\sigma(x) \parallel_n \underline{a} \cdot y \rightarrow \underline{\delta}$	RDRTM8
$\sigma(x) \parallel_n (\underline{a} + y) \rightarrow \sigma(x) \parallel_n y$	RDRTM9
$\sigma(x) \parallel_n (\underline{a} \cdot y + z) \rightarrow \sigma(x) \parallel_n z$	RDRTM10
$\sigma(x) \parallel_n \sigma(y) \rightarrow \sigma(x \parallel_n y)$	RDRTM11

Table 5.11: Additional term-rewriting rules for $ACP_{\text{drt}}^- \text{-ID}'$, Part I.

$$\begin{aligned}
& \underline{a} \cdot x \parallel_n \underline{b} \succ_{\text{lpo}} \underline{a} \cdot x \parallel_n^* \underline{b} \succ_{\text{lpo}} (\underline{a} \cdot x \parallel_n^* \underline{b}) \cdot (\underline{a} \cdot x \parallel_n^* \underline{b}) \succ_{\text{lpo}} (\underline{a} \cdot^* x \parallel_n \underline{b}) \cdot (\underline{a} \cdot x) \\
& \succ_{\text{lpo}} (\underline{a} \parallel_n \underline{b}) \cdot (\underline{a} \cdot x) \succ_{\text{lpo}} (\underline{a} \parallel_n \underline{b}) \cdot^* (\underline{a} \cdot x) \succ_{\text{lpo}} (\underline{a} \parallel_n \underline{b}) \cdot (\underline{a} \cdot^* x) \\
& \succ_{\text{lpo}} (\underline{a} \parallel_n \underline{b}) \cdot x \\
& \underline{a} \cdot x \parallel_{n+1} \underline{b} \cdot y \succ_{\text{lpo}} \underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y \succ_{\text{lpo}} (\underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y) \cdot (\underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y) \\
& \succ_{\text{lpo}} (\underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y) \cdot ((\underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y) \parallel_n (\underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y)) \\
& \succ_{\text{lpo}} (\underline{a} \cdot^* x \parallel_{n+1} \underline{b} \cdot^* y) \cdot ((\underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y) \parallel_n (\underline{a} \cdot x \parallel_{n+1}^* \underline{b} \cdot y)) \\
& \succ_{\text{lpo}} (\underline{a} \parallel_{n+1} \underline{b}) \cdot (\underline{a} \cdot x \parallel_n \underline{b} \cdot y) \succ_{\text{lpo}} (\underline{a} \parallel_{n+1} \underline{b}) \cdot (\underline{a} \cdot^* x \parallel_n \underline{b} \cdot^* y) \\
& \succ_{\text{lpo}} (\underline{a} \parallel_{n+1} \underline{b}) \cdot (x \parallel_n y) \\
& \sigma(x) \parallel_n \sigma(y) \succ_{\text{lpo}} \sigma(x) \parallel_n^* \sigma(y) \succ_{\text{lpo}} \sigma(\sigma(x) \parallel_n^* \sigma(y)) \\
& \succ_{\text{lpo}} \sigma(\sigma \cdot^*(x) \parallel_n \sigma \cdot^*(y)) \succ_{\text{lpo}} \sigma(x \parallel_n y) \\
& \underline{a} \parallel_n \sigma(x) \succ_{\text{lpo}} \underline{a} \parallel_n^* \sigma(x) \\
& \succ_{\text{lpo}} \underline{\delta} \\
& \sigma(x) \parallel_n \underline{a} \succ_{\text{lpo}} \sigma(x) \parallel_n^* \underline{a} \\
& \succ_{\text{lpo}} \underline{\delta} \\
& \underline{a} \cdot x \parallel_n \sigma(y) \succ_{\text{lpo}} \underline{a} \cdot x \parallel_n^* \sigma(y) \\
& \succ_{\text{lpo}} \underline{\delta} \\
& \sigma(x) \parallel_n \underline{a} \cdot y \succ_{\text{lpo}} \sigma(x) \parallel_n^* \underline{a} \cdot y \\
& \succ_{\text{lpo}} \underline{\delta} \\
& (x + y) \parallel_n z \succ_{\text{lpo}} (x + y) \parallel_n^* z \succ_{\text{lpo}} (x + y) \parallel_n^* z + (x + y) \parallel_n^* z \\
& \succ_{\text{lpo}} (x +^* y) \parallel_n z + (x +^* y) \parallel_n z \succ_{\text{lpo}} (x \parallel_n z) + (y \parallel_n z) \\
& x \parallel_n (y + z) \succ_{\text{lpo}} x \parallel_n^* (y + z) \succ_{\text{lpo}} x \parallel_n^* (y + z) + x \parallel_n^* (y + z) \\
& \succ_{\text{lpo}} x \parallel_n (y +^* z) + x \parallel_n (y +^* z) \succ_{\text{lpo}} x \parallel_n (y) + x \parallel_n (z)
\end{aligned}$$

$\underline{a} \mid_n \underline{b} \rightarrow \underline{c}$	if $\gamma(a, b) = c$	RDRTCF
$\underline{a} \mid_n \underline{b} \cdot x \rightarrow (\underline{a} \mid_n \underline{b}) \cdot x$		RDRTCM2
$\underline{a} \cdot x \mid_n \underline{b} \rightarrow (\underline{a} \mid_n \underline{b}) \cdot x$		RDRTCM3
$\underline{a} \cdot x \mid_{n+1} \underline{b} \cdot y \rightarrow (\underline{a} \mid_{n+1} \underline{b}) \cdot (x \parallel_n y)$		RDRTCM4
$\sigma(x) \mid_n \sigma(y) \rightarrow \sigma(x \mid_n y)$		RDRTCM5
$\underline{a} \mid_n \sigma(x) \rightarrow \underline{\delta}$		RDRTCM8
$\sigma(x) \mid_n \underline{a} \rightarrow \underline{\delta}$		RDRTCM9
$\underline{a} \cdot x \mid_n \sigma(y) \rightarrow \underline{\delta}$		RDRTCM10
$\sigma(x) \mid_n \underline{a} \cdot y \rightarrow \underline{\delta}$		RDRTCM11
$(x + y) \mid_n z \rightarrow x \mid_n z + y \mid_n z$		RDRTCM12
$x \mid_n (y + z) \rightarrow x \mid_n y + x \mid_n z$		RDRTCM13

Table 5.12: Additional term-rewriting rules for $\text{ACP}_{\text{drt}}^- \text{-ID}'$, Part II.

$\partial_H(\underline{a}) \rightarrow \underline{a}$	RDRTD1
$\partial_H(\underline{a}) \rightarrow \underline{\delta}$	RDRTD2
$\partial_H(x + y) \rightarrow \partial_H(x) + \partial_H(y)$	RDRTD3
$\partial_H(x \cdot y) \rightarrow \partial_H(x) \cdot \partial_H(y)$	RDRTD4
$\partial_H(\sigma(x)) \rightarrow \sigma(\partial_H(x))$	RDRTD5

Table 5.13: Additional term-rewriting rules for $\text{ACP}_{\text{drt}}^- \text{-ID}'$, Part III.

$\partial_H(\underline{a}) \succ_{\text{lpo}} \partial_H^*(\underline{a})$
$\succ_{\text{lpo}} \underline{a}$
$\partial_H(\underline{a}) \succ_{\text{lpo}} \partial_H^*(\underline{a})$
$\succ_{\text{lpo}} \underline{\delta}$
$\partial_H(x + y) \succ_{\text{lpo}} \partial_H^*(x + y) \succ_{\text{lpo}} \partial_H^*(x + y) + \partial_H^*(x + y) \succ_{\text{lpo}} \partial_H(x + * y) + \partial_H(x + * y)$
$\succ_{\text{lpo}} \partial_H(x) + \partial_H(y)$
$\partial_H(x \cdot y) \succ_{\text{lpo}} \partial_H^*(x \cdot y) \succ_{\text{lpo}} \partial_H^*(x \cdot y) \cdot \partial_H^*(x \cdot y) \succ_{\text{lpo}} \partial_H(x \cdot * y) \cdot \partial_H(x \cdot * y)$
$\succ_{\text{lpo}} \partial_H(x) \cdot \partial_H(y)$
$\partial_H(\sigma(x)) \succ_{\text{lpo}} \partial_H^*(\sigma(x)) \succ_{\text{lpo}} \sigma(\partial_H^*(\sigma(x))) \succ_{\text{lpo}} \sigma(\partial_H(\sigma^*(x)))$
$\succ_{\text{lpo}} \sigma(\partial_H(x))$

Note that we do not give reductions for RDRT1 and RDRTM2–RDRTM11, as these already have been given in the proof of Theorem 5.2.2.7 on page 114, and since the new ordering is a proper extension of the old one, these proofs remain valid.

It remains to prove that every normal form of a closed $ACP_{\text{drt}}^- \text{-ID}'$ term is a closed $BPA_{\text{drt}}^- \text{-ID}$ term. We prove this as follows: suppose that s is a normal form of a closed $ACP_{\text{drt}}^- \text{-ID}'$ term, and furthermore suppose that s is not a closed $BPA_{\text{drt}}^- \text{-ID}$ term. Now consider the smallest subterm s' of s that is not a closed $BPA_{\text{drt}}^- \text{-ID}$ term. Then, s' must be of the form $s' \equiv s_1 \parallel s_2$, of the form $s' \equiv s_1 \perp\!\!\!\perp s_2$, of the form $s' \equiv s_1 | s_2$, or of the form $s' \equiv \partial_H(s_1)$, for closed $BPA_{\text{drt}}^- \text{-ID}$ terms s_1 and s_2 . By the elimination theorem for $BPA_{\text{drt}}^- \text{-ID}$, Theorem 4.3.2.1 on page 79, we may assume that s_1 and s_2 are basic terms. Now in the first case, $s' \equiv s_1 \parallel s_2$, clearly rewriting rule RDRTCM1 is applicable. This contradicts the assumption that s is a normal form, so this case cannot occur. That the second case, $s' \equiv s_1 \perp\!\!\!\perp s_2$, cannot occur is proven in the same way as already done in the proof of the elimination theorem for $PA_{\text{drt}}^- \text{-ID}'$, Theorem 5.2.2.7 on page 114. In the fourth case, $s' \equiv \partial_H(s_1)$, always one of the rewriting rules RDRTD1–RDRTD5 is applicable, so this case cannot occur either. Finally, it remains to derive a contradiction for the third case, $s' \equiv s_1 | s_2$. The following cases can be considered for basic terms s_1 and s_2 (for some $a, b \in A_\delta$ and basic terms s'_1, s''_1, s'_2, s''_2):

- (i). If $s_1 \equiv \underline{a}$ and $s_2 \equiv \underline{b}$ we can apply RDRTCF.
- (ii). If $s_1 \equiv \underline{a}$ and $s_2 \equiv \underline{b} \cdot s'_2$, we can apply RDRTCM2.
- (iii). If $s_1 \equiv \underline{a}$ and $s_2 \equiv \sigma(s'_2)$, we can apply RDRTCM8.
- (iv). If $s_1 \equiv \underline{a} \cdot s'_1$ and $s_2 \equiv \underline{b}$, we can apply RDRTCM3.
- (v). If $s_1 \equiv \underline{a} \cdot s'_1$ and $s_2 \equiv \underline{b} \cdot s'_2$, we can apply RDRTCM4.
- (vi). If $s_1 \equiv \underline{a} \cdot s'_1$ and $s_2 \equiv \sigma(s'_2)$, we can apply RDRTCM10.
- (vii). If $s_1 \equiv \sigma(s'_1)$ and $s_2 \equiv \underline{b}$, we can apply RDRTCM9.
- (viii). If $s_1 \equiv \sigma(s'_1)$ and $s_2 \equiv \underline{b} \cdot s'_2$, we can apply RDRTCM11.
- (ix). If $s_1 \equiv \sigma(s'_1)$ and $s_2 \equiv \sigma(s'_2)$, we can apply RDRTCM5.
- (x). If $s_1 \equiv s'_1 + s''_1$ and s_2 is of an arbitrary form, we can apply RDRTCM12.
- (xi). If s_1 is of an arbitrary form and $s_2 \equiv s'_2 + s''_2$, we can apply RDRTCM13.

This sums up all possible sixteen cases (with seven cases thrown together in (x). and (xi).). In all of these cases we can apply one of the rewriting rules, so s' is not a normal form. This contradicts the assumption that s is a normal form. From this contradiction we conclude that s does not contain a merge operator. Therefore, s must be a closed $BPA_{\text{drt}}^- \text{-ID}$ term, which had to be proven. ■

Corollary 5.2.4.8 (Elimination for $ACP_{\text{drt}}^- \text{-ID}'$)

Let t be a closed $ACP_{\text{drt}}^- \text{-ID}'$ term. Then there is a basic term s such that $ACP_{\text{drt}}^- \text{-ID}' \vdash s = t$.

Proof This follows immediately from:

- (i). The elimination theorem for $ACP_{\text{drt}}^- \text{-ID}'$ (see Theorem 5.2.4.7),
- (ii). the elimination theorem for $BPA_{\text{drt}}^- \text{-ID}$ (see Theorem 4.3.2.1),

(iii). the fact that all axioms of $\text{BPA}_{\text{drt}}^- \text{-ID}$ are also contained in $\text{ACP}_{\text{drt}}^- \text{-ID}'$.

■

Remark 5.2.4.9 (Elimination for $\text{ACP}_{\text{drt}}^- \text{-ID}'$)

Elimination for a slightly different version of $\text{ACP}_{\text{drt}}^- \text{-ID}'$ is also claimed (without proof) in Theorem 3.6.4 of BAETEN AND VERHOEF [37], (where $\text{ACP}_{\text{drt}}^- \text{-ID}'$ is called ACP_{dt}).

Theorem 5.2.4.10 (Axiom Ground Equivalence of $\text{ACP}_{\text{drt}}^- \text{-ID}$ and $\text{ACP}_{\text{drt}}^- \text{-ID}'$)

For all closed $\text{ACP}_{\text{drt}}^- \text{-ID}$ terms s and t we have $\text{ACP}_{\text{drt}}^- \text{-ID} \vdash s = t$ if and only if $\text{ACP}_{\text{drt}}^- \text{-ID}' \vdash s = t$.

Proof It suffices to show that, for closed terms, every axiom of $\text{ACP}_{\text{drt}}^- \text{-ID}'$ is derivable from the axioms of $\text{ACP}_{\text{drt}}^- \text{-ID}$, and vice versa, that, for closed terms, every axiom of $\text{ACP}_{\text{drt}}^- \text{-ID}$ is derivable from the axioms of $\text{ACP}_{\text{drt}}^- \text{-ID}'$. We can restrict ourselves to the axioms that are not shared by both theories. Furthermore, the proofs regarding Axioms DRTM5–DRTM11 that are given in Theorem 5.2.2.13 on page 117, with respect to $\text{PA}_{\text{drt}}^- \text{-ID}$ and $\text{PA}_{\text{drt}}^- \text{-ID}'$, remain valid with respect to $\text{ACP}_{\text{drt}}^- \text{-ID}$ and $\text{ACP}_{\text{drt}}^- \text{-ID}'$.

Part I

First, we show that Axioms DRTCM8–DRTCM11 of $\text{ACP}_{\text{drt}}^- \text{-ID}'$ are derivable in $\text{ACP}_{\text{drt}}^- \text{-ID}$:

Axiom DRTCM8

$$\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \underline{a} \mid \sigma(x) = \underline{\delta}$$

Consider the following derivation:

$$\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \underline{a} \mid \sigma(x) = \nu(\underline{a}) \mid \sigma(x) = \underline{\delta}$$

Axiom DRTCM9

$$\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \mid \underline{a} = \underline{\delta}$$

This axiom is treated symmetrically to the previous axiom.

Axiom DRTCM10

$$\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \underline{a} \cdot x \mid \sigma(y) = \underline{\delta}$$

Consider the following derivation:

$$\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \underline{a} \cdot x \mid \sigma(y) = \nu(\underline{a}) \cdot x \mid \sigma(y) = \nu(\underline{a} \cdot x) \mid \sigma(y) = \underline{\delta}$$

Axiom DRTCM11

$$\text{ACP}_{\text{drt}}^- \text{-ID} \vdash \sigma(x) \mid \underline{a} \cdot y = \underline{\delta}$$

This axiom is treated symmetrically to the previous axiom.

Part II

Secondly, we show that Axioms DRTCM6-DRTCM7 of ACP_{drt}^- -ID are also derivable in ACP_{drt}^- -ID':

Axiom DRTCM6

$$ACP_{\text{drt}}^- \text{-ID}' \vdash \sigma(x) \mid \nu(y) = \underline{\underline{\delta}}$$

Use the general form of basic term y . Take:

$$y \equiv \sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \sigma(u_k)$$

for $m, n, p \in \mathbb{N}$, $a_i, b_j \in A_\delta$, and basic terms t_i and u_k . Then we have:

$$\begin{aligned} ACP_{\text{drt}}^- \text{-ID}' \vdash \sigma(x) \mid \nu(y) &= \\ \sigma(x) \mid \nu \left(\sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \sigma(u_k) \right) &= \\ \sigma(x) \mid \left(\sum_{i < m} \nu(\underline{\underline{a_i}} \cdot t_i) + \sum_{j < n} \nu(\underline{\underline{b_j}}) + \sum_{k < p} \nu(\sigma(u_k)) \right) &= \\ \sigma(x) \mid \left(\sum_{i < m} \nu(\underline{\underline{a_i}}) \cdot t_i + \sum_{j < n} \nu(\underline{\underline{b_j}}) + \sum_{k < p} \nu(\sigma(u_k)) \right) &= \\ \sigma(x) \mid \left(\sum_{i < m} \underline{\underline{a_i}} \cdot t_i + \sum_{j < n} \underline{\underline{b_j}} + \sum_{k < p} \underline{\underline{\delta}} \right) &= \\ \left(\sigma(x) \mid \sum_{i < m} \underline{\underline{a_i}} \cdot t_i \right) + \left(\sigma(x) \mid \sum_{j < n} \underline{\underline{b_j}} \right) + \left(\sigma(x) \mid \sum_{k < p} \underline{\underline{\delta}} \right) &= \\ \left(\sum_{i < m} \sigma(x) \mid \underline{\underline{a_i}} \cdot t_i \right) + \left(\sum_{j < n} \sigma(x) \mid \underline{\underline{b_j}} \right) + \left(\sum_{k < p} \sigma(x) \mid \underline{\underline{\delta}} \right) &= \\ \sum_{i < m} \underline{\underline{\delta}} + \sum_{j < n} \underline{\underline{\delta}} + \sum_{k < p} \underline{\underline{\delta}} &= \\ \underline{\underline{\delta}} & \end{aligned}$$

Axiom DRTCM7

$$ACP_{\text{drt}}^- \text{-ID}' \vdash \nu(x) \mid \sigma(y) = \underline{\underline{\delta}}$$

This axiom is treated symmetrically to the previous axiom. ■

Theorem 5.2.4.11 (Deduction-System Ground Equiv. of ACP_{drt}^- -ID and ACP_{drt}^- -ID')
 ACP_{drt}^- -ID and ACP_{drt}^- -ID' are deduction-system ground equivalent.

Proof Both $T(ACP_{\text{drt}}^- \text{-ID})$ and $T(ACP_{\text{drt}}^- \text{-ID}')$ have the same signature and the same set of deduction rules, so trivially the same equalities hold between closed terms in the respective bisimulation models. ■

Corollary 5.2.4.12 (Soundness of ACP_{drt}^-ID')

The set of all closed ACP_{drt}^-ID' terms modulo bisimulation equivalence is a model of ACP_{drt}^-ID' .

Proof We use the ground equivalence method described in Proof Outline 4.2.2.3 on page 70. The proof then follows immediately from the following observations and Theorem 5.2.2.15:

- (i). The process algebras ACP_{drt}^-ID and ACP_{drt}^-ID' are ground equivalent (see Theorems 5.2.4.10 and 5.2.4.11),
- (ii). Soundness of ACP_{drt}^-ID (see Theorem 5.2.3.10).

■

Remark 5.2.4.13 (Soundness of ACP_{drt}^-ID')

Soundness of a slightly different version of ACP_{drt}^-ID' is also claimed (without proof) in Theorem 3.6.5 of BAETEN AND VERHOEF [37], (where ACP_{drt}^-ID' is called ACP_{dt}).

Corollary 5.2.4.14 (Completeness of ACP_{drt}^-ID')

The axiom system ACP_{drt}^-ID' is a complete axiomatization of the set of closed ACP_{drt}^-ID' terms modulo bisimulation equivalence.

Proof We use the ground equivalence method described in Proof Outline 4.2.3.3 on page 71. The proof then follows immediately from the following observations and Theorem 5.2.2.15:

- (i). The process algebras ACP_{drt}^-ID and ACP_{drt}^-ID' are ground equivalent (see Theorems 5.2.4.10 and 5.2.4.11),
- (ii). Completeness of ACP_{drt}^-ID (see Theorem 5.2.3.13).

■

Remark 5.2.4.15 (Completeness of ACP_{drt}^-ID')

Completeness of a slightly different version of ACP_{drt}^-ID' is also claimed (without proof) in Theorem 3.6.7 of BAETEN AND VERHOEF [37], (where ACP_{drt}^-ID' is called ACP_{dt}).

5.3 Process Algebras with Delayable Actions

We introduce the process algebras PA_{drt}^+ , PA'_{drt} , ACP_{drt}^+ , ACP'_{drt} , and ACP''_{drt} . These are all based on BPA_{drt}^+ , and hence do contain delayable actions and the immediate deadlock.

5.3.1 PA_{drt}^+

In this section, we extend BPA_{drt} to PA_{drt} by introducing axioms for the left merge that take into account the presence of the immediate deadlock. The new axioms are in the σ/ν -style of PA_{drt}^- -ID from Section 5.2.1.

Definition 5.3.1.1 (Signature of PA_{drt})

The signature of PA_{drt} consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *delayable actions* $\{a \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *delayable deadlock constant* δ , the *immediate deadlock constant* $\dot{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , the “now” operator ν_{rel} , the *unbounded start delay operator* $[\]^\omega$, the *free merge operator* \parallel , and the *left merge operator* \ll .

Definition 5.3.1.2 (Axioms of PA_{drt})

The process algebra PA_{drt} is axiomatized by the axioms of BPA_{drt} that are given in Definition 3.2.4.5 on page 55, Axioms DRTM1 and DRTM4 shown in Table 5.1 on page 106, Axiom DRTM6 shown in Table 5.2 on page 106, and finally Axioms DRTM2ID–DRTM3ID, DRTM5ID, and DRTMID1–DRTMID2 shown in Table 5.14: $PA_{\text{drt}} = A1\text{--}A5 + A6\text{ID} + A7\text{ID} + DRT1\text{--}DRT5 + DRTSID + DCS1\text{--}DCS4 + DCSID + ATS + USD + DRTM1 + DRTM2\text{ID}\text{--}DRTM3\text{ID} + DRTM4 + DRTM5\text{ID} + DRTM6 + DRTMID1\text{--}DRTMID2$.

$\underline{a} \ll (x + \underline{\delta}) = \underline{a} \cdot (x + \underline{\delta})$	DRTM2ID
$\underline{a} \cdot x \ll (y + \underline{\delta}) = \underline{a} \cdot (x \parallel (y + \underline{\delta}))$	DRTM3ID
$\sigma_{\text{rel}}(x) \ll (\nu_{\text{rel}}(y) + \underline{\delta}) = \underline{\delta}$	DRTM5ID
$x \ll \dot{\delta} = \dot{\delta}$	DRTMID1
$\dot{\delta} \ll x = \dot{\delta}$	DRTMID2

Table 5.14: Additional axioms for PA_{drt} .

☞ Axioms DRTM2ID, DRTM3ID, and DRTM5ID are weakened versions of Axioms DRTM2 and DRTM3 of Table 5.1 on page 106, and Axiom DRTM5 of Table 5.2 on page 106. In all three cases, the axiom is weakened by adding $\underline{\delta}$ to the right argument of the left merge, in order to avoid the axiom from applying to the case where the right argument is $\dot{\delta}$ (see also the comment on page 27). Note that Axioms DRTM2ID and DRTM3ID can also be viewed as reformulations of their untimed counterparts, Axioms M2ID and M3ID from Table 2.16 on page 26.

Axioms DRTMID1 and DRTMID2 express that an immediate deadlock on either side of a left merge collapses the entire left merge to immediate deadlock. They are identical to Axioms MID1 and MID2 we already encountered in Table 2.16 on page 26.

Definition 5.3.1.3 (Semantics of PA_{drt})

The semantics of PA_{drt} are given by the term-deduction system $T(PA_{\text{drt}})$ induced by the deduction rules for BPA_{drt} given in Definition 3.2.4.9 on page 57 and the deduction rules for the free merge with immediate deadlock shown in Table 5.15 on the facing page.

$\frac{x \xrightarrow{a} x', \neg \text{ID}(y)}{x \parallel y \xrightarrow{a} x' \parallel y}$	$\frac{y \xrightarrow{a} y', \neg \text{ID}(x)}{x \parallel y \xrightarrow{a} x \parallel y'}$	$\frac{x \xrightarrow{a} x', \neg \text{ID}(y)}{x \llcorner y \xrightarrow{a} x' \parallel y}$
$\frac{x \xrightarrow{a} \surd, \neg \text{ID}(y)}{x \parallel y \xrightarrow{a} y}$	$\frac{y \xrightarrow{a} \surd, \neg \text{ID}(x)}{x \parallel y \xrightarrow{a} x}$	$\frac{x \xrightarrow{a} \surd, \neg \text{ID}(y)}{x \llcorner y \xrightarrow{a} y}$
$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \parallel y \xrightarrow{\sigma} x' \parallel y'}$	$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \llcorner y \xrightarrow{\sigma} x' \llcorner y'}$	
$\frac{\text{ID}(x)}{\text{ID}(x \parallel y)}$	$\frac{\text{ID}(y)}{\text{ID}(x \parallel y)}$	
$\frac{\text{ID}(x)}{\text{ID}(x \llcorner y)}$	$\frac{\text{ID}(y)}{\text{ID}(x \llcorner y)}$	

Table 5.15: Deduction rules for free merge with immediate deadlock.

☞ The deduction rules for the free merge in the presence of the immediate deadlock are similar to those for the free merge shown in Table 5.3 on page 107, with the exception that we now require that one side of the free merge or left merge can only execute an action if the ID predicate does not hold for the other side. Furthermore, there are four deduction rules to define the immediate deadlock predicate on the free merge and the left merge.

Definition 5.3.1.4 (Bisimulation and Bisimulation Model for PA_{drt})

Bisimulation for PA_{drt} and the corresponding bisimulation model are defined in the same way as for $\text{BPA}_{\text{drt}}^-$ and BPA respectively. Replace “ $\text{BPA}_{\text{drt}}^-$ ” by “ PA_{drt} ” in Definition 3.2.3.5 on page 53 and “ BPA ” by “ PA_{drt} ” in Definition 2.3.1.16 on page 12.

Definition 5.3.1.5 (Basic Terms of PA_{drt})

If we speak of basic terms in the context of PA_{drt} , we mean $(\sigma, \underline{\delta}, \delta, \dot{\delta})$ -basic terms as defined in Definition 3.2.4.11 on page 57.

Definition 5.3.1.6 (Number of Symbols of a PA_{drt} term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). We define $n(\dot{\delta}) = 1$,
- (ii). for $a \in A_\delta$, we define $n(\underline{a}) = n(a) = 1$,
- (iii). for closed PA_{drt} terms x and y , we define $n(x+y) = n(x \cdot y) = n(x \parallel y) = n(x \llcorner y) = n(x) + n(y) + 1$,
- (iv). for a closed PA_{drt} term x , we define $n(\sigma(x)) = n(\nu(x)) = n(\lfloor x \rfloor^\omega) = n(x) + 1$.

Proposition 5.3.1.7 (Properties of PA_{drt}^+)

For PA_{drt} terms x and y , and any $a \in A_\delta$, we have the following equalities:

- (i). $PA_{drt}^+ \vdash \lfloor a \rfloor^\omega = a$
- (ii). $PA_{drt}^+ \vdash \lfloor x \cdot y \rfloor^\omega = \lfloor x \rfloor^\omega \cdot y$
- (iii). $PA_{drt}^+ \vdash \lfloor x + y \rfloor^\omega = \lfloor x \rfloor^\omega + \lfloor y \rfloor^\omega$
- (iv). $PA_{drt}^+ \vdash \lfloor \sigma(x) \rfloor^\omega = \delta$
- (v). $PA_{drt}^+ \vdash \lfloor \dot{\delta} \rfloor^\omega = \delta$
- (vi). $PA_{drt}^+ \vdash a \ll \lfloor x \rfloor^\omega = a \cdot \lfloor x \rfloor^\omega$
- (vii). $PA_{drt}^+ \vdash a \cdot x \ll \lfloor y \rfloor^\omega = a \cdot (x \ll \lfloor y \rfloor^\omega)$
- (viii). $PA_{drt} \vdash \nu(a) = \underline{a}$
- (ix). $PA_{drt} \vdash \lfloor x \rfloor^\omega + \underline{\delta} = \lfloor x \rfloor^\omega$

Proof The proofs for equality (i)–(v) and (viii)–(ix) given in Proposition 3.2.4.14 on page 58, with respect to BPA_{drt} , remain valid in the setting of PA_{drt} , as can be easily checked.

Equality (vi) and (vii) do not appear in Proposition 3.2.4.14. Consider the following computation for equality (vi):

$$\begin{aligned}
PA_{drt} \vdash a \ll \lfloor x \rfloor^\omega &= \lfloor \underline{a} \rfloor^\omega \ll \lfloor x \rfloor^\omega \\
&= (\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \ll \lfloor x \rfloor^\omega \\
&= (\underline{a} + \sigma(a)) \ll \lfloor x \rfloor^\omega \\
&= \underline{a} \ll \lfloor x \rfloor^\omega + \sigma(a) \ll \lfloor x \rfloor^\omega \\
&= \underline{a} \ll (\lfloor x \rfloor^\omega + \underline{\delta}) + \sigma(a) \ll (\nu(x) + \sigma(\lfloor x \rfloor^\omega)) \\
&= \underline{a} \cdot (\lfloor x \rfloor^\omega + \underline{\delta}) + \sigma(a \ll \lfloor x \rfloor^\omega) \\
&= \nu(\underline{a}) \cdot \lfloor x \rfloor^\omega + \sigma(a \ll \lfloor x \rfloor^\omega) \\
&= \nu(\underline{a} \cdot \lfloor x \rfloor^\omega) + \sigma(a \ll \lfloor x \rfloor^\omega)
\end{aligned}$$

Using RSP(USD) we obtain:

$$PA_{drt}^+ \vdash a \ll \lfloor x \rfloor^\omega = \lfloor \underline{a} \cdot \lfloor x \rfloor^\omega \rfloor^\omega = \lfloor \underline{a} \rfloor^\omega \cdot \lfloor x \rfloor^\omega = a \cdot \lfloor x \rfloor^\omega$$

Finally, consider the following computation for equality (vii):

$$\begin{aligned}
PA_{drt} \vdash a \cdot x \ll \lfloor y \rfloor^\omega &= \lfloor \underline{a} \rfloor^\omega \cdot x \ll \lfloor y \rfloor^\omega \\
&= (\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \cdot x \ll \lfloor y \rfloor^\omega \\
&= (\underline{a} + \sigma(a)) \cdot x \ll \lfloor y \rfloor^\omega \\
&= (\underline{a} \cdot x + \sigma(a \cdot x)) \ll \lfloor y \rfloor^\omega \\
&= \underline{a} \cdot x \ll \lfloor y \rfloor^\omega + \sigma(a \cdot x) \ll \lfloor y \rfloor^\omega \\
&= \underline{a} \cdot x \ll (\lfloor y \rfloor^\omega + \underline{\delta}) + \sigma(a \cdot x) \ll (\nu(y) + \sigma(\lfloor y \rfloor^\omega)) \\
&= \underline{a} \cdot (x \ll (\lfloor y \rfloor^\omega + \underline{\delta})) + \sigma(a \cdot x \ll \lfloor y \rfloor^\omega) \\
&= \nu(\underline{a}) \cdot (x \ll \lfloor y \rfloor^\omega) + \sigma(a \cdot x \ll \lfloor y \rfloor^\omega) \\
&= \nu(\underline{a} \cdot (x \ll \lfloor y \rfloor^\omega)) + \sigma(a \cdot x \ll \lfloor y \rfloor^\omega)
\end{aligned}$$

Using RSP(USD) we obtain:

$$\text{PA}_{\text{drt}}^+ \vdash a \cdot x \ll [y]^\omega = [\underline{a} \cdot (x \ll [y]^\omega)]^\omega = [\underline{a}]^\omega \cdot (x \ll [y]^\omega) = a \cdot (x \ll [y]^\omega)$$

■

☞ The equalities of Proposition 5.3.1.7 on the facing page are needed in the elimination proof for PA_{drt}^+ . By collecting them all into one proposition, we are able to prove elimination without explicitly referring to RSP(USD). This has the advantage that we can keep track of the places *where* RSP(USD) is needed, and *why* it is needed there. Using this knowledge, we introduce, in Section 5.3.2, new (unconditional) axioms to replace RSP(USD), and do so in a systematic way.

Theorem 5.3.1.8 (Elimination for PA_{drt}^+)

Let t be a closed PA_{drt} term. Then there is a closed BPA_{drt} term s such that $\text{PA}_{\text{drt}}^+ \vdash t = s$.

Proof We use the direct method we described in Proof Outline 4.2.1.2 on page 68. Let t be a closed PA_{drt} term. The theorem is proven by induction on $n(t)$ and case distinction on the general structure of t .

- (i). $t \equiv \delta$. Then t is a closed BPA_{drt} term.
- (ii). $t \equiv \underline{a}$ for some $a \in A_\delta$. Then t is a closed BPA_{drt} term.
- (iii). $t \equiv a$ for some $a \in A_\delta$. Then t is a closed BPA_{drt} term.
- (iv). $t \equiv t_1 + t_2$ for closed PA_{drt} terms t_1 and t_2 . By induction there are closed BPA_{drt} terms s_1 and s_2 such that $\text{PA}_{\text{drt}}^+ \vdash t_1 = s_1$ and $\text{PA}_{\text{drt}}^+ \vdash t_2 = s_2$. But then also $\text{PA}_{\text{drt}}^+ \vdash t_1 + t_2 = s_1 + s_2$ and $s_1 + s_2$ is a closed BPA_{drt} term.
- (v). $t \equiv t_1 \cdot t_2$ for closed PA_{drt} terms t_1 and t_2 . This case is treated analogously to case (iv).
- (vi). $t \equiv \sigma(t_1)$ for a closed PA_{drt} term t_1 . This case is treated analogously to case (iv).
- (vii). $t \equiv \nu(t_1)$ for a closed PA_{drt} term t_1 . This case is treated analogously to case (iv).
- (viii). $t \equiv [t_1]^\omega$ for a closed PA_{drt} term t_1 . This case is treated analogously to case (iv).
- (ix). $t \equiv t_1 \ll t_2$ for closed PA_{drt} terms t_1 and t_2 . By induction there are closed BPA_{drt} terms s_1 and s_2 such that $\text{PA}_{\text{drt}}^+ \vdash t_1 = s_1$ and $\text{PA}_{\text{drt}}^+ \vdash t_2 = s_2$. By Theorem 4.3.4.1, the elimination theorem for BPA_{drt} , there are basic terms r_1 and r_2 such that $\text{BPA}_{\text{drt}}^+ \vdash s_1 = r_1$ and $\text{BPA}_{\text{drt}}^+ \vdash s_2 = r_2$. But then also, $\text{PA}_{\text{drt}}^+ \vdash t_1 = r_1$, $\text{PA}_{\text{drt}}^+ \vdash t_2 = r_2$, and $\text{PA}_{\text{drt}}^+ \vdash t_1 \ll t_2 = r_1 \ll r_2$. We proceed by induction on the structure of basic terms, and distinguish all possible cases for basic term r_1 :
 - (a) $r_1 \equiv \delta$. Then $\text{PA}_{\text{drt}}^+ \vdash t_1 \ll t_2 = r_1 \ll r_2 = \delta \ll r_2 = \delta$, and δ is a closed BPA_{drt} term.
 - (b) $r_1 \equiv \underline{a}$ for some $a \in A_\delta$. Using Lemma 3.2.4.19 we distinguish two cases:
 1. $r_2 = \delta$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \ll t_2 = r_1 \ll r_2 = r_1 \ll \delta = \delta$, and δ is a closed BPA_{drt} term.

2. $r_2 = r_2 + \underline{\underline{\delta}}$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{\underline{a}} \parallel (r_2 + \underline{\underline{\delta}}) = \underline{\underline{a}} \cdot (r_2 + \underline{\underline{\delta}}) = \underline{\underline{a}} \cdot r_2$, and $\underline{\underline{a}} \cdot r_2$ is a closed BPA_{drt} term.
- (c) $r_1 \equiv a$ for some $a \in A_\delta$. Using Lemma 3.2.4.18 we distinguish four cases:
1. $r_2 = \underline{\underline{\delta}}$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r_1 \parallel \underline{\underline{\delta}} = \underline{\underline{\delta}}$, and $\underline{\underline{\delta}}$ is a closed BPA_{drt} term.
 2. $r_2 = \nu(r_2) + \underline{\underline{\delta}}$. Then we have:

$$\begin{aligned}
\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
&= a \parallel r_2 \\
&= [\underline{\underline{a}}]^\omega \parallel r_2 \\
&= (\nu(\underline{\underline{a}}) + \sigma([\underline{\underline{a}}]^\omega)) \parallel r_2 \\
&= (\underline{\underline{a}} + \sigma(a)) \parallel r_2 \\
&= \underline{\underline{a}} \parallel r_2 + \sigma(a) \parallel r_2 \\
&= \underline{\underline{a}} \parallel (\nu(r_2) + \underline{\underline{\delta}}) + \sigma(a) \parallel (\nu(r_2) + \underline{\underline{\delta}}) \\
&= \underline{\underline{a}} \cdot (\nu(r_2) + \underline{\underline{\delta}}) + \underline{\underline{\delta}} \\
&= \underline{\underline{a}} \cdot r_2 + \underline{\underline{\delta}} \\
&= \underline{\underline{a}} \cdot r_2 + \underline{\underline{\delta}} \cdot r_2 \\
&= (\underline{\underline{a}} + \underline{\underline{\delta}}) \cdot r_2 \\
&= \underline{\underline{a}} \cdot r_2,
\end{aligned}$$

and $\underline{\underline{a}} \cdot r_2$ is a closed BPA_{drt} term.

3. $r_2 = [r_2]^\omega$. Then, using Proposition 5.3.1.7(vi), we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = a \parallel [r_2]^\omega = a \cdot [r_2]^\omega = a \cdot r_2$, and $a \cdot r_2$ is a closed BPA_{drt} term.
4. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 such that $n(r'_2) < n(r_2)$. Then we have:

$$\begin{aligned}
\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
&= a \parallel r_2 \\
&= [\underline{\underline{a}}]^\omega \parallel r_2 \\
&= (\nu(\underline{\underline{a}}) + \sigma([\underline{\underline{a}}]^\omega)) \parallel r_2 \\
&= (\underline{\underline{a}} + \sigma(a)) \parallel r_2 \\
&= \underline{\underline{a}} \parallel r_2 + \sigma(a) \parallel r_2 \\
&= \underline{\underline{a}} \parallel (\nu(r_2) + \sigma(r'_2)) + \\
&\quad \sigma(a) \parallel (\nu(r_2) + \sigma(r'_2)) \\
&= \underline{\underline{a}} \parallel (\nu(r_2) + \sigma(r'_2) + \underline{\underline{\delta}}) + \sigma(a \parallel r'_2) \\
&= \underline{\underline{a}} \cdot (\nu(r_2) + \sigma(r'_2) + \underline{\underline{\delta}}) + \sigma(a \parallel r'_2) \\
&= \underline{\underline{a}} \cdot (\nu(r_2) + \sigma(r'_2)) + \sigma(a \parallel r'_2) \\
&= \underline{\underline{a}} \cdot r_2 + \sigma(a \parallel r'_2).
\end{aligned}$$

By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{PA}_{\text{drt}}^+ \vdash a \parallel r'_2 = p$. Then, $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{\underline{a}} \cdot r_2 + \sigma(a \parallel r'_2) = \underline{\underline{a}} \cdot r_2 + \sigma(p)$, and $\underline{\underline{a}} \cdot r_2 + \sigma(p)$ is a closed BPA_{drt} term.

(d) $r_1 \equiv \underline{a} \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Using Lemma 3.2.4.19 we distinguish two cases:

1. $r_2 = \dot{\delta}$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r_1 \parallel \dot{\delta} = \dot{\delta}$, and $\dot{\delta}$ is a closed BPA_{drt} term.
2. $r_2 = r_2 + \underline{\delta}$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{a} \cdot r'_1 \parallel (r_2 + \underline{\delta}) = \underline{a} \cdot (r'_1 \parallel (r_2 + \underline{\delta})) = \underline{a} \cdot (r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{PA}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) = \underline{a} \cdot p$, and $\underline{a} \cdot p$ is a closed BPA_{drt} term.

(e) $r_1 \equiv a \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Using Lemma 3.2.4.18 we distinguish four cases:

1. $r_2 = \dot{\delta}$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r_1 \parallel \dot{\delta} = \dot{\delta}$, and $\dot{\delta}$ is a closed BPA_{drt} term.
2. $r_2 = \nu(r_2) + \underline{\delta}$. Then we have:

$$\begin{aligned}
\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
&= a \cdot r'_1 \parallel r_2 \\
&= \lfloor \underline{a} \rfloor^\omega \cdot r'_1 \parallel r_2 \\
&= (\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \cdot r'_1 \parallel r_2 \\
&= (\underline{a} + \sigma(a)) \cdot r'_1 \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a) \cdot r'_1) \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a \cdot r'_1)) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel r_2 + \sigma(a \cdot r'_1) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel (\nu(r_2) + \underline{\delta}) + \sigma(a \cdot r'_1) \parallel (\nu(r_2) + \underline{\delta}) \\
&= \underline{a} \cdot (r'_1 \parallel (\nu(r_2) + \underline{\delta})) + \underline{\delta} \\
&= \underline{a} \cdot (r'_1 \parallel r_2) + \underline{\delta} \\
&= \underline{a} \cdot (r'_1 \parallel r_2) + \underline{\delta} \cdot (r'_1 \parallel r_2) \\
&= (\underline{a} + \underline{\delta}) \cdot (r'_1 \parallel r_2) \\
&= \underline{a} \cdot (r'_1 \parallel r_2).
\end{aligned}$$

By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{PA}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) = \underline{a} \cdot p$, and $\underline{a} \cdot p$ is a closed BPA_{drt} term.

3. $r_2 = \lfloor r_2 \rfloor^\omega$. Then, using Proposition 5.3.1.7(vii), we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = a \cdot r'_1 \parallel \lfloor r_2 \rfloor^\omega = a \cdot (r'_1 \parallel \lfloor r_2 \rfloor^\omega) = a \cdot (r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{PA}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = a \cdot (r'_1 \parallel r_2) = a \cdot p$, and $a \cdot p$ is a closed BPA_{drt} term.
4. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 such that $n(r'_2) < n(r_2)$. Then we have:

$$\begin{aligned}
\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
&= a \cdot r'_1 \parallel r_2 \\
&= \lfloor \underline{a} \rfloor^\omega \cdot r'_1 \parallel r_2 \\
&= (\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \cdot r'_1 \parallel r_2
\end{aligned}$$

$$\begin{aligned}
&= (\underline{a} + \sigma(a)) \cdot r'_1 \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a) \cdot r'_1) \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a \cdot r'_1)) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel r_2 + \sigma(a \cdot r'_1) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel (\nu(r_2) + \sigma(r'_2)) + \\
&\quad \sigma(a \cdot r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) \\
&= \underline{a} \cdot r'_1 \parallel (\nu(r_2) + \sigma(r'_2) + \underline{\delta}) + \\
&\quad \sigma(a \cdot r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) \\
&= \underline{a} \cdot (r'_1 \parallel (\nu(r_2) + \sigma(r'_2) + \underline{\delta})) + \sigma(a \cdot r'_1 \parallel r'_2) \\
&= \underline{a} \cdot (r'_1 \parallel (\nu(r_2) + \sigma(r'_2))) + \sigma(r_1 \parallel r'_2) \\
&= \underline{a} \cdot (r'_1 \parallel r_2) + \sigma(r_1 \parallel r'_2).
\end{aligned}$$

By the induction hypothesis there exist closed BPA_{drt} terms p_1 and p_2 such that $\text{PA}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p_1$ and $\text{PA}_{\text{drt}}^+ \vdash r_1 \parallel r'_2 = p_2$. Then, $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) + \sigma(r_1 \parallel r'_2) = \underline{a} \cdot p_1 + \sigma(p_2)$, and $\underline{a} \cdot p_1 + \sigma(p_2)$ is a closed BPA_{drt} term.

- (f) $r_1 \equiv r'_1 + r''_1$ for basic terms r'_1 and r''_1 . Then $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = (r'_1 + r''_1) \parallel r_2 = r'_1 \parallel r_2 + r''_1 \parallel r_2$. By induction there exist closed BPA_{drt} terms p_1 and p_2 such that $\text{PA}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p_1$ and $\text{PA}_{\text{drt}}^+ \vdash r''_1 \parallel r_2 = p_2$. Then also $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r'_1 \parallel r_2 + r''_1 \parallel r_2 = p_1 + p_2$, and $p_1 + p_2$ is a closed BPA_{drt} term.
- (g) $r_1 \equiv \sigma(r'_1)$ for a basic term r'_1 . Using Lemma 3.2.4.18 we distinguish four cases:
1. $r_2 = \dot{\delta}$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel \dot{\delta} = \dot{\delta}$, and $\dot{\delta}$ is a closed BPA_{drt} term.
 2. $r_2 = \nu(r_2) + \underline{\delta}$. Then $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel (\nu(r_2) + \underline{\delta}) = \underline{\delta}$, and $\underline{\delta}$ is a closed BPA_{drt} term.
 3. $r_2 = \lfloor r_2 \rfloor^\omega$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel \lfloor r_2 \rfloor^\omega = \sigma(r'_1) \parallel (\nu(r_2) + \sigma(\lfloor r_2 \rfloor^\omega)) = \sigma(r'_1 \parallel \lfloor r_2 \rfloor^\omega) = \sigma(r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{PA}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \sigma(r'_1 \parallel r_2) = \sigma(p)$, and $\sigma(p)$ is a closed BPA_{drt} term.
 4. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 such that $n(r'_2) < n(r_2)$. Then we have: $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) = \sigma(r'_1 \parallel r'_2)$. By the induction hypothesis there is a closed BPA_{drt} term p such that $\text{PA}_{\text{drt}}^+ \vdash r'_1 \parallel r'_2 = p$. Then, $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \sigma(r'_1 \parallel r'_2) = \sigma(p)$, and $\sigma(p)$ is a closed BPA_{drt} term.
- (x). $t \equiv t_1 \parallel t_2$ for closed PA_{drt} terms t_1 and t_2 . Then $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1$. By (ix) there are closed BPA_{drt} terms p_1 and p_2 such that $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = p_1$ and $\text{PA}_{\text{drt}}^+ \vdash t_2 \parallel t_1 = p_2$. But then also $\text{PA}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1 = p_1 + p_2$, and $p_1 + p_2$ is a closed BPA_{drt} term. ■

Corollary 5.3.1.9 (Elimination for PA_{drt}^+)

Let t be a closed PA_{drt} term. Then there is a basic term s such that $PA_{drt}^+ \vdash s = t$.

Proof This follows immediately from:

- (i). The elimination theorem for PA_{drt}^+ (see Theorem 5.3.1.8),
- (ii). the elimination theorem for BPA_{drt}^+ (see Theorem 4.3.4.1),
- (iii). the fact that all axioms of BPA_{drt}^+ are also contained in PA_{drt}^+ .

■

Remark 5.3.1.10 (Elimination for PA_{drt})

Elimination for a somewhat different version of PA_{drt} is also claimed (without proof) in Section 3.9 of BAETEN AND BERGSTRA [24].

Theorem 5.3.1.11 (Soundness of PA_{drt}^+)

The set of closed PA_{drt} terms modulo bisimulation equivalence is a model of PA_{drt}^+ .

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. We only prove soundness for the axioms of PA_{drt} that have not been treated in earlier soundness proofs. Note that to extend these proofs to PA_{drt} , we have to check that the bisimulations given in previous soundness proofs respect the ID predicate (as required by transfer condition (iv.) in Definition 3.2.3.5 on page 53). However, as the fact that they do can be easily checked, we will not give details.

Axiom DRTM2ID Take the relation:

$$R = \{(s, s), (\underline{a} \parallel (s + \underline{\delta}), \underline{a} \cdot (s + \underline{\delta})) \mid s \in C(PA_{drt})\}$$

We look at the transitions of both sides at the same time. The only transition of the left-hand side is $\underline{a} \parallel (s + \underline{\delta}) \xrightarrow{a} s + \underline{\delta}$, and the only transition of the right-hand side is $\underline{a} \cdot (s + \underline{\delta}) \xrightarrow{a} s + \underline{\delta}$, and note that $(s + \underline{\delta}, s + \underline{\delta}) \in R$. Finally, neither side satisfies the ID predicate: $\neg ID(\underline{a} \parallel (s + \underline{\delta}))$ and $\neg ID(\underline{a} \cdot (s + \underline{\delta}))$ (note that $\neg ID(s + \underline{\delta})$ even if $ID(s)$).

Axiom DRTM3ID Take the relation:

$$R = \{(s, s), (\underline{a} \cdot s \parallel (t + \underline{\delta}), \underline{a} \cdot (s \parallel (t + \underline{\delta}))) \mid s, t \in C(PA_{drt})\}$$

We look at the transitions of both sides at the same time. The only transition of the left-hand side is $\underline{a} \cdot s \parallel (t + \underline{\delta}) \xrightarrow{a} s \parallel (t + \underline{\delta})$, and the only transition of the right-hand side is $\underline{a} \cdot (s \parallel (t + \underline{\delta})) \xrightarrow{a} s \parallel (t + \underline{\delta})$, and note that $(s \parallel (t + \underline{\delta}), s \parallel (t + \underline{\delta})) \in R$. Finally, neither side satisfies the ID predicate: $\neg ID(\underline{a} \cdot s \parallel (t + \underline{\delta}))$ and $\neg ID(\underline{a} \cdot (s \parallel (t + \underline{\delta})))$.

Axiom DRTM5ID Take the relation:

$$R = \{(\sigma(s) \parallel (\nu(t) + \underline{\delta}), \underline{\delta}) \mid s, t \in C(PA_{drt})\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $\sigma(s) \parallel (\nu(t) + \underline{\delta}) \nrightarrow$. Also for the right-hand side there are no transitions possible: $\underline{\delta} \nrightarrow$. Finally, neither side satisfies the ID predicate: $\neg ID(\sigma(s) \parallel (\nu(t) + \underline{\delta}))$ and $\neg ID(\underline{\delta})$ (note that $\neg ID(\nu(t) + \underline{\delta})$ even if $ID(t)$).

Axiom DRTMID1 Take the relation:

$$R = \{(s \parallel \dot{\delta}, \dot{\delta}) \mid s \in C(\text{PA}_{\text{drt}})\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $s \not\parallel \dot{\delta} \rightarrow$. Also for the right-hand side there are no transitions possible: $\dot{\delta} \not\rightarrow$. Finally, both sides satisfy the ID predicate: $\text{ID}(s \parallel \dot{\delta})$ and $\text{ID}(\dot{\delta})$ (note that $\text{ID}(s \parallel \dot{\delta})$ even if $\neg \text{ID}(s)$).

Axiom DRTMID2 Take the relation:

$$R = \{(\dot{\delta}, s \parallel \dot{\delta}) \mid s \in C(\text{PA}_{\text{drt}})\}$$

This case is treated symmetrically to the previous case. ■

Remark 5.3.1.12 (Soundness of PA_{drt})

Soundness of a somewhat different version of PA_{drt} is also claimed (without proof) in Section 3.9 of BAETEN AND BERGSTRA [24].

Theorem 5.3.1.13 (Conservativity of PA_{drt}^+ with respect to $\text{BPA}_{\text{drt}}^+$)

The process algebra PA_{drt}^+ is a conservative extension of the process algebra $\text{BPA}_{\text{drt}}^+$.

Proof In order to prove conservativity it is sufficient to verify that the following conditions are satisfied:

- (i). Bisimulation equivalence is definable in terms of predicate and relation symbols only,
- (ii). $\text{BPA}_{\text{drt}}^+$ is a complete axiomatization with respect to the bisimulation equivalence model induced by $T(\text{BPA}_{\text{drt}})$ (see Theorem 4.3.4.9),
- (iii). PA_{drt}^+ is a sound axiomatization with respect to the bisimulation equivalence model induced by $T(\text{PA}_{\text{drt}})$ (see Theorem 5.3.1.11),
- (iv). $T(\text{PA}_{\text{drt}})$ is an operationally conservative extension of $T(\text{BPA}_{\text{drt}})$.

And in order for $T(\text{PA}_{\text{drt}})$ to be an operationally conservative extension of $T(\text{BPA}_{\text{drt}})$ we must verify the following conditions:

- (i). $T(\text{BPA}_{\text{drt}})$ is a pure, well-founded term-deduction system in path format,
- (ii). $T(\text{PA}_{\text{drt}})$ is a term-deduction system in path format,
- (iii). $T(\text{BPA}_{\text{drt}}) \oplus T(\text{PA}_{\text{drt}})$ is defined.

That the above properties hold can be trivially checked from the relevant definitions. ■

Theorem 5.3.1.14 (Completeness of PA_{drt}^+)

The axiom system PA_{drt}^+ is a complete axiomatization of the set of closed PA_{drt} terms modulo bisimulation equivalence.

Proof We use Verhoef's method described in Proof Outline 4.2.3.4 on page 71. Completeness then follows immediately from:

- (i). PA_{drt}^+ has the elimination property for BPA_{drt} (see Theorem 5.3.1.8),
- (ii). PA_{drt}^+ is a conservative extension of BPA_{drt}^+ (see Theorem 5.3.1.13).

■

Remark 5.3.1.15 (Completeness of PA_{drt})

Completeness of a somewhat different version of PA_{drt} is also claimed (without proof) in Section 3.9 of BAETEN AND BERGSTRA [24].

5.3.2 PA'_{drt}

In this section, we define a process algebra named PA'_{drt} , that is almost identical to PA_{drt} , except that it has seven additional axioms. These are chosen such that elimination, soundness, and completeness results for PA'_{drt} follow as corollaries from the corresponding results for PA_{drt}^+ .

Definition 5.3.2.1 (Axioms of PA'_{drt})

The process algebra PA'_{drt} is axiomatized by the axioms of PA_{drt} that are given in Definition 5.3.1.2 on page 142, Axioms USD1–USD5 shown in Table 4.5 on page 102, and Axioms USD6–USD7 shown in Table 5.16: $PA'_{drt} = A1-A5 + A6ID + A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD + DRTM1 + DRTM2ID-DRTM3ID + DRTM4 + DRTM5ID + DRTM6 + DRTMID1-DRTMID2 + USD1-USD7$.

$$\begin{array}{ll} a \parallel [x]^\omega = a \cdot [x]^\omega & \text{USD6} \\ a \cdot x \parallel [y]^\omega = a \cdot (x \parallel [y]^\omega) & \text{USD7} \end{array}$$

Table 5.16: Additional axioms for unbounded start delay and left merge.

⇒ Axioms USD1–USD7 precisely correspond to equalities (i)–(vii) of Proposition 5.3.1.7 on page 144. In this way, we obtain an axiomatization that is in many ways (elimination, soundness, completeness) like PA_{drt}^+ , but is also purely equational, i.e., does not contain conditional axioms or recursion principles.

Definition 5.3.2.2 (Signature, Semantics, and Basic Terms of PA'_{drt})

The signature, semantics, bisimulation, bisimulation model, and basic terms of PA'_{drt} are the same as those of PA_{drt} .

Corollary 5.3.2.3 (Elimination for PA'_{drt})

Let t be a closed PA'_{drt} term. Then there is a basic term s such that $PA'_{drt} \vdash t = s$.

Proof In the same way as Theorem 5.3.1.8 and Corollary 5.3.1.9. Note that all equalities of Proposition 5.3.1.7 that are used in the proof of Theorem 5.3.1.8 correspond to derivable equalities in PA'_{drt} . ■

Corollary 5.3.2.4 (Soundness of PA'_{drt})

The set of closed PA'_{drt} terms modulo bisimulation equivalence is a model of PA'_{drt} .

Proof We use the indirect method of Proof Outline 4.2.2.2 on page 70. The result follows directly from the soundness of PA_{drt}^+ (see Theorem 5.3.1.11 on page 149) combined with the fact that Axioms USD1-USD7 are derivable in PA_{drt}^+ (see Proposition 5.3.1.7 on page 144). ■

Corollary 5.3.2.5 (Completeness of PA'_{drt})

The axiom system PA'_{drt} is a complete axiomatization of the set of closed PA'_{drt} terms modulo bisimulation equivalence.

Proof We use the indirect method of Proof Outline 4.2.3.2 on page 71. Careful inspection of the dependencies between the proofs in this section reveals that the proof of Theorem 5.3.1.14 only relies upon RSP(USD) to ensure Proposition 5.3.1.7(i)-(vii). So, we obviously do not need RSP(USD) anymore if we add the corresponding Axioms USD1-USD7, and the result follows. ■

5.3.3 ACP_{drt}^+

In this section, we modify PA_{drt} by extending the free merge to a merge, where the axioms for the communication merge are in the σ/ν -style of ACP_{drt}^- -ID, but take into account the presence of the immediate deadlock. The resulting process algebra is called ACP_{drt} .

Definition 5.3.3.1 (Signature of ACP_{drt})

The signature of ACP_{drt} consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *delayable actions* $\{a \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *delayable deadlock constant* δ , the *immediate deadlock constant* $\hat{\delta}$, the *alternative composition operator* $+$, the *sequential composition operator* $;$, the *time-unit delay operator* σ_{rel} , the *“now” operator* ν_{rel} , the *unbounded start delay operator* $\lfloor \rfloor^\omega$, the *merge operator* \parallel , the *left merge operator* $\lfloor\!\!\lfloor$, the *communication merge operator* \mid , and the *encapsulation operator* ∂_H .

Definition 5.3.3.2 (Axioms of ACP_{drt})

The process algebra ACP_{drt} is axiomatized by the axioms of PA_{drt} that are given in Definition 5.3.1.2 on page 142 *minus* Axiom DRTM1, *plus* Axioms DRTCM1-DRTCM5, DRTCM12-DRTCM13, DRTCF, and DRTD1-DRTD5 shown in Table 5.6 on page 122, and the Axioms DRTMID3-DRTMID4, DRTCM6ID-DRTCM7ID, and DRTD6 shown in Table 5.17 on the next page: $ACP_{drt} = A1-A5 + A6ID + A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD + DRTM2ID-DRTM3ID + DRTM4 + DRTM5ID + DRTM6 + DRTCM1-DRTCM5 + DRTCM6ID-DRTCM7ID + DRTCM12-DRTCM13 + DRTCF + DRTD1-DRTD6 + DRTMID1-DRTMID4$.

$\sigma_{\text{rel}}(x) \mid (\nu_{\text{rel}}(y) + \underline{\underline{\delta}}) = \underline{\underline{\delta}}$	DRTCM6ID
$(\nu_{\text{rel}}(x) + \underline{\underline{\delta}}) \mid \sigma_{\text{rel}}(y) = \underline{\underline{\delta}}$	DRTCM7ID
$x \mid \dot{\delta} = \dot{\delta}$	DRTMID3
$\dot{\delta} \mid x = \dot{\delta}$	DRTMID4
$\partial_H(\dot{\delta}) = \dot{\delta}$	DRTD6

Table 5.17: Additional axioms for ACP_{drt} .

☞ Axioms DRTCM6ID and DRTCM7ID are weakened versions of Axioms DRTCM6 and DRTCM7 of Table 5.6 on page 122. In both cases, the axiom is weakened by adding $\underline{\underline{\delta}}$ to the argument of the communication merge that cannot do a time step, in order to avoid the axiom from applying to the case where that argument is $\dot{\delta}$ (see also the comment on page 27).

Axioms DRTMID3 and DRTMID4 express that an immediate deadlock on either side of a communication merge collapses the entire communication merge to immediate deadlock, and Axiom DRTD6 expresses that the immediate deadlock cannot be encapsulated. They are identical to Axioms MID3, MID4, and D6 we already encountered in Table 2.24 on page 34.

Definition 5.3.3.3 (Semantics of ACP_{drt})

The semantics of ACP_{drt} are given by the term-deduction system $T(\text{ACP}_{\text{drt}})$ induced by the deduction rules for PA_{drt} given in Definition 5.3.1.3 on page 142, the deduction rules for the communication merge shown in Table 5.8 on page 123, and the additional deduction rules for ACP_{drt} shown in Table 5.18.

$\frac{\text{ID}(x)}{\text{ID}(x \mid y)}$	$\frac{\text{ID}(y)}{\text{ID}(x \mid y)}$	$\frac{\text{ID}(x)}{\text{ID}(\partial_H(x))}$
--	--	---

Table 5.18: Additional deduction rules for ACP_{drt} .

☞ We only introduce three new deduction rules in order to define the immediate-deadlock predicate on the communication merge and the encapsulation.

Definition 5.3.3.4 (Bisimulation and Bisimulation Model for ACP_{drt})

Bisimulation for ACP_{drt} and the corresponding bisimulation model are defined in the same way as for $\text{BPA}_{\text{drt}}^-$ and BPA respectively. Replace “ $\text{BPA}_{\text{drt}}^-$ ” by “ ACP_{drt} ” in Definition 3.2.3.5 on page 53 and “ BPA ” by “ ACP_{drt} ” in Definition 2.3.1.16 on page 12.

Definition 5.3.3.5 (Basic Terms of ACP_{drt})

If we speak of basic terms in the context of ACP_{drt} , we mean $(\sigma, \underline{\underline{\delta}}, \delta, \dot{\delta})$ -basic terms as defined in Definition 3.2.4.11 on page 57.

Definition 5.3.3.6 (Number of Symbols of an ACP_{drt} term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). We define $n(\delta) = 1$,
- (ii). for $a \in A_\delta$, we define $n(\underline{a}) = n(a) = 1$,
- (iii). for closed ACP_{drt} terms x and y , we define $n(x+y) = n(x \cdot y) = n(x \parallel y) = n(x \parallel\!\!\! \perp y) = n(x \mid y) = n(x) + n(y) + 1$,
- (iv). for a closed ACP_{drt} term x , we define $n(\sigma(x)) = n(\nu(x)) = n(\lfloor x \rfloor^\omega) = n(\partial_H(x)) = n(x) + 1$.

Proposition 5.3.3.7 (Properties of ACP_{drt}^+ , Part I)

For ACP_{drt} terms x and y , and any $a \in A_\delta$, we have the following equalities:

- (i). $ACP_{\text{drt}}^+ \vdash \lfloor a \rfloor^\omega = a$
- (ii). $ACP_{\text{drt}}^+ \vdash \lfloor x \cdot y \rfloor^\omega = \lfloor x \rfloor^\omega \cdot y$
- (iii). $ACP_{\text{drt}}^+ \vdash \lfloor x + y \rfloor^\omega = \lfloor x \rfloor^\omega + \lfloor y \rfloor^\omega$
- (iv). $ACP_{\text{drt}}^+ \vdash \lfloor \sigma(x) \rfloor^\omega = \delta$
- (v). $ACP_{\text{drt}}^+ \vdash \lfloor \delta \rfloor^\omega = \delta$
- (vi). $ACP_{\text{drt}}^+ \vdash a \parallel\!\!\! \perp \lfloor x \rfloor^\omega = a \cdot \lfloor x \rfloor^\omega$
- (vii). $ACP_{\text{drt}}^+ \vdash a \cdot x \parallel\!\!\! \perp \lfloor y \rfloor^\omega = a \cdot (x \parallel\!\!\! \perp \lfloor y \rfloor^\omega)$
- (viii). $ACP_{\text{drt}} \vdash \nu(a) = \underline{a}$
- (ix). $ACP_{\text{drt}} \vdash \lfloor x \rfloor^\omega + \underline{\delta} = \lfloor x \rfloor^\omega$

Proof The proofs for these equalities given in Proposition 5.3.1.7 on page 144, with respect to PA_{drt} , remain valid in the setting of ACP_{drt} , as can be easily checked. ■

Proposition 5.3.3.8 (Properties of ACP_{drt}^+ , Part II)

For ACP_{drt} terms x and y and any $a, b, c \in A_\delta$, we have the following equalities:

- (i). $ACP_{\text{drt}}^+ \vdash a \mid b = c$ if $\gamma(a, b) = c$
- (ii). $ACP_{\text{drt}}^+ \vdash a \mid b \cdot x = (a \mid b) \cdot x$
- (iii). $ACP_{\text{drt}}^+ \vdash a \cdot x \mid b = (a \mid b) \cdot x$
- (iv). $ACP_{\text{drt}}^+ \vdash a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel\!\!\! \perp y)$
- (v). $ACP_{\text{drt}}^+ \vdash \partial_H(a) = a$ if $a \notin H$
- (vi). $ACP_{\text{drt}}^+ \vdash \partial_H(a) = \delta$ if $a \in H$

Proof

(i). Consider the following computation:

$$\begin{aligned}
\text{ACP}_{\text{drt}} \vdash a \mid b &= [\underline{a}]^\omega \mid [\underline{b}]^\omega \\
&= (\nu(\underline{a}) + \sigma([\underline{a}]^\omega)) \mid (\nu(\underline{b}) + \sigma([\underline{b}]^\omega)) \\
&= (\underline{a} + \sigma(a)) \mid (\underline{b} + \sigma(b)) \\
&= \underline{a} \mid \underline{b} + \underline{a} \mid \sigma(b) + \sigma(a) \mid \underline{b} + \sigma(a) \mid \sigma(b) \\
&= \underline{a} \mid \underline{b} + \nu(\underline{a}) \mid \sigma(b) + \sigma(a) \mid \nu(\underline{b}) + \sigma(a) \mid \sigma(b) \\
&= \underline{c} + \underline{\delta} + \underline{\delta} + \sigma(a \mid b) \\
&= \underline{c} + \sigma(a \mid b) \\
&= \nu(\underline{c}) + \sigma(a \mid b)
\end{aligned}$$

Using RSP(USD), we obtain:

$$\text{ACP}_{\text{drt}}^+ \vdash a \mid b = [\underline{c}]^\omega = c$$

(ii). Consider the following computation:

$$\begin{aligned}
\text{ACP}_{\text{drt}} \vdash a \mid b \cdot x &= [\underline{a}]^\omega \mid [\underline{b}]^\omega \cdot x \\
&= (\nu(\underline{a}) + \sigma([\underline{a}]^\omega)) \mid (\nu(\underline{b}) + \sigma([\underline{b}]^\omega)) \cdot x \\
&= (\underline{a} + \sigma(a)) \mid (\underline{b} + \sigma(b)) \cdot x \\
&= (\underline{a} + \sigma(a)) \mid (\underline{b} \cdot x + \sigma(b) \cdot x) \\
&= \underline{a} \mid \underline{b} \cdot x + \underline{a} \mid \sigma(b) \cdot x + \\
&\quad \sigma(a) \mid \underline{b} \cdot x + \sigma(a) \mid \sigma(b) \cdot x \\
&= \underline{a} \mid \underline{b} \cdot x + \nu(\underline{a}) \mid \sigma(b) \cdot x + \\
&\quad \sigma(a) \mid \nu(\underline{b}) \cdot x + \sigma(a) \mid \sigma(b) \cdot x \\
&= \underline{a} \mid \underline{b} \cdot x + \nu(\underline{a}) \mid \sigma(b \cdot x) + \\
&\quad \sigma(a) \mid \nu(\underline{b} \cdot x) + \sigma(a) \mid \sigma(b \cdot x) \\
&= \underline{\gamma(a, b)} \cdot x + \underline{\delta} + \underline{\delta} + \sigma(a \mid b \cdot x) \\
&= \underline{\gamma(a, b)} \cdot x + \sigma(a \mid b \cdot x) \\
&= \nu(\underline{\gamma(a, b)}) \cdot x + \sigma(a \mid b \cdot x) \\
&= \nu(\underline{\underline{\gamma(a, b)}} \cdot x) + \sigma(a \mid b \cdot x)
\end{aligned}$$

Using RSP(USD), we obtain:

$$\begin{aligned}
\text{ACP}_{\text{drt}}^+ \vdash a \mid b \cdot x &= [\underline{\underline{\gamma(a, b)}} \cdot x]^\omega = [\underline{\underline{\gamma(a, b)}}]^\omega \cdot x = \underline{\underline{\gamma(a, b)}} \cdot x \\
&= (\underline{a} \mid \underline{b}) \cdot x
\end{aligned}$$

(iii). This case is treated symmetrically to the previous case.

(iv). Consider the following computation:

$$\begin{aligned}
\text{ACP}_{\text{drt}} \vdash a \cdot x \mid b \cdot y &= [\underline{a}]^\omega \cdot x \mid [\underline{b}]^\omega \cdot y \\
&= (\nu(\underline{a}) + \sigma([\underline{a}]^\omega)) \cdot x \mid (\nu(\underline{b}) + \sigma([\underline{b}]^\omega)) \cdot y \\
&= (\underline{a} + \sigma(a)) \cdot x \mid (\underline{b} + \sigma(b)) \cdot y \\
&= (\underline{a} \cdot x + \sigma(a) \cdot x) \mid (\underline{b} \cdot y + \sigma(b) \cdot y) \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y + \underline{a} \cdot x \mid \sigma(b) \cdot y + \\
&\quad \sigma(a) \cdot x \mid \underline{b} \cdot y + \sigma(a) \cdot x \mid \sigma(b) \cdot y \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y + \nu(\underline{a}) \cdot x \mid \sigma(b) \cdot y + \\
&\quad \sigma(a) \cdot x \mid \nu(\underline{b}) \cdot y + \sigma(a) \cdot x \mid \sigma(b) \cdot y \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y + \nu(\underline{a} \cdot x) \mid \sigma(b \cdot y) + \\
&\quad \sigma(a \cdot x) \mid \nu(\underline{b} \cdot y) + \sigma(a \cdot x) \mid \sigma(b \cdot y) \\
&= \underline{\gamma(a, b)} \cdot (x \parallel y) + \underline{\delta} + \underline{\delta} + \sigma(a \cdot x \mid b \cdot y) \\
&= \underline{\gamma(a, b)} \cdot (x \parallel y) + \sigma(a \cdot x \mid b \cdot y) \\
&= \nu(\underline{\gamma(a, b)}) \cdot (x \parallel y) + \sigma(a \cdot x \mid b \cdot y) \\
&= \nu(\underline{\underline{\gamma(a, b)}} \cdot (x \parallel y)) + \sigma(a \cdot x \mid b \cdot y)
\end{aligned}$$

Using RSP(USD), we obtain:

$$\begin{aligned}
\text{ACP}_{\text{drt}}^+ \vdash a \cdot x \mid b \cdot y &= [\underline{\underline{\gamma(a, b)}} \cdot (x \parallel y)]^\omega \\
&= [\underline{\underline{\gamma(a, b)}}]^\omega \cdot (x \parallel y) \\
&= \underline{\underline{\gamma(a, b)}} \cdot (x \parallel y) \\
&= (a \mid b) \cdot (x \parallel y)
\end{aligned}$$

(v). Consider the following computation:

$$\begin{aligned}
\text{ACP}_{\text{drt}} \vdash \partial_H(a) &= \partial_H([\underline{a}]^\omega) \\
&= \partial_H(\nu(\underline{a}) + \sigma([\underline{a}]^\omega)) \\
&= \partial_H(\underline{a} + \sigma(a)) \\
&= \partial_H(\underline{a}) + \partial_H(\sigma(a)) \\
&= \underline{a} + \sigma(\partial_H(a)) \\
&= \nu(\underline{a}) + \sigma(\partial_H(a))
\end{aligned}$$

Using RSP(USD), we obtain:

$$\text{ACP}_{\text{drt}}^+ \vdash \partial_H(a) = [\underline{a}]^\omega = a$$

(vi). This case is treated analogously to the previous case.

■

Remark 5.3.3.9 (Properties of ACP_{drt}^+ , Part II)

Note that the equalities of Proposition 5.3.3.8 on page 154 are delayable reformulations of Axioms DRTCF, DRTCM2–DRTCM4 for the communication merge, and Axioms DRTD1 and DRTD2 for the encapsulation. Such reformulations are however not possible for the axioms for the left merge. Take for example DRTM2ID; although we do have:

$$ACP_{drt}^+ \vdash \underline{a} \parallel (x + \underline{\delta}) = \underline{a} \cdot (x + \underline{\delta})$$

the delayable reformulation does not hold:

$$ACP_{drt}^+ \not\vdash a \parallel (x + \underline{\delta}) = a \cdot (x + \underline{\delta})$$

as can be seen by instantiating x with any x such that $x \xrightarrow{\sigma}$. In that case, namely, $a \cdot (x + \underline{\delta})$ can delay, while $a \parallel (x + \underline{\delta})$ cannot, because x cannot.

Proposition 5.3.3.10 (Properties of ACP_{drt} , Part III)

For ACP_{drt} terms x and y , and any $a, b \in A_\delta$, we have the following equalities:

- (i). $ACP_{drt} \vdash \underline{a} \mid b = \underline{a} \mid \underline{b}$
- (ii). $ACP_{drt} \vdash a \mid \underline{b} = \underline{a} \mid \underline{b}$
- (iii). $ACP_{drt} \vdash \underline{a} \mid b \cdot x = (\underline{a} \mid \underline{b}) \cdot x$
- (iv). $ACP_{drt} \vdash a \cdot x \mid \underline{b} = (\underline{a} \mid \underline{b}) \cdot x$
- (v). $ACP_{drt} \vdash a \mid \underline{b} \cdot x = (\underline{a} \mid \underline{b}) \cdot x$
- (vi). $ACP_{drt} \vdash \underline{a} \cdot x \mid b = (\underline{a} \mid \underline{b}) \cdot x$
- (vii). $ACP_{drt} \vdash \underline{a} \cdot x \mid b \cdot y = (\underline{a} \mid \underline{b}) \cdot (x \parallel y)$
- (viii). $ACP_{drt} \vdash a \cdot x \mid \underline{b} \cdot y = (\underline{a} \mid \underline{b}) \cdot (x \parallel y)$

Proof

- (i). Consider the following computation:

$$\begin{aligned} ACP_{drt} \vdash \underline{a} \mid b &= \underline{a} \mid \lfloor \underline{b} \rfloor^\omega \\ &= \underline{a} \mid (\nu(\underline{b}) + \sigma(\lfloor \underline{b} \rfloor^\omega)) \\ &= \underline{a} \mid (\underline{b} + \sigma(b)) \\ &= \underline{a} \mid \underline{b} + \underline{a} \mid \sigma(b) \\ &= \underline{a} \mid \underline{b} + \nu(\underline{a}) \mid \sigma(b) \\ &= \underline{a} \mid \underline{b} + \underline{\delta} \\ &= \underline{a} \mid \underline{b} \end{aligned}$$

- (ii). This case is treated symmetrically to the previous case.

(iii). Consider the following computation:

$$\begin{aligned}
\text{ACP}_{\text{drt}} \vdash \underline{a} \mid b \cdot x &= \underline{a} \mid \underline{[b]}^\omega \cdot x \\
&= \underline{a} \mid (\nu(\underline{b}) + \sigma(\underline{[b]}^\omega)) \cdot x \\
&= \underline{a} \mid (\underline{b} + \sigma(b)) \cdot x \\
&= \underline{a} \mid (\underline{b} \cdot x + \sigma(b) \cdot x) \\
&= \underline{a} \mid \underline{b} \cdot x + \underline{a} \mid \sigma(b) \cdot x \\
&= \underline{a} \mid \underline{b} \cdot x + \nu(\underline{a}) \mid \sigma(b \cdot x) \\
&= \underline{a} \mid \underline{b} \cdot x + \underline{\delta} \\
&= \underline{a} \mid \underline{b} \cdot x \\
&= (\underline{a} \mid \underline{b}) \cdot x
\end{aligned}$$

(iv). This case is treated symmetrically to the previous case.

(v). Consider the following computation:

$$\begin{aligned}
\text{ACP}_{\text{drt}} \vdash a \mid \underline{b} \cdot x &= \underline{[a]}^\omega \mid \underline{b} \cdot x \\
&= (\nu(\underline{a}) + \sigma(\underline{[a]}^\omega)) \mid \underline{b} \cdot x \\
&= (\underline{a} + \sigma(a)) \mid \underline{b} \cdot x \\
&= \underline{a} \mid \underline{b} \cdot x + \sigma(a) \mid \underline{b} \cdot x \\
&= \underline{a} \mid \underline{b} \cdot x + \sigma(a) \mid \nu(\underline{b}) \cdot x \\
&= \underline{a} \mid \underline{b} \cdot x + \sigma(a) \mid \nu(\underline{b} \cdot x) \\
&= \underline{a} \mid \underline{b} \cdot x + \underline{\delta} \\
&= \underline{a} \mid \underline{b} \cdot x \\
&= (\underline{a} \mid \underline{b}) \cdot x
\end{aligned}$$

(vi). This case is treated symmetrically to the previous case.

(vii). Consider the following computation:

$$\begin{aligned}
\text{ACP}_{\text{drt}} \vdash \underline{a} \cdot x \mid b \cdot y &= \underline{a} \cdot x \mid \underline{[b]}^\omega \cdot y \\
&= \underline{a} \cdot x \mid (\nu(\underline{b}) + \sigma(\underline{[b]}^\omega)) \cdot y \\
&= \underline{a} \cdot x \mid (\underline{b} + \sigma(b)) \cdot y \\
&= \underline{a} \cdot x \mid (\underline{b} \cdot y + \sigma(b) \cdot y) \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y + \underline{a} \cdot x \mid \sigma(b) \cdot y \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y + \nu(\underline{a}) \cdot x \mid \sigma(b) \cdot y \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y + \nu(\underline{a} \cdot x) \mid \sigma(b \cdot y) \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y + \underline{\delta} \\
&= \underline{a} \cdot x \mid \underline{b} \cdot y \\
&= (\underline{a} \mid \underline{b}) \cdot (x \parallel y)
\end{aligned}$$

(viii). This case is treated symmetrically to the previous case.

■

Theorem 5.3.3.11 (Elimination for ACP_{drt}^+)

Let t be a closed ACP_{drt} term. Then there is a closed BPA_{drt} term s such that $ACP_{drt}^+ \vdash t = s$.

Proof We use the direct method we described in Proof Outline 4.2.1.2 on page 68. Let t be a closed ACP_{drt} term. The theorem is proven by induction on $n(t)$ and case distinction on the general structure of t .

- (i). $t \equiv \delta$. Then t is a closed BPA_{drt} term.
- (ii). $t \equiv \underline{a}$ for some $a \in A_\delta$. Then t is a closed BPA_{drt} term.
- (iii). $t \equiv a$ for some $a \in A_\delta$. Then t is a closed BPA_{drt} term.
- (iv). $t \equiv t_1 + t_2$ for closed ACP_{drt} terms t_1 and t_2 . By induction there are closed BPA_{drt} terms s_1 and s_2 such that $ACP_{drt}^+ \vdash t_1 = s_1$ and $ACP_{drt}^+ \vdash t_2 = s_2$. But then also $ACP_{drt}^+ \vdash t_1 + t_2 = s_1 + s_2$ and $s_1 + s_2$ is a closed BPA_{drt} term.
- (v). $t \equiv t_1 \cdot t_2$ for closed ACP_{drt} terms t_1 and t_2 . This case is treated analogously to case (iv).
- (vi). $t \equiv \sigma(t_1)$ for a closed ACP_{drt} term t_1 . This case is treated analogously to case (iv).
- (vii). $t \equiv \nu(t_1)$ for a closed ACP_{drt} term t_1 . This case is treated analogously to case (iv).
- (viii). $t \equiv [t_1]^\omega$ for a closed ACP_{drt} term t_1 . This case is treated analogously to case (iv).
- (ix). $t \equiv t_1 \parallel t_2$ for closed ACP_{drt} terms t_1 and t_2 . By induction there are closed BPA_{drt} terms s_1 and s_2 such that $ACP_{drt}^+ \vdash t_1 = s_1$ and $ACP_{drt}^+ \vdash t_2 = s_2$. By Theorem 4.3.4.1, the elimination theorem for BPA_{drt} , there are basic terms r_1 and r_2 such that $BPA_{drt}^+ \vdash s_1 = r_1$ and $BPA_{drt}^+ \vdash s_2 = r_2$. But then also, $ACP_{drt}^+ \vdash t_1 = r_1$, $ACP_{drt}^+ \vdash t_2 = r_2$, and $ACP_{drt}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2$. We proceed by induction on the structure of basic terms, and distinguish all possible cases for basic term r_1 :
 - (a) $r_1 \equiv \delta$. Then $ACP_{drt}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \delta \parallel r_2 = \delta$, and δ is a closed BPA_{drt} term.
 - (b) $r_1 \equiv \underline{a}$ for some $a \in A_\delta$. Using Lemma 3.2.4.19 we distinguish two cases:
 - 1. $r_2 = \delta$. Then we have: $ACP_{drt}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r_1 \parallel \delta = \delta$, and δ is a closed BPA_{drt} term.
 - 2. $r_2 = r_2 + \underline{\delta}$. Then we have: $ACP_{drt}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{a} \parallel (r_2 + \underline{\delta}) = \underline{a} \cdot (r_2 + \underline{\delta}) = \underline{a} \cdot r_2$, and $\underline{a} \cdot r_2$ is a closed BPA_{drt} term.
 - (c) $r_1 \equiv a$ for some $a \in A_\delta$. Using Lemma 3.2.4.18 we distinguish four cases:
 - 1. $r_2 = \delta$. Then we have: $ACP_{drt}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r_1 \parallel \delta = \delta$, and δ is a closed BPA_{drt} term.
 - 2. $r_2 = \nu(r_2) + \underline{\delta}$. Then we have:

$$\begin{aligned}
 ACP_{drt}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
 &= a \parallel r_2 \\
 &= [\underline{a}]^\omega \parallel r_2 \\
 &= (\nu(\underline{a}) + \sigma([\underline{a}]^\omega)) \parallel r_2
 \end{aligned}$$

$$\begin{aligned}
&= (\underline{a} + \sigma(a)) \parallel r_2 \\
&= \underline{a} \parallel r_2 + \sigma(a) \parallel r_2 \\
&= \underline{a} \parallel (\nu(r_2) + \underline{\delta}) + \sigma(a) \parallel (\nu(r_2) + \underline{\delta}) \\
&= \underline{a} \cdot (\nu(r_2) + \underline{\delta}) + \underline{\delta} \\
&= \underline{a} \cdot r_2 + \underline{\delta} \\
&= \underline{a} \cdot r_2 + \underline{\delta} \cdot r_2 \\
&= (\underline{a} + \underline{\delta}) \cdot r_2 \\
&= \underline{a} \cdot r_2,
\end{aligned}$$

and $\underline{a} \cdot r_2$ is a closed BPA_{drt} term.

3. $r_2 = \lfloor r_2 \rfloor^\omega$. Then, using Proposition 5.3.3.7(vi), we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = a \parallel \lfloor r_2 \rfloor^\omega = a \cdot \lfloor r_2 \rfloor^\omega = a \cdot r_2$, and $a \cdot r_2$ is a closed BPA_{drt} term.
4. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 such that $n(r'_2) < n(r_2)$. Then we have:

$$\begin{aligned}
\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
&= a \parallel r_2 \\
&= \lfloor \underline{a} \rfloor^\omega \parallel r_2 \\
&= (\nu(\underline{a}) + \sigma(\lfloor \underline{a} \rfloor^\omega)) \parallel r_2 \\
&= (\underline{a} + \sigma(a)) \parallel r_2 \\
&= \underline{a} \parallel r_2 + \sigma(a) \parallel r_2 \\
&= \underline{a} \parallel (\nu(r_2) + \sigma(r'_2)) + \\
&\quad \sigma(a) \parallel (\nu(r_2) + \sigma(r'_2)) \\
&= \underline{a} \parallel (\nu(r_2) + \sigma(r'_2) + \underline{\delta}) + \sigma(a \parallel r'_2) \\
&= \underline{a} \cdot (\nu(r_2) + \sigma(r'_2) + \underline{\delta}) + \sigma(a \parallel r'_2) \\
&= \underline{a} \cdot (\nu(r_2) + \sigma(r'_2)) + \sigma(a \parallel r'_2) \\
&= \underline{a} \cdot r_2 + \sigma(a \parallel r'_2).
\end{aligned}$$

By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash a \parallel r'_2 = p$. Then, $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{a} \cdot r_2 + \sigma(a \parallel r'_2) = \underline{a} \cdot r_2 + \sigma(p)$, and $\underline{a} \cdot r_2 + \sigma(p)$ is a closed BPA_{drt} term.

- (d) $r_1 \equiv \underline{a} \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Using Lemma 3.2.4.19 we distinguish two cases:
 1. $r_2 = \dot{\delta}$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r_1 \parallel \dot{\delta} = \dot{\delta}$, and $\dot{\delta}$ is a closed BPA_{drt} term.
 2. $r_2 = r_2 + \underline{\delta}$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \underline{a} \cdot r'_1 \parallel (r_2 + \underline{\delta}) = \underline{a} \cdot (r'_1 \parallel (r_2 + \underline{\delta})) = \underline{a} \cdot (r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) = \underline{a} \cdot p$, and $\underline{a} \cdot p$ is a closed BPA_{drt} term.
- (e) $r_1 \equiv a \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Using Lemma 3.2.4.18 we distinguish four cases:
 1. $r_2 = \dot{\delta}$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = r_1 \parallel \dot{\delta} = \dot{\delta}$, and $\dot{\delta}$ is a closed BPA_{drt} term.

2. $r_2 = \nu(r_2) + \underline{\delta}$. Then we have:

$$\begin{aligned}
\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
&= a \cdot r'_1 \parallel r_2 \\
&= [\underline{a}]^\omega \cdot r'_1 \parallel r_2 \\
&= (\nu(\underline{a}) + \sigma([\underline{a}]^\omega)) \cdot r'_1 \parallel r_2 \\
&= (\underline{a} + \sigma(a)) \cdot r'_1 \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a) \cdot r'_1) \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a \cdot r'_1)) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel r_2 + \sigma(a \cdot r'_1) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel (\nu(r_2) + \underline{\delta}) + \sigma(a \cdot r'_1) \parallel (\nu(r_2) + \underline{\delta}) \\
&= \underline{a} \cdot (r'_1 \parallel (\nu(r_2) + \underline{\delta})) + \underline{\delta} \\
&= \underline{a} \cdot (r'_1 \parallel r_2) + \underline{\delta} \\
&= \underline{a} \cdot (r'_1 \parallel r_2) + \underline{\delta} \cdot (r'_1 \parallel r_2) \\
&= (\underline{a} + \underline{\delta}) \cdot (r'_1 \parallel r_2) \\
&= \underline{a} \cdot (r'_1 \parallel r_2).
\end{aligned}$$

By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) = \underline{a} \cdot p$, and $\underline{a} \cdot p$ is a closed BPA_{drt} term.

3. $r_2 = [r_2]^\omega$. Then, using Proposition 5.3.3.7(vii), we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = a \cdot r'_1 \parallel [r_2]^\omega = a \cdot (r'_1 \parallel [r_2]^\omega) = a \cdot (r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = a \cdot (r'_1 \parallel r_2) = a \cdot p$, and $a \cdot p$ is a closed BPA_{drt} term.
4. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 such that $n(r'_2) < n(r_2)$. Then we have:

$$\begin{aligned}
\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 &= r_1 \parallel r_2 \\
&= a \cdot r'_1 \parallel r_2 \\
&= [\underline{a}]^\omega \cdot r'_1 \parallel r_2 \\
&= (\nu(\underline{a}) + \sigma([\underline{a}]^\omega)) \cdot r'_1 \parallel r_2 \\
&= (\underline{a} + \sigma(a)) \cdot r'_1 \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a) \cdot r'_1) \parallel r_2 \\
&= (\underline{a} \cdot r'_1 + \sigma(a \cdot r'_1)) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel r_2 + \sigma(a \cdot r'_1) \parallel r_2 \\
&= \underline{a} \cdot r'_1 \parallel (\nu(r_2) + \sigma(r'_2)) + \\
&\quad \sigma(a \cdot r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) \\
&= \underline{a} \cdot r'_1 \parallel (\nu(r_2) + \sigma(r'_2) + \underline{\delta}) + \\
&\quad \sigma(a \cdot r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) \\
&= \underline{a} \cdot (r'_1 \parallel (\nu(r_2) + \sigma(r'_2) + \underline{\delta})) + \sigma(a \cdot r'_1 \parallel r'_2)
\end{aligned}$$

$$\begin{aligned}
&= \underline{a} \cdot (r'_1 \parallel (\nu(r_2) + \sigma(r'_2))) + \sigma(r_1 \parallel r'_2) \\
&= \underline{a} \cdot (r'_1 \parallel r_2) + \sigma(r_1 \parallel r'_2).
\end{aligned}$$

By the induction hypothesis there exist closed BPA_{drt} terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p_1$ and $\text{ACP}_{\text{drt}}^+ \vdash r_1 \parallel r'_2 = p_2$. Then, $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \underline{a} \cdot (r'_1 \parallel r_2) + \sigma(r_1 \parallel r'_2) = \underline{a} \cdot p_1 + \sigma(p_2)$, and $\underline{a} \cdot p_1 + \sigma(p_2)$ is a closed BPA_{drt} term.

- (f) $r_1 \equiv r'_1 + r'_1$ for basic terms r'_1 and r'_1 . Then $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = (r'_1 + r'_1) \parallel r_2 = r'_1 \parallel r_2 + r'_1 \parallel r_2$. By induction there exist closed BPA_{drt} terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p_1$ and $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p_2$. Then also $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r'_1 \parallel r_2 + r'_1 \parallel r_2 = p_1 + p_2$, and $p_1 + p_2$ is a closed BPA_{drt} term.
- (g) $r_1 \equiv \sigma(r'_1)$ for a basic term r'_1 . Using Lemma 3.2.4.18 we distinguish four cases:
1. $r_2 = \delta$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel \delta = \delta$, and δ is a closed BPA_{drt} term.
 2. $r_2 = \nu(r_2) + \underline{\delta}$. Then $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel (\nu(r_2) + \underline{\delta}) = \underline{\delta}$, and $\underline{\delta}$ is a closed BPA_{drt} term.
 3. $r_2 = \lfloor r_2 \rfloor^\omega$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel \lfloor r_2 \rfloor^\omega = \sigma(r'_1) \parallel (\nu(r_2) + \sigma(\lfloor r_2 \rfloor^\omega)) = \sigma(r'_1 \parallel \lfloor r_2 \rfloor^\omega) = \sigma(r'_1 \parallel r_2)$. By the induction hypothesis there exists a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r_2 = p$. Then, $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \sigma(r'_1 \parallel r_2) = \sigma(p)$, and $\sigma(p)$ is a closed BPA_{drt} term.
 4. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 such that $n(r'_2) < n(r_2)$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = r_1 \parallel r_2 = \sigma(r'_1) \parallel (\nu(r_2) + \sigma(r'_2)) = \sigma(r'_1 \parallel r'_2)$. By the induction hypothesis there is a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r'_2 = p$. Then, $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = \sigma(r'_1 \parallel r'_2) = \sigma(p)$, and $\sigma(p)$ is a closed BPA_{drt} term.
- (x). $t \equiv t_1 \mid t_2$ for closed ACP_{drt} terms t_1 and t_2 . By induction there are closed BPA_{drt} terms s_1 and s_2 such that $\text{ACP}_{\text{drt}}^+ \vdash t_1 = s_1$ and $\text{ACP}_{\text{drt}}^+ \vdash t_2 = s_2$. By Theorem 4.3.4.1, the elimination theorem for $\text{BPA}_{\text{drt}}^+$, there are basic terms r_1 and r_2 such that $\text{BPA}_{\text{drt}}^+ \vdash s_1 = r_1$ and $\text{BPA}_{\text{drt}}^+ \vdash s_2 = r_2$. But then also, $\text{ACP}_{\text{drt}}^+ \vdash t_1 = r_1$, $\text{ACP}_{\text{drt}}^+ \vdash t_2 = r_2$, and $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2$. We prove this case by simultaneous induction on the structure of basic terms r_1 and r_2 . We examine all possible cases (of which there are in total 49, some of which can be treated simultaneously, reducing our task to “just” 22 cases):
- (a) $r_1 \equiv \delta$ and r_2 is of arbitrary form. Then $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = \delta \mid r_2 = \delta$, and δ is a closed BPA_{drt} term.
 - (b) r_1 is of arbitrary form and $r_2 \equiv \delta$. This case is treated symmetrically to the previous case.
 - (c) $r_1 \equiv \underline{a}$ and $r_2 \equiv \underline{b}$ for some $a, b \in A_\delta$. Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \mid \underline{b} = \underline{c}$, and \underline{c} is a closed BPA_{drt} term.
 - (d) $r_1 \equiv \underline{a}$ and $r_2 \equiv b$ for some $a, b \in A_\delta$. Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \mid b = \underline{c}$, and \underline{c} is a closed BPA_{drt} term.

- (e) $r_1 \equiv a$ and $r_2 \equiv \underline{b}$ for some $a, b \in A_\delta$. This case is treated symmetrically to the previous case.
- (f) $r_1 \equiv a$ and $r_2 \equiv b$ for some $a, b \in A_\delta$. Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = a \mid b = c$, and c is a closed BPA_{drt} term.
- (g) $r_1 \equiv \underline{a}$ and $r_2 \equiv \underline{b} \cdot r'_2$ for some $a, b \in A_\delta$ and some basic term r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \mid \underline{b} \cdot r'_2 = \underline{c} \cdot r'_2$, and $\underline{c} \cdot r'_2$ is a closed BPA_{drt} term.
- (h) $r_1 \equiv \underline{a} \cdot r'_1$ and $r_2 \equiv \underline{b}$ for some $a, b \in A_\delta$ and some basic term r'_1 . This case is treated symmetrically to the previous case.
- (i) $r_1 \equiv \underline{a}$ and $r_2 \equiv b \cdot r'_2$ for some $a, b \in A_\delta$ and some basic term r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \mid b \cdot r'_2 = \underline{c} \cdot r'_2$, and $\underline{c} \cdot r'_2$ is a closed BPA_{drt} term.
- (j) $r_1 \equiv a \cdot r'_1$ and $r_2 \equiv \underline{b}$ for some $a, b \in A_\delta$ and some basic term r'_1 . This case is treated symmetrically to the previous case.
- (k) $r_1 \equiv a$ and $r_2 \equiv \underline{b} \cdot r'_2$ for some $a, b \in A_\delta$ and some basic term r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = a \mid \underline{b} \cdot r'_2 = \underline{c} \cdot r'_2$, and $\underline{c} \cdot r'_2$ is a closed BPA_{drt} term.
- (l) $r_1 \equiv \underline{a} \cdot r'_1$ and $r_2 \equiv b$ for some $a, b \in A_\delta$ and some basic term r'_1 . This case is treated symmetrically to the previous case.
- (m) $r_1 \equiv a$ and $r_2 \equiv b \cdot r'_2$ for some $a, b \in A_\delta$ and some basic term r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = a \mid b \cdot r'_2 = c \cdot r'_2$, and $c \cdot r'_2$ is a closed BPA_{drt} term.
- (n) $r_1 \equiv a \cdot r'_1$ and $r_2 \equiv b$ for some $a, b \in A_\delta$ and some basic term r'_1 . This case is treated symmetrically to the previous case.
- (o) $r_1 \equiv \underline{a} \cdot r'_1$ and $r_2 \equiv \underline{b} \cdot r'_2$ for some $a, b \in A_\delta$ and some basic terms r'_1 and r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \cdot r'_1 \mid \underline{b} \cdot r'_2 = \underline{c} \cdot (r'_1 \parallel r'_2)$. By the induction hypothesis there exists a closed BPA_{drt} term s' such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r'_2 = s'$. So $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = \underline{c} \cdot (r'_1 \parallel r'_2) = \underline{c} \cdot s'$, and $\underline{c} \cdot s'$ is a closed BPA_{drt} term.
- (p) $r_1 \equiv \underline{a} \cdot r'_1$ and $r_2 \equiv b \cdot r'_2$ for some $a, b \in A_\delta$ and some basic terms r'_1 and r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = \underline{a} \cdot r'_1 \mid b \cdot r'_2 = \underline{c} \cdot (r'_1 \parallel r'_2)$. By the induction hypothesis there exists a closed BPA_{drt} term s' such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r'_2 = s'$. So $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = \underline{c} \cdot (r'_1 \parallel r'_2) = \underline{c} \cdot s'$, and $\underline{c} \cdot s'$ is a closed BPA_{drt} term.
- (q) $r_1 \equiv a \cdot r'_1$ and $r_2 \equiv \underline{b} \cdot r'_2$ for some $a, b \in A_\delta$ and some basic terms r'_1 and r'_2 . This case is treated symmetrically to the previous case.
- (r) $r_1 \equiv a \cdot r'_1$ and $r_2 \equiv b \cdot r'_2$ for some $a, b \in A_\delta$ and some basic terms r'_1 and r'_2 . Suppose that $\gamma(a, b) = c$. Then we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = a \cdot r'_1 \mid b \cdot r'_2 = c \cdot (r'_1 \parallel r'_2)$. By the induction hypothesis there exists a closed BPA_{drt} term s' such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 \parallel r'_2 = s'$. So $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = c \cdot (r'_1 \parallel r'_2) = \underline{c} \cdot s'$, and $\underline{c} \cdot s'$ is a closed BPA_{drt} term.
- (s) $r_1 \equiv r'_1 + r''_1$ for some basic terms r'_1 and r''_1 , and r_2 is of arbitrary form. Then $\text{ACP}_{\text{drt}}^+ \vdash t_1 \mid t_2 = r_1 \mid r_2 = (r'_1 + r''_1) \mid r_2 = r'_1 \mid r_2 + r''_1 \mid r_2$. By the induction

hypothesis there exist closed BPA_{drt} terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 | r_2 = p_1$ and $\text{ACP}_{\text{drt}}^+ \vdash r'_1 | r_2 = p_2$. So, we have $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = r'_1 | r_2 + r'_1 | r_2 = p_1 + p_2$, and $p_1 + p_2$ is a closed BPA_{drt} term.

- (t) r_1 is of arbitrary form and $r_2 \equiv r'_2 + r''_2$ for some basic terms r'_2 and r''_2 . This case is treated symmetrically to the previous case.
- (u) $r_1 \equiv \sigma(r'_1)$ and r'_1 a basic term, and r_2 is of arbitrary form. Using Lemma 3.2.4.18 we distinguish four cases:
1. $r_2 = \delta$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = r_1 | r_2 = \sigma(r'_1) | \delta = \delta$, and δ is a closed BPA_{drt} term.
 2. $r_2 = \nu(r_2) + \underline{\delta}$. Then $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = r_1 | r_2 = \sigma(r'_1) | (\nu(r_2) + \underline{\delta}) = \underline{\delta}$, and $\underline{\delta}$ is a closed BPA_{drt} term.
 3. $r_2 = \lfloor r_2 \rfloor^\omega$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = r_1 | r_2 = \sigma(r'_1) | \lfloor r_2 \rfloor^\omega = \sigma(r'_1) | (\nu(r_2) + \sigma(\lfloor r_2 \rfloor^\omega)) = \sigma(r'_1 | \lfloor r_2 \rfloor^\omega) = \sigma(r'_1 | r_2)$. By the induction hypothesis there is a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 | r_2 = p$. But then also $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = \sigma(r'_1 | r_2) = \sigma(p)$, and $\sigma(p)$ is a closed BPA_{drt} term.
 4. $r_2 = \nu(r_2) + \sigma(r'_2)$ for a basic term r'_2 such that $n(r'_2) < n(r_2)$. Then we have: $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = r_1 | r_2 = \sigma(r'_1) | (\nu(r_2) + \sigma(r'_2)) = \sigma(r'_1) | \sigma(r'_2) = \sigma(r'_1 | r'_2)$. By the induction hypothesis there is a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash r'_1 | r'_2 = p$. But then also $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = \sigma(r'_1 | r'_2) = \sigma(p)$, and $\sigma(p)$ is a closed BPA_{drt} term.
- (v) r_1 is of arbitrary form and $r_2 \equiv \sigma(r'_2)$ and r'_2 a basic term. This case is treated symmetrically to the previous case.
- (xi). $t \equiv t_1 \parallel t_2$ for closed ACP_{drt} terms t_1 and t_2 . Then $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1 + t_1 | t_2$. By (ix) and (x) there are closed BPA_{drt} terms p_1, p_2 , and p_3 , such that $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = p_1$, $\text{ACP}_{\text{drt}}^+ \vdash t_2 \parallel t_1 = p_2$, and $\text{ACP}_{\text{drt}}^+ \vdash t_1 | t_2 = p_3$. But then also $\text{ACP}_{\text{drt}}^+ \vdash t_1 \parallel t_2 = t_1 \parallel t_2 + t_2 \parallel t_1 + t_1 | t_2 = p_1 + p_2 + p_3$, and $p_1 + p_2 + p_3$ is a closed BPA_{drt} term.
- (xii). $t \equiv \partial_H(t_1)$ for a closed ACP_{drt} term t_1 . By induction there is a closed BPA_{drt} term s_1 such that $\text{ACP}_{\text{drt}}^+ \vdash t_1 = s_1$. By Theorem 4.3.4.1, the elimination theorem for $\text{BPA}_{\text{drt}}^+$, there is a basic term r_1 such that $\text{BPA}_{\text{drt}}^+ \vdash s_1 = r_1$. But then also, $\text{ACP}_{\text{drt}}^+ \vdash t_1 = r_1$, and $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r_1)$. We prove this case by induction on the structure of basic term r_1 :
- (a) $r_1 \equiv \delta$. Then $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\delta) = \delta$, and δ is a closed BPA_{drt} term.
 - (b) $r_1 \equiv \underline{a}$ for some $a \in A_\delta$. Suppose $a \notin H$. Then, $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\underline{a}) = \underline{a}$, and \underline{a} is a closed BPA_{drt} term. Otherwise, $a \in H$, and we get $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\underline{a}) = \underline{\delta}$, and $\underline{\delta}$ is a closed BPA_{drt} term.
 - (c) $r_1 \equiv a$ for some $a \in A_\delta$. This case is treated analogously to case (b).
 - (d) $r_1 \equiv \underline{a} \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . Then $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\underline{a} \cdot r'_1) = \partial_H(\underline{a}) \cdot \partial_H(r'_1)$. By the induction hypothesis there exist closed BPA_{drt} terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(\underline{a}) = p_1$ and $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(r'_1) =$

- p_2 . Then also $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(\underline{a}) \cdot \partial_H(r'_1) = p_1 \cdot p_2$, and $p_1 \cdot p_2$ is a closed BPA_{drt} term.
- (e) $r_1 \equiv a \cdot r'_1$ for some $a \in A_\delta$ and basic term r'_1 . This case is treated analogously to case (d).
- (f) $r_1 \equiv r'_1 + r''_1$ for closed BPA_{drt} terms r'_1 and r''_1 . Then $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(r'_1 + r''_1) = \partial_H(r'_1) + \partial_H(r''_1)$. By the induction hypothesis there exist closed BPA_{drt} terms p_1 and p_2 such that $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(r'_1) = p_1$ and $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(r''_1) = p_2$. Then also $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r'_1) + \partial_H(r''_1) = p_1 + p_2$, and $p_1 + p_2$ is a closed BPA_{drt} term.
- (g) $r_1 \equiv \sigma(r'_1)$ for some closed BPA_{drt} term r'_1 . Then $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \partial_H(r_1) = \partial_H(\sigma(r'_1)) = \sigma(\partial_H(r'_1))$. By the induction hypothesis there exist a closed BPA_{drt} term p such that $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(r'_1) = p$. Then also $\text{ACP}_{\text{drt}}^+ \vdash \partial_H(t_1) = \sigma(\partial_H(r'_1)) = \sigma(p)$, and $\sigma(p)$ is a closed BPA_{drt} term.

■

Corollary 5.3.3.12 (Elimination for $\text{ACP}_{\text{drt}}^+$)

Let t be a closed ACP_{drt} term. Then there is a basic term s such that $\text{ACP}_{\text{drt}}^+ \vdash s = t$.

Proof This follows immediately from:

- (i). The elimination theorem for $\text{ACP}_{\text{drt}}^+$ (see Theorem 5.3.3.11),
- (ii). the elimination theorem for $\text{BPA}_{\text{drt}}^+$ (see Theorem 4.3.4.1),
- (iii). the fact that all axioms of $\text{BPA}_{\text{drt}}^+$ are also contained in $\text{ACP}_{\text{drt}}^+$.

■

Remark 5.3.3.13 (Elimination for ACP_{drt})

Elimination for a somewhat different version of ACP_{drt} is also claimed (without proof) in Section 3.10 of BAETEN AND BERGSTRA [24].

Theorem 5.3.3.14 (Soundness of $\text{ACP}_{\text{drt}}^+$)

The set of closed ACP_{drt} terms modulo bisimulation equivalence is a model of $\text{ACP}_{\text{drt}}^+$.

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69. We only prove soundness for the axioms of ACP_{drt} that have not been treated in earlier soundness proofs. Note that to extend these proofs to ACP_{drt} , we have to check that the bisimulations given in previous soundness proofs respect the ID predicate (as required by transfer condition (iv.) in Definition 3.2.3.5 on page 53). However, as the fact that they do can be easily checked, we will not give details.

Axiom DRTCM6ID Take the relation:

$$R = \{(\sigma(s) \mid (\nu(t) + \underline{\delta}), \underline{\delta}) \mid s, t \in C(\text{ACP}_{\text{drt}})\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $\sigma(s) \mid (\nu(t) + \underline{\delta}) \nrightarrow$. Also for the right-hand side there are no transitions possible: $\underline{\delta} \nrightarrow$. Finally, neither side satisfies the ID predicate: $\neg \text{ID}(\sigma(s) \mid (\nu(t) + \underline{\delta}))$ and $\neg \text{ID}(\underline{\delta})$.

Axiom DRTCM7ID Take the relation:

$$R = \{((\nu(s) + \underline{\delta}) \mid \sigma(t), \underline{\delta}) \mid s, t \in C(\text{ACP}_{\text{drt}})\}$$

This case is treated symmetrically to the previous case.

Axiom DRTMID3 Take the relation:

$$R = \{(s \mid \dot{\delta}, \dot{\delta}) \mid s \in C(\text{ACP}_{\text{drt}})\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $s \mid \dot{\delta} \not\rightarrow$. Also for the right-hand side there are no transitions possible: $\dot{\delta} \not\rightarrow$. Finally, both sides satisfy the ID predicate: $\text{ID}(s \mid \dot{\delta})$ and $\text{ID}(\dot{\delta})$.

Axiom DRTMID4 Take the relation:

$$R = \{(\dot{\delta} \mid s, \dot{\delta}) \mid s \in C(\text{ACP}_{\text{drt}})\}$$

This case is treated symmetrically to the previous case.

Axiom DRTD6 Take the relation:

$$R = \{(\partial_H(\dot{\delta}), \dot{\delta})\}$$

We look at the transitions of both sides at the same time. Observe that there are no transitions possible on the left-hand side: $\partial_H(\dot{\delta}) \not\rightarrow$. Also for the right-hand side there are no transitions possible: $\dot{\delta} \not\rightarrow$. Finally, both sides satisfy the ID predicate: $\text{ID}(\partial_H(\dot{\delta}))$ and $\text{ID}(\dot{\delta})$.

■

Remark 5.3.3.15 (Soundness of ACP_{drt})

Soundness of a somewhat different version of ACP_{drt} is also claimed (without proof) in Section 3.10 of BAETEN AND BERGSTRA [24].

Theorem 5.3.3.16 (Conservativity of $\text{ACP}_{\text{drt}}^+$ with respect to $\text{BPA}_{\text{drt}}^+$)

The process algebra $\text{ACP}_{\text{drt}}^+$ is a conservative extension of the process algebra $\text{BPA}_{\text{drt}}^+$.

Proof In order to prove conservativity it is sufficient to verify that the following conditions are satisfied:

- (i). Bisimulation equivalence is definable in terms of predicate and relation symbols only,
- (ii). $\text{BPA}_{\text{drt}}^+$ is a complete axiomatization with respect to the bisimulation equivalence model induced by $T(\text{BPA}_{\text{drt}})$ (see Theorem 4.3.4.9),
- (iii). $\text{ACP}_{\text{drt}}^+$ is a sound axiomatization with respect to the bisimulation equivalence model induced by $T(\text{ACP}_{\text{drt}})$ (see Theorem 5.3.3.14),
- (iv). $T(\text{ACP}_{\text{drt}})$ is an operationally conservative extension of $T(\text{BPA}_{\text{drt}})$.

And in order for $T(\text{ACP}_{\text{drt}})$ indeed to be an operationally conservative extension of $T(\text{BPA}_{\text{drt}})$ we must verify the following conditions:

- (i). $T(\text{BPA}_{\text{drt}})$ is a pure, well-founded term-deduction system in path format,
- (ii). $T(\text{ACP}_{\text{drt}})$ is a term-deduction system in path format,
- (iii). $T(\text{BPA}_{\text{drt}}) \oplus T(\text{ACP}_{\text{drt}})$ is defined.

That the above properties hold can be trivially checked from the relevant definitions. ■

Theorem 5.3.3.17 (Completeness of $\text{ACP}_{\text{drt}}^+$)

The axiom system $\text{ACP}_{\text{drt}}^+$ is a complete axiomatization of the set of closed ACP_{drt} terms modulo bisimulation equivalence.

Proof We use Verhoef's method described in Proof Outline 4.2.3.4 on page 71. Completeness then follows immediately from:

- (i). $\text{ACP}_{\text{drt}}^+$ has the elimination property for BPA_{drt} (see Theorem 5.3.3.11),
- (ii). $\text{ACP}_{\text{drt}}^+$ is a conservative extension of $\text{BPA}_{\text{drt}}^+$ (see Theorem 5.3.3.16).

■

Remark 5.3.3.18 (Completeness of ACP_{drt})

Completeness of a somewhat different version of ACP_{drt} is also claimed (without proof) in Section 3.10 of BAETEN AND BERGSTRA [24].

5.3.4 ACP'_{drt}

In this section, we define a process algebra named ACP'_{drt} , that is almost identical to ACP_{drt} , except that it has thirteen additional axioms. These are chosen such that elimination, soundness, and completeness results for ACP'_{drt} follow as corollaries from the corresponding results for $\text{ACP}_{\text{drt}}^+$.

Definition 5.3.4.1 (Axioms of ACP'_{drt})

The process algebra ACP'_{drt} is axiomatized by the axioms of ACP_{drt} that are given in Definition 5.3.3.2 on page 152, Axioms USD1–USD5 shown in Table 4.5 on page 102, Axioms USD6–USD7 shown in Table 5.16 on page 151, and Axioms USDCF, USDCM2–USDCM4, and USDD1–USDD2 shown in Table 5.19 on the next page: $\text{ACP}'_{\text{drt}} = \text{A1–A5} + \text{A6ID} + \text{A7ID} + \text{DRT1–DRT5} + \text{DRTSID} + \text{DCS1–DCS4} + \text{DCSID} + \text{ATS} + \text{USD} + \text{DRTM2ID–DRTM3ID} + \text{DRTM4} + \text{DRTM5ID} + \text{DRTM6} + \text{DRTCM1–DRTCM5} + \text{DRTCM6ID–DRTCM7ID} + \text{DRTCM12–DRTCM13} + \text{DRTCF} + \text{DRTD1–DRTD6} + \text{DRTMID1–DRTMID4} + \text{USD1–USD7} + \text{USDCF} + \text{USDCM2–USDCM4} + \text{USDD1–USDD2}$.

☞ Axioms USD1–USD7, USDCF, USDCM2–USDCM4, and USDD1–USDD2 precisely correspond to equalities (i)–(vii) of Proposition 5.3.3.7 on page 154 and equalities (i)–(vi) of Proposition 5.3.3.8 on page 154. In this way, we obtain an axiomatization that is in many ways (elimination, soundness, completeness) like $\text{ACP}_{\text{drt}}^+$, but is also purely equational, i.e., does not contain conditional axioms or recursion principles.

$a \mid b = c$	if $\gamma(a, b) = c$	USDCF
$a \mid b \cdot x = (a \mid b) \cdot x$		USDCM2
$a \cdot x \mid b = (a \mid b) \cdot x$		USDCM3
$a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel y)$		USDCM4
$\partial_H(a) = a$	if $a \notin H$	USDD1
$\partial_H(a) = \delta$	if $a \in H$	USDD2

Table 5.19: Axioms for communication merge and delayable actions.

Definition 5.3.4.2 (Signature, Semantics, and Basic Terms of ACP'_{drt})

The signature, semantics, bisimulation, bisimulation model, and basic terms of ACP'_{drt} are the same as those of ACP_{drt} .

Corollary 5.3.4.3 (Elimination for ACP'_{drt})

Let t be a closed ACP'_{drt} term. Then there is a basic term s such that $ACP'_{drt} \vdash t = s$.

Proof In the same way as Theorem 5.3.3.11 and Corollary 5.3.3.12. Note that all equalities of Propositions 5.3.3.7 and 5.3.3.8 that are used in the proof of Theorem 5.3.3.11 correspond to derivable equalities in ACP'_{drt} . ■

Corollary 5.3.4.4 (Soundness of ACP'_{drt})

The set of closed ACP'_{drt} terms modulo bisimulation equivalence is a model of ACP'_{drt} .

Proof We use the indirect method of Proof Outline 4.2.2.2 on page 70. The result follows directly from the soundness of ACP'_{drt} (see Theorem 5.3.3.14 on page 165) combined with the fact that Axioms USD1–USD7, USDCF, USDCM2–USDCM4, and USDD1–USDD2 are derivable in ACP'_{drt} (see Proposition 5.3.3.7 on page 154 and Proposition 5.3.3.8 on page 154). ■

Corollary 5.3.4.5 (Completeness of ACP'_{drt})

The axiom system ACP'_{drt} is a complete axiomatization of the set of closed ACP'_{drt} terms modulo bisimulation equivalence.

Proof We use the indirect method of Proof Outline 4.2.3.2 on page 71. Careful inspection of the dependencies between the proofs in this section reveals that the proof of Theorem 5.3.3.17 only relies upon RSP(USD) to ensure Proposition 5.3.3.7(i)–(vii) and Proposition 5.3.3.8(i)–(vi). So, we obviously do not need RSP(USD) anymore if we add the corresponding Axioms USD1–USD7, USDCF, USDCM2–USDCM4, and USDD1–USDD2, and the result follows. ■

5.3.5 ACP''_{drt}

In this section, we define a process algebra named ACP''_{drt} , that is almost identical to ACP'_{drt} , except that we have removed Axioms USDCF, USDCM2–USDCM4, and USDD1–USDD2 and introduced two new axioms that make the removed six axioms derivable.

Definition 5.3.5.1 (Axioms of ACP''_{drt})

The process algebra ACP''_{drt} is axiomatized by the axioms of ACP'_{drt} that are given in Definition 5.3.4.1 on page 167, *minus* Axioms USDCF and USDCM2–USDCM4, *plus* Axioms USD8 and USD9 shown in Table 5.20: $ACP''_{\text{drt}} = A1\text{--}A5 + A6\text{ID} + A7\text{ID} + \text{DRT1--DRT5} + \text{DRTSID} + \text{DCS1--DCS4} + \text{DCSID} + \text{ATS} + \text{USD} + \text{DRTM2ID--DRTM3ID} + \text{DRTM4} + \text{DRTM5ID} + \text{DRTM6} + \text{DRTCM1--DRTCM5} + \text{DRTCM6ID--DRTCM7ID} + \text{DRTCM12--DRTCM13} + \text{DRTCF} + \text{DRTD1--DRTD6} + \text{DRTMID1--DRTMID4} + \text{USD1--USD9}$.

$$\begin{aligned} [x \mid y]^\omega &= [x]^\omega \mid [y]^\omega && \text{USD8} \\ [\partial_H(x)]^\omega &= \partial_H([x]^\omega) && \text{USD9} \end{aligned}$$

Table 5.20: Additional axioms for unbounded start delay.

☞ Axiom USD8 expresses that the unbounded start delay distributes over the communication merge. As we will show, this enables us to derive the equalities of Axioms USDCF and USDCM2–USDCM4.

Axiom USD9 expresses that the unbounded start delay commutes with the encapsulation. This enables us to derive the equalities of Axioms USDD1 and USDD2.

Remark 5.3.5.2 (ACP'_{drt} versus ACP''_{drt})

ACP''_{drt} is very much like ACP'_{drt} , except that it has fewer and easier axioms. Keep in mind however, that the additional axioms of ACP'_{drt} were derived in a systematic manner (which can be generalized), while Axioms USD8 and USD9 are based on ad hoc insight (which cannot).

Definition 5.3.5.3 (Signature, Semantics, and Basic Terms of ACP''_{drt})

The signature, semantics, bisimulation, bisimulation model, and basic terms of ACP''_{drt} are the same as those of ACP_{drt} .

Proposition 5.3.5.4 (Properties of ACP''_{drt})

For ACP''_{drt} terms x and y and any $a, b, c \in A_\delta$, we have the following equalities:

- (i). $ACP''_{\text{drt}} \vdash a \mid b = c \quad \text{if } \gamma(a, b) = c$
- (ii). $ACP''_{\text{drt}} \vdash a \mid b \cdot x = (a \mid b) \cdot x$
- (iii). $ACP''_{\text{drt}} \vdash a \cdot x \mid b = (a \mid b) \cdot x$
- (iv). $ACP''_{\text{drt}} \vdash a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel y)$
- (v). $ACP''_{\text{drt}} \vdash \partial_H(a) = a \quad \text{if } a \notin H$
- (vi). $ACP''_{\text{drt}} \vdash \partial_H(a) = \delta \quad \text{if } a \in H$

Proof

- (i). $ACP''_{\text{drt}} \vdash a \mid b = [\underline{a}]^\omega \mid [\underline{b}]^\omega = [\underline{a} \mid \underline{b}]^\omega = [\underline{c}]^\omega = c$

- (ii). $ACP''_{drt} \vdash a \mid b \cdot x = \lfloor \underline{a} \rfloor^\omega \mid \lfloor \underline{b} \rfloor^\omega \cdot x = \lfloor \underline{a} \rfloor^\omega \mid \lfloor \underline{b} \cdot x \rfloor^\omega = \lfloor \underline{a} \mid \underline{b} \cdot x \rfloor^\omega = \lfloor (\underline{a} \mid \underline{b}) \cdot x \rfloor^\omega = \lfloor (\underline{a} \mid \underline{b}) \rfloor^\omega \cdot x = (\lfloor \underline{a} \rfloor^\omega \mid \lfloor \underline{b} \rfloor^\omega) \cdot x = (a \mid b) \cdot x$
- (iii). $ACP''_{drt} \vdash a \cdot x \mid b = \lfloor \underline{a} \rfloor^\omega \cdot x \mid \lfloor \underline{b} \rfloor^\omega = \lfloor \underline{a} \cdot x \rfloor^\omega \mid \lfloor \underline{b} \rfloor^\omega = \lfloor \underline{a} \cdot x \mid \underline{b} \rfloor^\omega = \lfloor (\underline{a} \mid \underline{b}) \cdot x \rfloor^\omega = \lfloor (\underline{a} \mid \underline{b}) \rfloor^\omega \cdot x = (\lfloor \underline{a} \rfloor^\omega \mid \lfloor \underline{b} \rfloor^\omega) \cdot x = (a \mid b) \cdot x$
- (iv). $ACP''_{drt} \vdash a \cdot x \mid b \cdot y = \lfloor \underline{a} \rfloor^\omega \cdot x \mid \lfloor \underline{b} \rfloor^\omega \cdot y = \lfloor \underline{a} \cdot x \rfloor^\omega \mid \lfloor \underline{b} \cdot y \rfloor^\omega = \lfloor \underline{a} \cdot x \mid \underline{b} \cdot y \rfloor^\omega = \lfloor (\underline{a} \mid \underline{b}) \cdot (x \parallel y) \rfloor^\omega = \lfloor (\underline{a} \mid \underline{b}) \rfloor^\omega \cdot (x \parallel y) = (\lfloor \underline{a} \rfloor^\omega \mid \lfloor \underline{b} \rfloor^\omega) \cdot (x \parallel y) = (a \mid b) \cdot (x \parallel y)$
- (v). $ACP''_{drt} \vdash \partial_H(a) = \partial_H(\lfloor \underline{a} \rfloor^\omega) = \lfloor \partial_H(\underline{a}) \rfloor^\omega = \lfloor \underline{a} \rfloor^\omega = a$
- (vi). $ACP''_{drt} \vdash \partial_H(a) = \partial_H(\lfloor \underline{a} \rfloor^\omega) = \lfloor \partial_H(\underline{a}) \rfloor^\omega = \lfloor \underline{\delta} \rfloor^\omega = \delta$

■

☞ Note that the equalities of Proposition 5.3.5.4 on the page before correspond precisely to the equalities of Proposition 5.3.3.8 on page 154.

Corollary 5.3.5.5 (Elimination for ACP''_{drt})

Let t be a closed ACP''_{drt} term. Then there is a basic term s such that $ACP''_{drt} \vdash t = s$.

Proof In the same way as Theorem 5.3.3.11 and Corollary 5.3.3.12. Note that all equalities of Propositions 5.3.3.7 and 5.3.3.8 that are used in the proof of Theorem 5.3.3.11 correspond to derivable equalities in ACP''_{drt} . ■

☞ As Axioms USD8 and USD9 have not previously been shown to be derivable for closed terms, we cannot use Proof Outline 4.2.2.2 on page 70 to prove soundness. So, we prove their soundness separately.

Theorem 5.3.5.6 (Soundness of USD8 and USD9)

The set of closed ACP_{drt} terms modulo bisimulation equivalence is a model of Axioms USD8 and USD9.

Proof We use the direct method described in Proof Outline 4.2.2.1 on page 69.

Axiom USD8 Take the following relation:

$$R = \{(s, s), (\lfloor s \mid t \rfloor^\omega, \lfloor s \rfloor^\omega \mid \lfloor t \rfloor^\omega) \mid s \in C(ACP_{drt})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose $\lfloor s \mid t \rfloor^\omega \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
- (a) $s \xrightarrow{b} p_1, t \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. Then $\lfloor s \rfloor^\omega \xrightarrow{b} p_1$ and $\lfloor t \rfloor^\omega \xrightarrow{c} p_2$, so $\lfloor s \rfloor^\omega \mid \lfloor t \rfloor^\omega \xrightarrow{a} p_1 \parallel p_2$, and $(p_1 \parallel p_2, p_1 \parallel p_2) \in R$.
- (b) $s \xrightarrow{b} \surd, t \xrightarrow{c} p_2, \gamma(b, c) = a$, and $p \equiv p_2$. Then $\lfloor s \rfloor^\omega \xrightarrow{b} \surd$ and $\lfloor t \rfloor^\omega \xrightarrow{c} p_2$, so $\lfloor s \rfloor^\omega \mid \lfloor t \rfloor^\omega \xrightarrow{a} p_2$, and $(p_2, p_2) \in R$.
- (c) $s \xrightarrow{b} p_1, t \xrightarrow{c} \surd, \gamma(b, c) = a$, and $p \equiv p_1$. Then $\lfloor s \rfloor^\omega \xrightarrow{b} p_1$ and $\lfloor t \rfloor^\omega \xrightarrow{c} \surd$, so $\lfloor s \rfloor^\omega \mid \lfloor t \rfloor^\omega \xrightarrow{a} p_1$, and $(p_1, p_1) \in R$.

- (ii). Suppose $[s \mid t]^\omega \xrightarrow{a} \surd$. By inspection of the deduction rules we can conclude that $s \xrightarrow{b} \surd$, $t \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. Then we have $[s]^\omega \xrightarrow{b} \surd$ and $[t]^\omega \xrightarrow{c} \surd$, so $[s]^\omega \mid [t]^\omega \xrightarrow{a} \surd$.
- (iii). Suppose $[s \mid t]^\omega \xrightarrow{\sigma} p$. By inspection of the deduction rules we can conclude that $p \equiv [s \mid t]^\omega$. We also have $[s]^\omega \mid [t]^\omega \xrightarrow{\sigma} [s]^\omega \mid [t]^\omega$, and $([s \mid t]^\omega, [s]^\omega \mid [t]^\omega) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose $[s]^\omega \mid [t]^\omega \xrightarrow{a} p$. By inspection of the deduction rules we distinguish the following cases:
 - (a) $s \xrightarrow{b} p_1$, $t \xrightarrow{c} p_2$, $\gamma(b, c) = a$, and $p \equiv p_1 \parallel p_2$. Then $[s \mid t]^\omega \xrightarrow{c} p_1 \parallel p_2$, and $(p_1 \parallel p_2, p_1 \parallel p_2) \in R$.
 - (b) $s \xrightarrow{b} \surd$, $t \xrightarrow{c} p_2$, $\gamma(b, c) = a$, and $p \equiv p_2$. Then $[s \mid t]^\omega \xrightarrow{a} p_2$, and $(p_2, p_2) \in R$.
 - (c) $s \xrightarrow{b} p_1$, $t \xrightarrow{c} \surd$, $\gamma(b, c) = a$, and $p \equiv p_1$. Then $[s \mid t]^\omega \xrightarrow{a} p_1$, and $(p_1, p_1) \in R$.
- (ii). Suppose $[s]^\omega \mid [t]^\omega \xrightarrow{a} \surd$. By inspection of the deduction rules we can conclude that $s \xrightarrow{b} \surd$, $t \xrightarrow{c} \surd$, and $\gamma(b, c) = a$. Then $[s \mid t]^\omega \xrightarrow{a} \surd$, and we are done.
- (iii). Suppose $[s]^\omega \mid [t]^\omega \xrightarrow{\sigma} p$. By inspection of the deduction rules we can conclude that $p \equiv [s]^\omega \mid [t]^\omega$. We also have $[s \mid t]^\omega \xrightarrow{\sigma} [s \mid t]^\omega$, and $([s \mid t]^\omega, [s]^\omega \mid [t]^\omega) \in R$.

Finally, we look at the immediate-deadlock predicate. Neither side has immediate deadlock: $\neg \text{ID}([s \mid t]^\omega)$ and $\neg \text{ID}([s]^\omega \mid [t]^\omega)$ (note that unbounded start delay removes immediate deadlock, see the comment on page 57).

Axiom USD9 Take the following relation:

$$R = \{(s, s), ([\partial_H(s)]^\omega, \partial_H([X]^\omega)) \mid s \in C(\text{ACP}_{\text{drt}})\}$$

First we look at the transitions of the left-hand side:

- (i). Suppose $[\partial_H(s)]^\omega \xrightarrow{a} p$. Then necessarily we have $s \xrightarrow{a} p_1$, $a \notin H$, and $p \equiv \partial_H(p_1)$. Therefore $[s]^\omega \xrightarrow{a} p_1$, and also $\partial_H([s]^\omega) \xrightarrow{a} \partial_H(p_1)$, and note that $(\partial_H(p_1), \partial_H(p_1)) \in R$.
- (ii). Suppose $[\partial_H(s)]^\omega \xrightarrow{a} \surd$. Then necessarily we have $s \xrightarrow{a} \surd$ and $a \notin H$. Therefore $[s]^\omega \xrightarrow{a} \surd$, and also $\partial_H([s]^\omega) \xrightarrow{a} \surd$.
- (iii). Suppose $[\partial_H(s)]^\omega \xrightarrow{\sigma} p$. Then necessarily we have $p \equiv [\partial_H(s)]^\omega$. Furthermore, we have $\partial_H([s]^\omega) \xrightarrow{\sigma} \partial_H([s]^\omega)$, and note that $([\partial_H(s)]^\omega, \partial_H([s]^\omega)) \in R$.

Secondly, we look at the transitions of the right-hand side:

- (i). Suppose $\partial_H([s]^\omega) \xrightarrow{a} p$. Then necessarily we have $s \xrightarrow{a} p_1$, $a \notin H$, and $p \equiv \partial_H(p_1)$. Therefore $\partial_H(s) \xrightarrow{a} \partial_H(p_1)$, and also $[\partial_H(s)]^\omega \xrightarrow{a} \partial_H(p_1)$, and note that $(\partial_H(p_1), \partial_H(p_1)) \in R$.
- (ii). Suppose $\partial_H([s]^\omega) \xrightarrow{a} \surd$. Then necessarily we have $s \xrightarrow{a} \surd$ and $a \notin H$. Therefore $\partial_H(s) \xrightarrow{a} \surd$, and also $[\partial_H(s)]^\omega \xrightarrow{a} \surd$.
- (iii). Suppose $\partial_H([s]^\omega) \xrightarrow{\sigma} p$. Then necessarily we have $p \equiv \partial_H([s]^\omega)$. Furthermore, we have $[\partial_H(s)]^\omega \xrightarrow{\sigma} [\partial_H(s)]^\omega$, and note that $([\partial_H(s)]^\omega, \partial_H([s]^\omega)) \in R$.

Finally, we look at the immediate-deadlock predicate. Neither side has immediate deadlock: $\neg \text{ID}(\lfloor \partial_H(s) \rfloor^\omega)$ and $\neg \text{ID}(\partial_H(\lfloor s \rfloor^\omega))$.

■

Corollary 5.3.5.7 (Soundness of $\text{ACP}_{\text{drt}}''$)

The set of closed $\text{ACP}_{\text{drt}}''$ terms modulo bisimulation equivalence is a model of $\text{ACP}_{\text{drt}}''$.

Proof We use the indirect method of Proof Outline 4.2.2.2 on page 70. The result follows directly from the soundness of $\text{ACP}_{\text{drt}}^+$ (see Theorem 5.3.3.14 on page 165) combined with the facts that Axioms USD1–USD7 are derivable in $\text{ACP}_{\text{drt}}^+$ (see Proposition 5.3.3.7 on page 154 and Proposition 5.3.3.8 on page 154), and that Axioms USD8 and USD9 are sound (see Theorem 5.3.5.6 on page 170). ■

Corollary 5.3.5.8 (Completeness of $\text{ACP}_{\text{drt}}''$)

The axiom system $\text{ACP}_{\text{drt}}''$ is a complete axiomatization of the set of closed $\text{ACP}_{\text{drt}}''$ terms modulo bisimulation equivalence.

Proof We use the indirect method of Proof Outline 4.2.3.2 on page 71. Careful inspection of the dependencies between the proofs in this section reveals that the proof of Theorem 5.3.3.17 only relies upon RSP(USD) to ensure Proposition 5.3.3.7(i)–(vii) and Proposition 5.3.3.8(i)–(vi). So, we obviously do not need RSP(USD) anymore if we add the Axioms USD1–USD9 from which the needed properties are derivable (see Proposition 5.3.5.4 on page 169), and the result follows. ■

5.4 Properties

In this section, we list (without proof) a number of properties of the process algebras we have treated in this chapter.

Remark 5.4.1.1 (Axioms of Standard Concurrency)

For all process algebras described in this chapter, the axioms of standard concurrency (see Property 2.6.1.11 on page 37) hold.

Remark 5.4.1.2 (Time Determinism)

For all process algebras described in this chapter, the time-determinism property (see Property 3.3.1.1 on page 65) holds.

Remark 5.4.1.3 (Embeddings)

The following embeddings hold between the process algebras given in this chapter, and between them and the process algebras given in previous chapters:

- (i). $\text{PA}_\delta \subseteq \text{PA}_{\text{drt}}^- \text{-ID}$
- (ii). $\text{PA}_\delta \subseteq \text{PA}_{\text{drt}}$
- (iii). $\text{PA}_{\text{drt}}^- \text{-ID} \subseteq \text{PA}_{\text{drt}} \subseteq \text{PA}_{\text{drt}}^+$
- (iv). $\text{PA}_{\text{drt}}^- \text{-ID} \subseteq \text{PA}_{\text{drt}}^- \text{-ID}'$, $\text{PA}_{\text{drt}}^- \text{-ID}' \subseteq \text{PA}_{\text{drt}}^- \text{-ID}$

- (v). $PA_{\text{drt}}^+ \subseteq PA'_{\text{drt}}, PA'_{\text{drt}} \subseteq PA_{\text{drt}}^+$
- (vi). $ACP \subseteq ACP_{\text{drt}}^- \text{-ID}$
- (vii). $ACP_{\delta} \subseteq ACP_{\text{drt}}$
- (viii). $ACP_{\text{drt}}^- \text{-ID} \subseteq ACP_{\text{drt}} \subseteq ACP_{\text{drt}}^+$
- (ix). $ACP_{\text{drt}}^- \text{-ID} \subseteq ACP_{\text{drt}}^- \text{-ID}', ACP_{\text{drt}}^- \text{-ID}' \subseteq ACP_{\text{drt}}^- \text{-ID}$
- (x). $ACP_{\text{drt}}^+ \subseteq ACP'_{\text{drt}}, ACP'_{\text{drt}} \subseteq ACP_{\text{drt}}^+$
- (xi). $ACP_{\text{drt}}^+ \subseteq ACP''_{\text{drt}}, ACP''_{\text{drt}} \subseteq ACP_{\text{drt}}^+$
- (xii). $ACP'_{\text{drt}} \subseteq ACP''_{\text{drt}}, ACP''_{\text{drt}} \subseteq ACP'_{\text{drt}}$
- (xiii). $BPA_{\text{drt}}^- \text{-ID} \subseteq PA_{\text{drt}}^- \text{-ID} \subseteq ACP_{\text{drt}}^- \text{-ID}$
- (xiv). $BPA_{\text{drt}} \subseteq PA_{\text{drt}} \subseteq ACP_{\text{drt}}$
- (xv). $BPA_{\text{drt}}^+ \subseteq PA_{\text{drt}}^+ \subseteq ACP_{\text{drt}}^+$

The embeddings of (i) and (vi) are achieved by projecting the untimed process a onto the undelayable process \underline{a} for $a \in A_{\delta}$, and everything else onto itself.

The embeddings of (ii) and (vii) can be achieved in two different ways: either by projecting the untimed process a onto the undelayable process \underline{a} for $a \in A_{\delta}$, and everything else onto itself, or by projecting the untimed process a onto the delayable process a for $a \in A_{\delta}$, and everything else onto itself.

All other embeddings are achieved by projecting everything onto itself.

5.5 Conclusions

To begin with, we are reasonably confident that the process algebras listed in Chapter 4 and Chapter 5 are sound and complete.

Before we started this work, we were also confident of the soundness and completeness of these axiomatizations, but at that time wrongly so. We discovered that the axiomatizations we started out with, most of which have been published and claimed sound and complete before, were neither sound nor complete. We highlight two characteristic cases:

- Weakening Axiom DRT4A to Axiom DRT4 brings the need to introduce Axiom DRT5, something we did not realize at first. This left all interesting process algebras incomplete. Only when Axiom DRT5 was needed in the proof of Lemma 4.3.4.7(iv), we found out about this mistake.
- Introducing the immediate deadlock in a context that supports communication brings the need to weaken Axiom DRTCM6 to Axiom DRTCM6ID, something we did not realize due to the “intuitive” and “obvious” nature of DRTCM6. This left some theories unsound. We found out this problem after we could not complete the last few “trivial details” of the proof of Theorem 5.3.3.14.

Both these problems were discovered when we were writing out all details of “trivial” proofs, proofs which we had originally not planned to do at all. So, we eventually decided to give as much and as detailed proofs as reasonably manageable. And, as to be expected, we found some more mistakes like the ones listed above. As a side-result, we gained insight into the various aspects of axiomatizations.

Firstly, when we weigh the merits of the “ ν/σ -axiomatization style” of BAETEN AND RENIERS [35] (with theories like PA_{drt}^- -ID and $\text{ACP}_{\text{drt}}^-$ -ID) against those of the “classic axiomatization style” of BAETEN AND BERGSTRA [24] (with theories like PA_{drt}^- -ID' and $\text{ACP}_{\text{drt}}^-$ -ID'), we conclude that the ν/σ -style is better suited towards practical applications, as it makes calculations easier. However, from a theoretical viewpoint this style is troublesome: it does not lend itself well to term-rewriting system analysis, and worse, it does not seem to be compatible with the addition of the empty process, as we will show in Chapter 6. On the other hand, the classic style is not ideal either. It appears less intuitive, and it needs more axioms. Compare for example Axioms DRTM5 and DRTM6 of Table 5.2 on page 106 with Axioms DRTM7–DRTM11 of Table 5.4 on page 113. Here the classic style needs five axioms to do what the ν/σ -style can do much clearer in two axioms. Consequently, calculations in the classic style are much longer too.

Second, we have shown how to eliminate the recursion principle $\text{RSP}(\text{USD})$ from the process algebras that contain it. As shown in Section 4.3.5, Section 5.3.2, and Section 5.3.4, one can straightforwardly derive unconditional axioms to replace the conditional axiom $\text{RSP}(\text{USD})$. The recipe is always the same: identify in the completeness proof of the conditional theory the places where $\text{RSP}(\text{USD})$ is used, put those applications in a separate lemma, and introduce an axiom for every clause of that lemma. Using this recipe, we introduced Axioms USD1–USD7, USDCF, and USDCM2–USDCM4. The advantage is clear: having a fully unconditional axiomatization enables us to reason fully algebraically, giving us a fuller apparatus of methods to work with. On the other hand, the principle $\text{RSP}(\text{USD})$ is clean, neat, and simple, and can be applied in any process algebra, while the “USD axiomatization style” requires new axioms for every new process algebra. That this can lead to unwieldy theories can be observed from BAETEN AND BERGSTRA [24]. We feel that the “ $\text{RSP}(\text{USD})$ axiomatization style”, which till now has only appeared in BAETEN AND BERGSTRA [23], BAETEN AND RENIERS [35], and RENIERS AND VEREIJKEN [180], deserves a wider audience.

Third, we find that the absence of the *empty process* (a constant $\underline{\underline{\varepsilon}}$ such that $\underline{\underline{\varepsilon}} \cdot x = x = x \cdot \underline{\underline{\varepsilon}}$) in the discrete-time process algebras we have treated so far, is a major nuisance. To begin with, the empty process would allow us to express our axioms much more compactly. For example, Axiom DRTM2 would be just an instance of Axiom DRTM3 if the x in the latter could take the value $\underline{\underline{\varepsilon}}$. Similarly, Axioms DRTCM2–DRTCM4 could also be collapsed to just one axiom with the help of $\underline{\underline{\varepsilon}}$ (and see BAETEN AND BERGSTRA [26] for a process algebra in which it is hard to find *any* axiom that could not be formulated better with the help of the empty process!). The absence of $\underline{\underline{\varepsilon}}$ is even felt worse when doing calculations. If we look for example at the proof of Theorem 5.3.3.11 on page 159, we see we have to distinguish 49 cases(!) when doing simultaneous induction on two variables, as a basic term in ACP_{drt} can take seven essentially different forms. With the help of $\underline{\underline{\varepsilon}}$, we could reduce this to five forms, and only 25 cases would have to be considered when doing induction on two variables. Similar considerations hold for the proof of Theorem 5.2.3.10 on page 127, where the absence of $\underline{\underline{\varepsilon}}$ in one case even forces us into a *sixteen-fold* increase in proof obligations.

We conclude that there is a clear need for the empty process in discrete-time process algebra. In Chapter 6 we will introduce the empty process in the setting of $\text{BPA}_{\text{drt}}\text{-ID}$, and examine some of the issues that arise.

Then, we hope that Chapter 3, Chapter 4, and Chapter 5 can serve as a reference point: to our knowledge this is the first work that list so many discrete-time process algebra theories, together with all relevant definitions and elementary theorems. Furthermore, as all proofs in these chapters are constructive, it should now be easy to develop a tool that can automatically rewrite two bisimilar ACP_{drt} terms into one another.

Finally, note that we have surveyed several distinct methods for proving soundness and completeness. To prove soundness we have used:

- the direct method (see Proof Outline 4.2.2.1 on page 69),
- the indirect method (see Proof Outline 4.2.2.2 on page 70), and,
- the ground equivalence method (see Proof Outline 4.2.2.3 on page 70).

To prove completeness we have used:

- the direct method (see Proof Outline 4.2.3.1 on page 70),
- the indirect method (see Proof Outline 4.2.3.2 on page 71),
- the ground equivalence method (see Proof Outline 4.2.3.3 on page 71), and,
- Verhoef's method (see Proof Outline 4.2.3.4 on page 71).

We believe that this spectrum of methods provides a convenient starting point to prove soundness and completeness of most (timed) process algebras that have been described in the literature, with the exception of theories that support abstraction.

As far as future research is concerned: we would like to generalize our results to a setting that includes abstraction. This seems however not at all trivial, and may require a substantial effort. In BAETEN, BERGSTRA, AND RENIERS [29], a first attempt is made at proving soundness and completeness for discrete-time abstract process algebras.

6

Adding the Empty Process

6.1 Introduction

One of the main features of process algebra is its modularity: it is relatively easy to add features for specific purposes, and, given two of such extensions, it is often straightforward to combine them into one process algebra.

In the past, the feature of the *empty process* (see Section 2.3.3), the process that does nothing, and terminates successfully (in contrast with the *deadlock process*, which does nothing, and terminates unsuccessfully), has been studied extensively (BAETEN AND VAN GLABBEEK [31], KOYMANS AND VRANCKEN [122], and VRANCKEN [196, 197]).

As already mentioned in the previous chapters, recently there has been described an elegant and coherent way to incorporate *discrete-time extensions* into the process-algebra framework (BAETEN AND BERGSTRA [21, 24, 25], BAETEN AND RENIERS [35], RENIERS AND VEREIJKEN [180]), and a case study has been made (BOS AND RENIERS [53]).

Given the fact that both extensions, empty process and discrete time, have been researched very well, it seemed reasonable to study how these two extensions go together. This is especially important, as the empty process does have great advantages in the specification of protocols. For example, a stack over an infinite data type can only be specified in finitely many equations (three, to be exact), when we have the empty process at our disposal (KOYMANS AND VRANCKEN [122]). Furthermore, the empty process is needed to give a process-algebra based semantics to the specification languages SDL (see ITU-T [102]) and MSC (see ITU-TS [103, 104, 105, 106], and the dissertation of RENIERS [179]), and hence a timed empty process is needed to give such a semantics to timed variants of these languages.

But, as the introduction of the empty process has historically often been viewed as troublesome, the modular combination of it with other features has hardly been studied. In this chapter, we remedy this situation for discrete-time process algebra with relative timing.

6.2 Design Goals for the Discrete-Time Empty Process

Often, when extending an already existing process algebra with some additional constant or operator, one has a lot of “maneuvering room” to make choices regarding the exact implementation and the fine details of its behavior. For example, a very broad class of “deadlock-like” processes has been described in the literature. To name a few variants (in order of increasing degree of catastrophe): “classic” (δ , see BAETEN AND WEIJLAND [38] or Section 2.3.2 of this thesis), “immediate” ($\delta^{\dot{}}$, see BAETEN AND BERGSTRA [24, 25] or Section 2.3.5 of this thesis), “inconsistent state” (\perp , see BAETEN AND BERGSTRA [26]), and “true zero” (0 , see BAETEN AND BERGSTRA [14, 19]). A similar observation holds for the implementation of the concept “choice”: half a dozen versions, ranging from completely deterministic to completely non-deterministic have been described.

It appears that the concept “empty process” is different; it leaves little room for differing implementations.

When we started thinking about the process that “does nothing” and terminates successfully (in the context of discrete-time process algebra), we had three design goals in mind for the empty process:

- (i). it should be a unit element with respect to the \cdot and \parallel operators,
- (ii). it ought not to destroy the commutativity and associativity of the \parallel and $|$ operators,
- (iii). it ought not to destroy the weak time-factorization property.

These properties we deemed essential; violating one of them would render our empty process useless. And, as it turned out, these goals very much restricted our freedom in choosing our definitions. In this chapter, we will motivate our choices on the grounds of the above design goals.

Finally, we had a few other design goals of lesser importance; we wanted that the existing “smaller” process algebras (without empty process, without time, or without either) could be embedded in the new process algebras in a simple way (i.e., in the terminology of BAETEN AND BERGSTRA [19], the old process algebras should be *Subalgebras of a Reduced Model* (SRM’s) of the new process algebras).

6.3 Process Algebras with Undelayable Actions

We introduce the process algebras $BPA_{\text{drt},\varepsilon}^-$ -ID, $PA_{\text{drt},\varepsilon}^-$ -ID, and $ACP_{\text{drt},\varepsilon}^-$ -ID that contain the empty process. These are all based on BPA_{drt}^- -ID, and hence do not contain delayable actions or the immediate deadlock.

6.3.1 $BPA_{\text{drt},\varepsilon}^-$ -ID

In this section, we define the process algebra $BPA_{\text{drt},\varepsilon}^-$ -ID, which is basically the process algebra BPA_{drt}^- -ID of Section 3.2.2, extended with the empty process (indicated by the subscript “ ε ”).

Definition 6.3.1.1 (Signature of $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID)

The signature of $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *time-unit delay constant* $\underline{\sigma}$, the *undelayable empty process constant* $\underline{\varepsilon}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , and the “now” operator ν_{rel} .

Remark 6.3.1.2 (The Undelayable Empty Process and the Time-Unit Delay Constant)

The *undelayable empty process* $\underline{\varepsilon}$ in the signature of $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID has the intuitive meaning of doing nothing, and then terminating successfully. As such, it is a proper unit element of the sequential composition: $\underline{\varepsilon} \cdot x = x \cdot \underline{\varepsilon} = x$.

Now that we have a unit element, we do not need the time-unit delay *operator* anymore. This is because we can replace it by the time-unit delay *constant* $\underline{\sigma}$ (intuitively: $\underline{\sigma} = \sigma_{\text{rel}}(\underline{\varepsilon})$), the process that can move on to the following time-slice, and terminate there. The process formerly expressed as $\sigma_{\text{rel}}(x)$ can now be represented by $\underline{\sigma} \cdot x$, as, intuitively, $\sigma_{\text{rel}}(x) = \sigma_{\text{rel}}(\underline{\varepsilon} \cdot x) = \sigma_{\text{rel}}(\underline{\varepsilon}) \cdot x = \underline{\sigma} \cdot x$.

Definition 6.3.1.3 (Axioms of $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID)

The process algebra $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID is axiomatized by the axioms of BPA given in Definition 2.3.1.6 on page 8, and Axioms DRTE1-DRTE4, TF, and DCSE1-DCSE4 shown in Table 6.1: $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID = A1-A5 + DRTE1-DRTE4 + TF + DCSE1-DCSE4.

$x + \underline{\delta} = x$	DRTE1
$\underline{\delta} \cdot x = \underline{\delta}$	DRTE2
$x \cdot \underline{\varepsilon} = x$	DRTE3
$\underline{\varepsilon} \cdot x = x$	DRTE4
$\underline{\sigma} \cdot x + \underline{\sigma} \cdot y = \underline{\sigma} \cdot (x + y)$	TF
$\nu_{\text{rel}}(\underline{\varepsilon}) = \underline{\varepsilon}$	DCSE1
$\nu_{\text{rel}}(x + y) = \nu_{\text{rel}}(x) + \nu_{\text{rel}}(y)$	DCSE2
$\nu_{\text{rel}}(\underline{a} \cdot x) = \underline{a} \cdot x$	DCSE3
$\nu_{\text{rel}}(\underline{\sigma} \cdot x) = \underline{\delta}$	DCSE4

Table 6.1: Additional axioms for $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID.

☞ The undelayable deadlock behaves as expected: Axioms DRTE1 and DRTE2 are the discrete-time counterparts of Axioms A6 and A7 of Table 2.5 on page 14. Axioms DRTE3 and DRTE4 express that the undelayable empty process is a unit element with respect to the sequential composition.

When we compare $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID with $\text{BPA}_{\text{drt}}^-$ -ID, the presence of the undelayable empty process allows for two simplifications in the axioms. First, to express the weak time-factorization property associated with $\underline{\sigma}$, formerly done by Axiom DRT1, we now use Axiom TF. Secondly, the purpose of Axiom DRT2 disappears, as it becomes a special case of Axiom A5, the associativity of sequential composition: $(\underline{\sigma} \cdot x) \cdot y = \underline{\sigma} \cdot (x \cdot y)$.

Finally, Axioms DCSE1–DCSE4 express the properties of the “now” operator in a context with the undelayable empty process. Note that we do not anymore have that $\nu(x \cdot y) = \nu(x) \cdot y$, as that would lead to $\nu(\underline{\sigma}) = \nu(\underline{\varepsilon} \cdot \underline{\sigma}) = \nu(\underline{\varepsilon}) \cdot \underline{\sigma} = \underline{\varepsilon} \cdot \underline{\sigma} = \underline{\sigma}$, which is of course undesirable. Otherwise, Axioms DCSE1–DCSE4 behave very much like their $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID counterparts DCS1–DCS4 from Table 3.4 on page 47.

Definition 6.3.1.4 (Semantics of $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID)

The semantics of $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID are given by the term-deduction system $T(\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID})$, induced by the deduction rules for $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID shown in Table 6.2.

$\underline{a} \xrightarrow{a} \underline{\varepsilon}$	$\underline{\sigma} \xrightarrow{\sigma} \underline{\varepsilon}$	$\underline{\varepsilon} \downarrow$	
$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'}$	$\frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$	$\frac{x \downarrow}{(x + y) \downarrow}$	$\frac{y \downarrow}{(x + y) \downarrow}$
$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y}$	$\frac{x \downarrow, y \xrightarrow{a} y'}{x \cdot y \xrightarrow{a} y'}$	$\frac{x \downarrow, y \downarrow}{(x \cdot y) \downarrow}$	
$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} x' + y'}$	$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} x'}{x + y \xrightarrow{\sigma} x'}$	$\frac{x \xrightarrow{\sigma}, y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} y'}$	
$\frac{x \xrightarrow{\sigma} x', x \downarrow}{x \cdot y \xrightarrow{\sigma} x' \cdot y}$	$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma}}{x \cdot y \xrightarrow{\sigma} x' \cdot y}$	$\frac{x \xrightarrow{\sigma}, x \downarrow, y \xrightarrow{\sigma} y'}{x \cdot y \xrightarrow{\sigma} y'}$	$\frac{x \xrightarrow{\sigma} x', x \downarrow, y \xrightarrow{\sigma} y'}{x \cdot y \xrightarrow{\sigma} x' \cdot y + y'}$
$\frac{x \xrightarrow{a} x'}{\nu_{\text{rel}}(x) \xrightarrow{a} x'}$	$\frac{x \downarrow}{\nu_{\text{rel}}(x) \downarrow}$		

Table 6.2: Deduction rules for $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID.

☞ Many of these deduction rules are already familiar from either BPA_{ε} or $\text{BPA}_{\text{drt}}^-$ -ID, so we need not elaborate on them. The deduction rules for the sequential composition with respect to time transitions, shown in Table 6.2, however, are very subtle. We will try to provide the intuition behind them with an example.

Consider the expression $(\underline{\sigma} \cdot \underline{a} + \underline{\varepsilon}) \cdot (\underline{\sigma} \cdot \underline{b})$. Algebraically, we have:

$$\begin{aligned}
 (\underline{\sigma} \cdot \underline{a} + \underline{\varepsilon}) \cdot (\underline{\sigma} \cdot \underline{b}) &= \underline{\sigma} \cdot \underline{a} \cdot \underline{\sigma} \cdot \underline{b} + \underline{\varepsilon} \cdot \underline{\sigma} \cdot \underline{b} \\
 &= \underline{\sigma} \cdot \underline{a} \cdot \underline{\sigma} \cdot \underline{b} + \underline{\sigma} \cdot \underline{b} \\
 &= \underline{\sigma} \cdot (\underline{a} \cdot \underline{\sigma} \cdot \underline{b} + \underline{b})
 \end{aligned}$$

therefore, in our model we should have $(\underline{\sigma} \cdot \underline{a} + \underline{\varepsilon}) \cdot (\underline{\sigma} \cdot \underline{b}) \xrightarrow{\sigma} \underline{a} \cdot \underline{\sigma} \cdot \underline{b} + \underline{b}$ and in particular we should *not* have $(\underline{\sigma} \cdot \underline{a} + \underline{\varepsilon}) \cdot (\underline{\sigma} \cdot \underline{b}) \xrightarrow{\sigma} \underline{a} \cdot \underline{\sigma} \cdot \underline{b}$, as such a transition would lose the option

to do just a b , while the principle of weak time factorization mentioned before explicitly forbids that a time transition determines a choice here.

To obtain such behavior in our term-deduction system, we need four rules to define the interaction between time transitions and the sequential composition. The first two express that if x can do a σ -step to x' , then $x \cdot y$ can do a σ -step to $x' \cdot y$, *provided* either x does not have option to terminate, or y cannot do a σ -step. The fourth rule expresses that if this provision is not met, we get the behavior as sketched in the example above. These three rules together cover all cases where x can do a σ -step.

Remains the case that x cannot do a σ -step. If that is so, and x also does not have the option to terminate, then obviously $x \cdot y$ cannot do a σ -step, and hence there is no rule for this case. If x does have the option to terminate, and y can do a σ -step to y' , then $x \cdot y$ can do a σ -step to y' , as is expressed by the third rule. Finally, if x does have the option to terminate, but y cannot do a σ -step, then again $x \cdot y$ cannot do a σ -step, so there is no rule for that case either.

Theorem 6.3.1.5 (Time Determinism for $BPA_{\text{drt},\varepsilon}^-$ -ID)

Let x , y , and y' be closed $BPA_{\text{drt},\varepsilon}^-$ -ID terms. Then we have:

$$T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y' \implies y \equiv y'$$

Proof Suppose that $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y'$. We proceed by induction on the number of symbols in x , using case distinction on the form of x . For every case we either derive, by inspection of the deduction rules, that $y \equiv y'$, or arrive at a contradiction, indicating that the case under consideration does not occur.

- (i). $x \equiv \underline{\varepsilon}$. In contradiction with $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y$.
- (ii). $x \equiv \underline{a}$ for some $a \in A_\delta$. In contradiction with $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y$.
- (iii). $x \equiv \underline{\sigma}$. Then $y \equiv y' \equiv \underline{\varepsilon}$ (this is the base case of the induction).
- (iv). $x \equiv s \cdot t$ for closed $BPA_{\text{drt},\varepsilon}^-$ -ID terms s and t . We proceed by case distinction on the transitions of s and t .
 - (a) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', s \downarrow, t \xrightarrow{\sigma} t'$ (where, by the induction hypothesis, s' and t' are unique). Then $y \equiv y' \equiv s' \cdot t + t'$.
 - (b) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s'$ otherwise (where s' is unique). Then $y \equiv y' \equiv s' \cdot t$.
 - (c) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s, s \downarrow, t \xrightarrow{\sigma} t'$ (where t' is unique). Then $y \equiv y' \equiv t'$.
 - (d) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s$ otherwise. In contradiction with $x \xrightarrow{\sigma} y$.
- (v). $x \equiv s + t$ for closed $BPA_{\text{drt},\varepsilon}^-$ -ID terms s and t . We proceed by case distinction on the transitions of s and t .
 - (a) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', t \xrightarrow{\sigma} t'$ (where s' and t' are unique). Then $y \equiv y' \equiv s' + t'$.
 - (b) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', t \xrightarrow{\sigma} s$ (where s' is unique). Then $y \equiv y' \equiv s'$.
 - (c) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s, t \xrightarrow{\sigma} t'$ (where t' is unique). Then $y \equiv y' \equiv t'$.
 - (d) $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s, t \xrightarrow{\sigma} s$. In contradiction with $x \xrightarrow{\sigma} y$.
- (vi). $x \equiv \nu(s)$ for a closed $BPA_{\text{drt},\varepsilon}^-$ -ID term s . In contradiction with $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y$.

Having inspected all possible cases, we may now conclude that $y \equiv y'$. ■

Definition 6.3.1.6 (Bisimulation for $BPA_{\text{drt},\varepsilon}^-$ -ID)

Bisimulation for $BPA_{\text{drt},\varepsilon}^-$ -ID is defined as follows; a binary relation R on process terms is a bisimulation iff the following transfer conditions hold for all process terms p and q :

- (i). If $R^S(p, q)$ and $p \xrightarrow{u} p'$, where $u \in A_\sigma$, then there exists a process term q' such that $q \xrightarrow{u} q'$ and $R^S(p', q')$,
- (ii). If $R^S(p, q)$ and $p \downarrow$, then $q \downarrow$.

Definition 6.3.1.7 (Bisimulation Model for $BPA_{\text{drt},\varepsilon}^-$ -ID)

The bisimulation model for $BPA_{\text{drt},\varepsilon}^-$ -ID is defined in the same way as for BPA. Replace “BPA” by “ $BPA_{\text{drt},\varepsilon}^-$ -ID” in Definition 2.3.1.16 on page 12.

Definition 6.3.1.8 (Basic Terms of $BPA_{\text{drt},\varepsilon}^-$ -ID)

We define $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic terms inductively as follows:

- (i). The constant $\underline{\varepsilon}$ is a $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic term,
- (ii). if $a \in A_\delta$ and t is a $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic term, then $\underline{a} \cdot t$ is a $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic term,
- (iii). if s and t are $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic terms, then $s + t$ is a $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic term,
- (iv). if t is a $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic term, then $\underline{\sigma} \cdot t$ is a $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic term.

From now on, when we speak of basic terms in the context of $BPA_{\text{drt},\varepsilon}^-$ -ID, we mean $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic terms.

Example 6.3.1.9 (Basic Terms of $BPA_{\text{drt},\varepsilon}^-$ -ID)

The following are basic terms of $BPA_{\text{drt},\varepsilon}^-$ -ID: $\underline{\delta} \cdot \underline{\varepsilon}, \underline{a} \cdot \underline{\varepsilon}, \underline{a} \cdot (\underline{b} \cdot \underline{c} \cdot \underline{\varepsilon} + \underline{d} \cdot \underline{\varepsilon})$. Note, however, that the following are *not* basic terms: $\underline{\delta}, \underline{a}, \underline{a} \cdot (\underline{b} + \underline{c})$.

Definition 6.3.1.10 (Number of Symbols of a $BPA_{\text{drt},\varepsilon}^-$ -ID Term)

We define $n(x)$, the number of symbols of x , inductively as follows:

- (i). We define $n(\underline{\varepsilon}) = n(\underline{\sigma}) = 1$,
- (ii). for $a \in A_\delta$, we define $n(\underline{a}) = 1$,
- (iii). for closed $BPA_{\text{drt},\varepsilon}^-$ -ID terms x and y , we define $n(x + y) = n(x \cdot y) = n(x) + n(y) + 1$,
- (iv). for a closed $BPA_{\text{drt},\varepsilon}^-$ -ID term x we define $n(\nu_{\text{rel}}(x)) = n(x) + 1$.

Theorem 6.3.1.11 (General Form of Basic Terms of $BPA_{\text{drt},\varepsilon}^-$ -ID)

Modulo the commutativity and associativity of the $+$, all basic terms t of $BPA_{\text{drt},\varepsilon}^-$ -ID are of the form:

$$t \equiv \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{\sigma} \cdot u_j + \sum_{k < p} \underline{\varepsilon}$$

for $m, n, p \in \mathbb{N}$, $m + n + p \geq 1$, $a_i \in A_\delta$, and basic terms s_i and u_j .

Proof Trivial, by inspection of the definition of basic terms, Definition 6.3.1.8. Observe that the general form of basic terms is closed under the formation rules given in Definition 6.3.1.8. Note that by our summation convention, Definition 3.2.2.7 on page 49, the above general form never contains a summand $\underline{\delta}$. ■

Lemma 6.3.1.12 (Representation of $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID Terms)

Let t be a basic term. Then either $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash t = \nu(t)$, or there exists a basic term s such that $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash t = \nu(t) + \underline{\sigma} \cdot s$ and $n(s) < n(t)$.

Proof Let t be a basic term. By Theorem 6.3.1.11, we may now proceed by case analysis on the general form of basic terms:

(i). Either we have no $\underline{\sigma} \cdot u_j$ summands ($n = 0$ in Definition 6.3.1.11):

$$t \equiv \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{k < p} \underline{\varepsilon}$$

for $m, p \in \mathbb{N}$, $m + p \geq 1$, $a_i \in A_\delta$, and basic terms s_i . Then we have the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash t &= \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{k < p} \underline{\varepsilon} = \sum_{i < m} \nu(\underline{a}_i \cdot s_i) + \sum_{k < p} \nu(\underline{\varepsilon}) = \\ &= \nu \left(\sum_{i < m} \underline{a}_i \cdot s_i + \sum_{k < p} \underline{\varepsilon} \right) = \nu(t) \end{aligned}$$

(ii). Or we have at least one $\underline{\sigma} \cdot u_j$ summand ($n \geq 1$ in Definition 6.3.1.11):

$$t \equiv \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{\sigma} \cdot u_j + \sum_{k < p} \underline{\varepsilon}$$

for $m, n, p \in \mathbb{N}$, $m + n + p \geq 1$, $a_i \in A_\delta$, and basic terms s_i and u_j . Then we have the following computation:

$$\begin{aligned} \text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash t &= \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{\sigma} \cdot u_j + \sum_{k < p} \underline{\varepsilon} = \\ &= \sum_{i < m} \underline{a}_i \cdot s_i + \underline{\delta} + \sum_{k < p} \underline{\varepsilon} + \sum_{j < n} \underline{\sigma} \cdot u_j = \\ &= \sum_{i < m} \nu(\underline{a}_i \cdot s_i) + \sum_{j < n} \nu(\underline{\sigma} \cdot u_j) + \sum_{k < p} \nu(\underline{\varepsilon}) + \underline{\sigma} \cdot \left(\sum_{j < n} u_j \right) = \\ &= \nu \left(\sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{\sigma} \cdot u_j + \sum_{k < p} \underline{\varepsilon} \right) + \underline{\sigma} \cdot \left(\sum_{j < n} u_j \right) = \\ &= \nu(t) + \underline{\sigma} \cdot s \end{aligned}$$

Where we define:

$$s \equiv \sum_{j < n} u_j$$

Note that $n(s) < n(t)$ is now trivially satisfied, as for every summand u_j of s , there is a corresponding summand $\underline{\sigma} \cdot u_j$ of t , and at least one such summand exists as $n \geq 1$.

■

☞ The main use of Lemma 6.3.1.12 will be in induction proofs regarding the (not yet treated) process algebras $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$ and $\text{ACP}_{\text{drt},\varepsilon}^- \text{-ID}$ (see Sections 6.3.2 and 6.3.3).

Definition 6.3.1.13 (Alternative Normal Form for $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$)

We define the *alternative normal form (ANF) terms* of $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$ inductively as follows:

- (i). The constant $\underline{\delta}$ is an ANF term,
- (ii). the constant $\underline{\varepsilon}$ is an ANF term,
- (iii). if $a \in A_\delta$ and s and t are ANF terms, then $\underline{a} \cdot s + t$ is an ANF term,
- (iv). if t is an ANF term, then $\underline{\sigma} \cdot t$ is an ANF term,
- (v). if t is an ANF term, then $\underline{\sigma} \cdot t + \underline{\varepsilon}$ is an ANF term.

☞ Note that by this definition, an ANF term contains a most one $\underline{\varepsilon}$ summand, and at most one summand of the form $\underline{\sigma} \cdot x$.

We will need ANF terms in order to axiomatize the *left merge operator* to be treated in Section 6.3.2. As we will show in Theorem 6.3.1.14, basic terms and ANF terms are equivalent in the sense that any basic term can be rewritten into a ANF term, and vice versa.

Theorem 6.3.1.14 (Equivalence of Basic and ANF Terms)

For every ANF term x of $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$, there exists a basic term x' such that $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = x'$, and, vice versa, for every basic term x of $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$, there exists an ANF term x' such that $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = x'$.

Proof The first part is simple. Suppose that x is an ANF-term. We use induction on the structure of ANF terms, given in Definition 6.3.1.13. Suppose that $x \equiv \underline{\delta}$, then we take $x' \equiv \underline{\delta} \cdot \underline{\varepsilon}$. Now the property trivially holds for $x \equiv \underline{\varepsilon}$, $x \equiv \underline{a} \cdot s + t$, $x \equiv \underline{\sigma} \cdot t$, and $x \equiv \underline{\sigma} \cdot t + \underline{\varepsilon}$, as by the induction hypothesis the property already holds for the ANF terms s and t .

The second part is harder. Suppose that x is a basic term. Then, by Theorem 6.3.1.11 on page 182, we may assume that x is of the following form:

$$x \equiv \sum_{i < m} \underline{a}_i \cdot s_i + \sum_{j < n} \underline{\sigma} \cdot u_j + \sum_{k < p} \underline{\varepsilon}$$

for $m, n, p \in \mathbb{N}$, $m + n + p \geq 1$, $a_i \in A_\delta$, and basic terms s_i and u_j . We now distinguish six mutually exclusive cases:

- (i). There are only $\underline{\varepsilon}$ summands: $m = n = 0$ and $p \geq 1$.
- (ii). There are only $\underline{\sigma} \cdot u_j$ summands: $m = 0$, $n \geq 1$, and $p = 0$.
- (iii). There are both $\underline{\sigma} \cdot u_j$ and $\underline{\varepsilon}$ summands, but no other summands: $m = 0$, $n \geq 1$, and $p \geq 1$.
- (iv). There is exactly one $\underline{a}_i \cdot s_i$ summand, and no other summands: $m = 1$ and $n = p = 0$.
- (v). There is exactly one $\underline{a}_i \cdot s_i$ summand, and at least one other summand: $m = 1$ and $n + p \geq 1$.

(vi). There are at least two $\underline{a}_i \cdot s_i$ summands: $m \geq 2$.

As can be easily seen, this covers all cases. We now prove the property for all six cases, using induction on the number of symbols in x :

(i). We have $m = n = 0$ and $p \geq 1$. Then:

$$\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \sum_{k < p} \underline{\underline{\varepsilon}} = \underline{\underline{\varepsilon}}$$

and $\underline{\underline{\varepsilon}}$ is an ANF term. Note that the sum over k cannot be empty, as $p \geq 1$.

(ii). We have $m = 0$, $n \geq 1$, and $p = 0$. Then:

$$\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \sum_{j < n} \underline{\underline{\sigma}} \cdot u_j = \underline{\underline{\sigma}} \cdot \left(\sum_{j < n} u_j \right)$$

and this last term is an ANF term, since by the induction hypothesis the property already holds for $\sum_{j < n} u_j$. Note that the sum over j cannot be empty, as $n \geq 1$.

(iii). We have $m = 0$, $n \geq 1$, and $p \geq 1$. Then:

$$\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \sum_{j < n} \underline{\underline{\sigma}} \cdot u_j + \sum_{k < p} \underline{\underline{\varepsilon}} = \underline{\underline{\sigma}} \cdot \left(\sum_{j < n} u_j \right) + \underline{\underline{\varepsilon}}$$

and this last term is an ANF term, since by the induction hypothesis the property already holds for $\sum_{j < n} u_j$. Note that neither the sum over j nor the one over k can be empty, as $n \geq 1$ and $p \geq 1$.

(iv). We have $m = 1$ and $n = p = 0$. Then:

$$\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \underline{\underline{a_0}} \cdot s_0 = \underline{\underline{a_0}} \cdot s_0 + \underline{\underline{\delta}}$$

and $\underline{\underline{a_0}} \cdot s_0 + \underline{\underline{\delta}}$ is an ANF term since by the induction hypothesis the property already holds for s_0 .

(v). We have $m = 1$ and $n + p \geq 1$. Then:

$$\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \underline{\underline{a_0}} \cdot s_0 + \left(\sum_{j < n} \underline{\underline{\sigma}} \cdot u_j + \sum_{k < p} \underline{\underline{\varepsilon}} \right)$$

and this last term is an ANF term, since by the induction hypothesis the property already holds for s_0 and the second summand. Note that the second summand cannot be empty, as $n + p \geq 1$.

(vi). We have $m \geq 2$. Then:

$$\begin{aligned} \text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x &= \sum_{i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{\sigma}} \cdot u_j + \sum_{k < p} \underline{\underline{\varepsilon}} = \\ &\underline{\underline{a_0}} \cdot s_0 + \left(\sum_{1 \leq i < m} \underline{\underline{a_i}} \cdot s_i + \sum_{j < n} \underline{\underline{\sigma}} \cdot u_j + \sum_{k < p} \underline{\underline{\varepsilon}} \right) \end{aligned}$$

and this last term is an ANF term, since by the induction hypothesis the property already holds for s_0 and the second summand. Note that the second summand cannot be empty, as $m \geq 2$.

■

Corollary 6.3.1.15 (Structural Induction over ANF)

Any proposition of the form “For all basic terms ...”, that we would normally prove by structural induction over the definition of basic terms, may now also be proven by induction over the definition of ANF terms.

Proof This follows directly from Theorem 6.3.1.14 on page 184. ■

Theorem 6.3.1.16 (Elimination for $BPA_{\text{drt},\varepsilon}^-$ -ID)

Let t be a closed $BPA_{\text{drt},\varepsilon}^-$ -ID term. Then there is a basic term s such that $BPA_{\text{drt},\varepsilon}^- \text{-ID} \vdash s = t$.

Proof Use the lexicographical path ordering method described in Proof Outline 4.2.1.1 on page 68. We give no details. ■

Theorem 6.3.1.17 (Soundness of $BPA_{\text{drt},\varepsilon}^-$ -ID)

The set of closed $BPA_{\text{drt},\varepsilon}^-$ -ID terms modulo bisimulation equivalence is a model of the axioms of $BPA_{\text{drt},\varepsilon}^-$ -ID.

Proof Use the direct method described in Proof Outline 4.2.2.1 on page 69. We give no details. ■

Lemma 6.3.1.18 (Towards Completeness of $BPA_{\text{drt},\varepsilon}^-$ -ID)

Let x be a closed $BPA_{\text{drt},\varepsilon}^-$ -ID term and let $a \in A$. Then we have:

- (i). $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{a} y \implies BPA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \underline{a} \cdot y + x$,
- (ii). $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \downarrow \implies BPA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \underline{\varepsilon} + x$,
- (iii). $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} \implies BPA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \nu(x)$,
- (iv). $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y \implies BPA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = \underline{\sigma} \cdot y + x$,
- (v). $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{a} y \implies n(x) > n(y)$,
- (vi). $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y \implies n(x) > n(y)$.

Proof In the same way as the proof of Lemma 4.3.2.5 on page 82. We give no details. ■

Theorem 6.3.1.19 (Completeness of $BPA_{\text{drt},\varepsilon}^-$ -ID)

The axiom system $BPA_{\text{drt},\varepsilon}^- \text{-ID}$ is a complete axiomatization of the set of closed $BPA_{\text{drt},\varepsilon}^-$ -ID terms modulo bisimulation equivalence.

Proof We use the direct method described in Proof Outline 4.2.3.1 on page 70. Suppose $x + y \sim_{BPA_{\text{drt},\varepsilon}^- \text{-ID}} y$. We then prove that $BPA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x + y = y$.

- (i). $x \equiv \underline{\varepsilon}$. From the deduction rules we have $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models x \downarrow$, and $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models (x + y) \downarrow$. Since $x + y \sim_{BPA_{\text{drt},\varepsilon}^- \text{-ID}} y$, we then also have $T(BPA_{\text{drt},\varepsilon}^- \text{-ID}) \models y \downarrow$. Then, using Lemma 6.3.1.18(ii), we have $BPA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x + y = \underline{\varepsilon} + y = y$.

- (ii). $x \equiv \underline{\underline{\delta}} \cdot t$, where t is a basic term. Then we have $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x + y = \underline{\underline{\delta}} \cdot t + y = \underline{\underline{\delta}} + y = y$.
- (iii). $x \equiv \underline{\underline{a}} \cdot t$, where $a \in A$ and t is a basic term. From the deduction rules we obtain $T(\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{a} t$ and $T(\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}) \models x + y \xrightarrow{a} t$. Since $x + y \sim_{\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}} y$, we then also have $T(\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}) \models y \xrightarrow{a} s$ for some s such that $t \sim_{\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}} s$. By the induction hypothesis we have $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash s = t$. From Lemma 6.3.1.18(i) we have $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \underline{\underline{a}} \cdot s + y$. So, $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x + y = \underline{\underline{a}} \cdot t + y = \underline{\underline{a}} \cdot s + y = y$.
- (iv). $x \equiv s + t$, where s and t are basic terms. Since $x + y \sim_{\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}} y$, we also have $s + y \sim_{\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}} y$ and $t + y \sim_{\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}} y$. By the induction hypothesis we then have $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash s + y = y, t + y = y$. So, $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x + y = s + t + y = s + y = y$.
- (v). $x \equiv \underline{\underline{\sigma}} \cdot t$, where t is a basic term. From the deduction rules we now have that $T(\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} t$ and since $x + y \sim_{\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}} y$ we also have $T(\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}) \models y \xrightarrow{\sigma} s, x + y \xrightarrow{\sigma} t + s$ for some s such that $t + s \sim_{\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}} s$. By Lemma 6.3.1.18(iv) we have $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \underline{\underline{\sigma}} \cdot s + y$. By the induction hypothesis we have $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash t + s = s$. So, $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x + y = \underline{\underline{\sigma}} \cdot t + y = \underline{\underline{\sigma}} \cdot t + \underline{\underline{\sigma}} \cdot s + y = \underline{\underline{\sigma}} \cdot (t + s) + y = \underline{\underline{\sigma}} \cdot s + y = y$.

■

6.3.2 $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$

In this section, we extend the process algebra $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$ to the process algebra $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$ by introducing axioms for the free merge. The axioms will be in the style of $\text{PA}_{\text{drt}}^- \text{-ID}'$, as the ν/σ -style of $\text{PA}_{\text{drt}}^- \text{-ID}$ appears to be unsuitable in a context with the empty process.

Definition 6.3.2.1 (Signature of $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$)

The signature of $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$ consists of the *undelayable actions* $\{\underline{\underline{a}} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\underline{\delta}}$, the *time-unit delay constant* $\underline{\underline{\sigma}}$, the *undelayable empty process constant* $\underline{\underline{\varepsilon}}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *“now” operator* ν_{rel} , the *free merge operator* \parallel , and the *left merge operator* \llcorner .

Definition 6.3.2.2 (Axioms of $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$)

The process algebra $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$ is axiomatized by the axioms of $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$ given in Definition 6.3.1.3 on page 179, and Axioms DRTEM1–DRTEM12 shown in Table 6.3 on the next page: $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} = \text{A1–A5} + \text{DRTE1–DRTE4} + \text{TF} + \text{DCSE1–DCSE4} + \text{DRTEM1–DRTEM12}$.

☞ Axioms DRTEM1–DRTEM3 are identical to Axioms DRTM1, DRTM3, and DRTM4 of $\text{PA}_{\text{drt}}^- \text{-ID}$, so we will not elaborate on them.

Axioms DRTEM4–DRTEM7 ensure that $x \parallel y$ has a summand $\underline{\underline{\varepsilon}}$ iff x and y both have a summand $\underline{\underline{\varepsilon}}$. Note that $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\underline{\varepsilon}} \llcorner x = \underline{\underline{\delta}}$ iff $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \not\vdash x = \underline{\underline{\varepsilon}}$ (we will prove this in Proposition 6.3.2.15 on page 193).

Axioms DRTEM8–DRTEM12 define the interaction between the undelayable empty process and the left merge. They work by first eliminating all summands $\underline{\underline{a}} \cdot y$ on the right-hand side of the left merge (Axiom DRTEM8), and then distinguishing between terms that only have a $\underline{\underline{\sigma}} \cdot y$ summand left (Axiom DRTEM11), only a $\underline{\underline{\varepsilon}}$ summand left (Axiom DRTEM10), both (Axiom DRTEM12), or none (Axiom DRTEM9). Notice that this is exactly the structure of ANF terms given in Definition 6.3.1.13 on page 184.

$x \parallel y = x \parallel y + y \parallel x$	DRTEM1
$\underline{a} \cdot x \parallel y = \underline{a} \cdot (x \parallel y)$	DRTEM2
$(x + y) \parallel z = x \parallel z + y \parallel z$	DRTEM3
$\underline{\underline{\varepsilon}} \parallel \underline{\underline{\varepsilon}} = \underline{\underline{\varepsilon}}$	DRTEM4
$\underline{\underline{\varepsilon}} \parallel \underline{a} \cdot x = \underline{\underline{\delta}}$	DRTEM5
$\underline{\underline{\varepsilon}} \parallel \underline{\underline{\sigma}} \cdot x = \underline{\underline{\delta}}$	DRTEM6
$\underline{\underline{\varepsilon}} \parallel (x + y) = \underline{\underline{\varepsilon}} \parallel x + \underline{\underline{\varepsilon}} \parallel y$	DRTEM7
$\underline{\underline{\sigma}} \cdot x \parallel (\underline{a} \cdot y + z) = \underline{\underline{\sigma}} \cdot x \parallel z$	DRTEM8
$\underline{\underline{\sigma}} \cdot x \parallel \underline{\underline{\delta}} = \underline{\underline{\delta}}$	DRTEM9
$\underline{\underline{\sigma}} \cdot x \parallel \underline{\underline{\varepsilon}} = \underline{\underline{\sigma}} \cdot x$	DRTEM10
$\underline{\underline{\sigma}} \cdot x \parallel \underline{\underline{\sigma}} \cdot y = \underline{\underline{\sigma}} \cdot (x \parallel y)$	DRTEM11
$\underline{\underline{\sigma}} \cdot x \parallel (\underline{\underline{\sigma}} \cdot y + \underline{\underline{\varepsilon}}) = \underline{\underline{\sigma}} \cdot (x \parallel y)$	DRTEM12

Table 6.3: Additional axioms for $\text{PA}_{\text{dr},\varepsilon}^-$ -ID.

Remark 6.3.2.3 (Axioms of $\text{PA}_{\text{dr},\varepsilon}^-$ -ID, Part I)

Note that, unlike in a setting without empty process, we do not need an axiom $\underline{a} \parallel x = \underline{a} \cdot x$, as that equality is derivable for all closed terms x (we will prove this in Proposition 6.3.2.14 on page 192). For example:

$$\begin{aligned}
\underline{\underline{a}} \parallel \underline{\underline{b}} &= \underline{\underline{a}} \cdot \underline{\underline{\varepsilon}} \parallel \underline{\underline{b}} = \underline{\underline{a}} \cdot (\underline{\underline{\varepsilon}} \parallel \underline{\underline{b}}) = \underline{\underline{a}} \cdot (\underline{\underline{\varepsilon}} \parallel \underline{\underline{b}} + \underline{\underline{b}} \parallel \underline{\underline{\varepsilon}}) = \underline{\underline{a}} \cdot (\underline{\underline{\delta}} + \underline{\underline{b}} \cdot \underline{\underline{\varepsilon}} \parallel \underline{\underline{\varepsilon}}) \\
&= \underline{\underline{a}} \cdot \underline{\underline{b}} \cdot (\underline{\underline{\varepsilon}} \parallel \underline{\underline{\varepsilon}}) = \underline{\underline{a}} \cdot \underline{\underline{b}} \cdot (\underline{\underline{\varepsilon}} \parallel \underline{\underline{\varepsilon}} + \underline{\underline{\varepsilon}} \parallel \underline{\underline{\varepsilon}}) = \underline{\underline{a}} \cdot \underline{\underline{b}} \cdot (\underline{\underline{\varepsilon}} \parallel \underline{\underline{\varepsilon}}) = \underline{\underline{a}} \cdot \underline{\underline{b}} \cdot \underline{\underline{\varepsilon}} \\
&= \underline{\underline{a}} \cdot \underline{\underline{b}}
\end{aligned}$$

Remark 6.3.2.4 (Axioms of $\text{PA}_{\text{dr},\varepsilon}^-$ -ID, Part II)

The axiomatization as given in Definition 6.3.2.2 on the page before has some consequences that may not be obvious at first sight. For example, consider what we get if we eliminate the merge operator from the expression $(\underline{a} + \underline{\underline{\varepsilon}}) \parallel \underline{\underline{b}}$ using the axioms:

$$\begin{aligned}
(\underline{a} + \underline{\underline{\varepsilon}}) \parallel \underline{\underline{b}} &= (\underline{a} + \underline{\underline{\varepsilon}}) \parallel \underline{\underline{b}} + \underline{\underline{b}} \parallel (\underline{a} + \underline{\underline{\varepsilon}}) \\
&= \underline{a} \parallel \underline{\underline{b}} + \underline{\underline{\varepsilon}} \parallel \underline{\underline{b}} + \underline{\underline{b}} \cdot (\underline{a} + \underline{\underline{\varepsilon}}) \\
&= \underline{a} \cdot \underline{\underline{b}} + \underline{\underline{\delta}} + \underline{\underline{b}} \cdot (\underline{a} + \underline{\underline{\varepsilon}}) \\
&= \underline{a} \cdot \underline{\underline{b}} + \underline{\underline{b}} \cdot (\underline{a} + \underline{\underline{\varepsilon}})
\end{aligned}$$

So, if we execute a b first, then an a can always still follow. This may seem counter-intuitive; apparently it is not possible for the $\underline{\underline{\varepsilon}}$ to the left of the merge to execute before the b does, in which case a would *not* be enabled after the execution of the b .

The rationale behind this, is that the $\underline{\underline{\varepsilon}}$ is not something that “executes”; it merely represents an *option* to terminate. Hence, the option remains open until the process actually *does terminate* or performs an action. It does not “just get lost”. So for example the process $\underline{a} + \underline{\underline{\varepsilon}}$ cannot drop the $\underline{\underline{\varepsilon}}$ to turn itself into the process \underline{a} . If it wants to lose the $\underline{\underline{\varepsilon}}$ it has to execute an a .

In the context of a (multiple) merge, this means that an $\underline{\underline{\varepsilon}}$ summand in one of the merge components does not manifest itself as long as the other components of the merge still have to execute one or more actions before they can terminate or move on to the following time-slice.

One could be tempted to “repair” this behavior, for example by dropping Axioms DRTEM4–DRTEM7, and adding an axiom like $\underline{\underline{\varepsilon}} \ll x = x$ (i.e., treating $\underline{\underline{\varepsilon}}$ like it was an ordinary action). Although this does give us the “desired” equality $(\underline{\underline{a}} + \underline{\underline{\varepsilon}}) \parallel \underline{\underline{b}} = (\underline{\underline{a}} + \underline{\underline{\varepsilon}}) \cdot \underline{\underline{b}} + \underline{\underline{b}} \cdot (\underline{\underline{a}} + \underline{\underline{\varepsilon}})$, it backfires immediately, as it destroys the associativity of the merge. This can be easily checked by expanding $((\underline{\underline{a}} + \underline{\underline{\varepsilon}}) \parallel \underline{\underline{b}}) \parallel \underline{\underline{c}}$ and $(\underline{\underline{a}} + \underline{\underline{\varepsilon}}) \parallel (\underline{\underline{b}} \parallel \underline{\underline{c}})$; it turns out that the second process has a summand $\underline{\underline{c}} \cdot \underline{\underline{b}}$, while the first process does not. This is of course unacceptable, as we already stated in our design goals. Any attempt at such “repairs” appears to have this consequence, unless one is willing to sacrifice the right-distributivity of the \cdot over the $+$ (Axiom A4), which, again, would violate our design goals.

For other discussions of the above dilemma, see BAETEN AND VAN GLABBEK [31] and VRANCKEN [197].

Remark 6.3.2.5 (Axioms of $\text{PA}_{\text{drt},\varepsilon}^-$ -ID, Part III)

The behavior of $\underline{\underline{\varepsilon}}$ with respect to the merge manifests itself even more unexpectedly when time comes into play. For example, consider the expression $(\underline{\underline{\sigma}} \cdot \underline{\underline{a}} + \underline{\underline{\varepsilon}}) \parallel \underline{\underline{\sigma}} \cdot \underline{\underline{b}}$. Applying the axioms we get:

$$\begin{aligned} (\underline{\underline{\sigma}} \cdot \underline{\underline{a}} + \underline{\underline{\varepsilon}}) \parallel \underline{\underline{\sigma}} \cdot \underline{\underline{b}} &= (\underline{\underline{\sigma}} \cdot \underline{\underline{a}} + \underline{\underline{\varepsilon}}) \ll \underline{\underline{\sigma}} \cdot \underline{\underline{b}} + \underline{\underline{\sigma}} \cdot \underline{\underline{b}} \ll (\underline{\underline{\sigma}} \cdot \underline{\underline{a}} + \underline{\underline{\varepsilon}}) \\ &= \underline{\underline{\sigma}} \cdot \underline{\underline{a}} \ll \underline{\underline{\sigma}} \cdot \underline{\underline{b}} + \underline{\underline{\varepsilon}} \ll \underline{\underline{\sigma}} \cdot \underline{\underline{b}} + \underline{\underline{\sigma}} \cdot \underline{\underline{b}} \ll (\underline{\underline{\sigma}} \cdot \underline{\underline{a}} + \underline{\underline{\varepsilon}}) \\ &= \underline{\underline{\sigma}} \cdot \underline{\underline{a}} \ll \underline{\underline{\sigma}} \cdot \underline{\underline{b}} + \underline{\underline{\delta}} + \underline{\underline{\sigma}} \cdot \underline{\underline{b}} \ll \underline{\underline{\sigma}} \cdot \underline{\underline{a}} \\ &= \underline{\underline{\sigma}} \cdot \underline{\underline{a}} \parallel \underline{\underline{\sigma}} \cdot \underline{\underline{b}} \end{aligned}$$

So, the $\underline{\underline{\varepsilon}}$ -summand on the left side of the merge is immaterial! How should we interpret this?

Again, by viewing $\underline{\underline{\varepsilon}}$ as the option to terminate (in the first time-slice, in this case), we can see that this option can never be exercised. That is to say, because the right component of the merge cannot terminate in the first time-slice, the option to terminate of the left component remains open till we enter the second time-slice. But as we leave the first time-slice, the $\underline{\underline{\varepsilon}}$ disappears, as it only presents an option to terminate *in that time-slice*. Hence, it vanishes into thin air.

As we have argued before, attempting to “repair” this behavior by treating $\underline{\underline{\varepsilon}}$ as an action, means we lose the associativity of the merge. Alternatively, we could replace Axioms DRTEM12 with the following:

$$\underline{\underline{\sigma}} \cdot x \ll (\underline{\underline{\sigma}} \cdot y + \underline{\underline{\varepsilon}}) = \underline{\underline{\sigma}} \cdot x \ll \underline{\underline{\sigma}} \cdot y + \underline{\underline{\sigma}} \cdot x \ll \underline{\underline{\varepsilon}}$$

Then the $\underline{\underline{\varepsilon}}$ does not vanish, but leads to an extra summand $\underline{\underline{\sigma}} \cdot b$ in the above example. This alternative, however, violates the time-determinism principle: $\underline{\underline{\sigma}} \cdot \underline{\underline{a}} \ll (\underline{\underline{\sigma}} \cdot \underline{\underline{b}} + \underline{\underline{\varepsilon}})$ can do a time step to $\underline{\underline{a}} \parallel \underline{\underline{b}}$, while $\underline{\underline{\sigma}} \cdot \underline{\underline{a}} \ll \underline{\underline{\sigma}} \cdot \underline{\underline{b}} + \underline{\underline{\sigma}} \cdot \underline{\underline{b}}$ can do a time step to $\underline{\underline{a}} \parallel \underline{\underline{b}} + \underline{\underline{b}}$. As stated in our design goals, this is not acceptable. Again, we are stuck.

Now look at the following example. Applying our axioms, we have:

$$\begin{aligned}
(\underline{a} + \underline{\varepsilon}) \parallel \underline{\sigma} \cdot \underline{b} &= (\underline{a} + \underline{\varepsilon}) \parallel \underline{\sigma} \cdot \underline{b} + \underline{\sigma} \cdot \underline{b} \parallel (\underline{a} + \underline{\varepsilon}) \\
&= \underline{a} \parallel \underline{\sigma} \cdot \underline{b} + \underline{\varepsilon} \parallel \underline{\sigma} \cdot \underline{b} + \underline{\sigma} \cdot \underline{b} \parallel (\underline{a} + \underline{\varepsilon}) \\
&= \underline{a} \cdot \underline{\sigma} \cdot \underline{b} + \underline{\delta} + \underline{\sigma} \cdot \underline{b} \parallel \underline{\varepsilon} \\
&= \underline{a} \cdot \underline{\sigma} \cdot \underline{b} + \underline{\sigma} \cdot \underline{b} \\
&= (\underline{a} + \underline{\varepsilon}) \cdot \underline{\sigma} \cdot \underline{b}
\end{aligned}$$

Here, the $\underline{\varepsilon}$ -summand on the left side of the merge does materialize. This is due to the fact that $\underline{a} + \underline{\varepsilon}$ cannot move on to the second time-slice, while $\underline{\sigma} \cdot \underline{b}$ can. So, $\underline{a} + \underline{\varepsilon}$ is forced to terminate before $\underline{\sigma} \cdot \underline{b}$ can move on, and the $\underline{\varepsilon}$ does not vanish.

If, for the sake of orthogonality, we would want the $\underline{\varepsilon}$ -summand to disappear in this case also, we could for example replace Axioms DRTEM10 by $\underline{\sigma} \cdot x \parallel \underline{\varepsilon} = \underline{\delta}$. That however, would lead to $\underline{\sigma} \parallel \underline{\varepsilon} = \underline{\delta}$, destroying the unit property of $\underline{\varepsilon}$ with respect to the \parallel . Furthermore, this leads to $\underline{\sigma} \parallel \underline{a} = \underline{a} \cdot \underline{\delta}$, in itself enough reason to dismiss this alternative. If we attempt to save what can be saved, for example by putting $\underline{\varepsilon} \parallel x = x$, we again lose the associativity of the merge.

Concluding: when x can terminate, and y can do a time step, then the termination option of x materializes in $x \parallel y$ if and only if x cannot do a time step. This may seem, and probably is, counter intuitive. However, a simpler way is not conceivable. If the termination option would *always* materialize, we either lose the associativity of the merge or time determinism. If it would *never* materialize, we lose the unit property with respect to the merge and the associativity of the merge. So, both simple ways are too simple, and lead to violations of our design goals.

Remark 6.3.2.6 (Axioms of $PA_{\text{drt},\varepsilon}^-$ -ID, Part IV)

In the case where one of the n components in a multiple merge has *only* the option to terminate (so the component has no enabled actions or time steps), while the other components do not all have that option, the $\underline{\varepsilon}$ again vanishes, collapsing the merge to $n - 1$ components. So for example, $\underline{\varepsilon} \parallel \underline{\sigma} \cdot \underline{a} \parallel \underline{\sigma} \cdot \underline{b} = \underline{\sigma} \cdot \underline{a} \parallel \underline{\sigma} \cdot \underline{b}$. Or, more generally, $\underline{\varepsilon} \parallel x = x \parallel \underline{\varepsilon} = x$ for any closed process term x : with respect to closed terms, $\underline{\varepsilon}$ is also a proper unit element for the merge. We will prove this in Proposition 6.3.2.14 on page 192.

Definition 6.3.2.7 (Semantics of $PA_{\text{drt},\varepsilon}^-$ -ID)

The semantics of $PA_{\text{drt},\varepsilon}^-$ -ID are given by the term-deduction system $T(PA_{\text{drt},\varepsilon}^-$ -ID), induced by the deduction rules for $BPA_{\text{drt},\varepsilon}^-$ -ID given in Definition 6.3.1.4 on page 180, and the additional deduction rules for $PA_{\text{drt},\varepsilon}^-$ -ID shown in Table 6.4 on the facing page.

☞ The deduction rules for the parallel composition with respect to time transitions, shown in Table 6.4 on the next page, suffer from the same complications we saw in the deduction rules for the sequential composition with respect to time transitions. We give two examples to clarify them.

First, as we have $\underline{\varepsilon} \parallel \underline{\sigma} \cdot \underline{a} = \underline{\sigma} \cdot \underline{a}$, we should have $\underline{\varepsilon} \parallel \underline{\sigma} \cdot \underline{a} \xrightarrow{\sigma} \underline{a}$. Likewise, as $(\underline{\sigma} \cdot \underline{a} + \underline{\varepsilon}) \parallel \underline{\sigma} \cdot \underline{b} = \underline{\sigma} \cdot (\underline{a} \parallel \underline{b})$, we should have $(\underline{\sigma} \cdot \underline{a} + \underline{\varepsilon}) \parallel \underline{\sigma} \cdot \underline{b} \xrightarrow{\sigma} \underline{a} \parallel \underline{b}$, and not $(\underline{\sigma} \cdot \underline{a} + \underline{\varepsilon}) \parallel \underline{\sigma} \cdot \underline{b} \xrightarrow{\sigma} \underline{b}$.

Theorem 6.3.2.8 (Time Determinism for $PA_{\text{drt},\varepsilon}^-$ -ID)

Let x , y , and y' be closed $PA_{\text{drt},\varepsilon}^-$ -ID terms. Then we have:

$$T(PA_{\text{drt},\varepsilon}^-$$
-ID) $\models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y' \implies y \equiv y'$

$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y}$	$\frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'}$	$\frac{x \xrightarrow{a} x'}{x \perp\!\!\!\perp y \xrightarrow{a} x' \parallel y}$
$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \parallel y \xrightarrow{\sigma} x' \parallel y'}$	$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \perp\!\!\!\perp y \xrightarrow{\sigma} x' \perp\!\!\!\perp y'}$	
$\frac{x \xrightarrow{\sigma}, x\downarrow, y \xrightarrow{\sigma} y'}{x \parallel y \xrightarrow{\sigma} y'}$	$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma}, y\downarrow}{x \parallel y \xrightarrow{\sigma} x'}$	$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma}, y\downarrow}{x \perp\!\!\!\perp y \xrightarrow{\sigma} x'}$
$\frac{x\downarrow, y\downarrow}{(x \parallel y)\downarrow}$	$\frac{x\downarrow, y\downarrow}{(x \perp\!\!\!\perp y)\downarrow}$	

Table 6.4: Additional deduction rules for $\text{PA}_{\text{drt},\varepsilon}^-$ -ID.

Proof We proceed in the manner outlined in Theorem 6.3.1.5 on page 181. As the only new deduction rules are for the free merge and the left merge, we do not repeat the cases (i)–(vi) already listed in Theorem 6.3.1.5 on page 181.

(vii). $x \equiv s \parallel t$ for closed $\text{PA}_{\text{drt},\varepsilon}^-$ -ID terms s and t . We proceed by case distinction on the transitions of s and t .

- (a) $T(\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', t \xrightarrow{\sigma} t'$ (where, by the induction hypothesis, s' and t' are unique). Then $y \equiv y' \equiv s' \parallel t'$.
- (b) $T(\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma}, s\downarrow, t \xrightarrow{\sigma} t'$ (where t' is unique). Then $y \equiv y' \equiv t'$.
- (c) $T(\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', t \xrightarrow{\sigma}, t\downarrow$ (where s' is unique). Then $y \equiv y' \equiv s'$.
- (d) Otherwise. In contradiction with $T(\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y$.

(viii). $x \equiv s \perp\!\!\!\perp t$ for closed $\text{PA}_{\text{drt},\varepsilon}^-$ -ID terms s and t . We proceed by case distinction on the transitions of s and t .

- (a) $T(\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', t \xrightarrow{\sigma} t'$ (where s' and t' are unique). Then $y \equiv y' \equiv s' \perp\!\!\!\perp t'$.
- (b) $T(\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', t \xrightarrow{\sigma}, t\downarrow$ (where s' is unique). Then $y \equiv y' \equiv s'$.
- (c) Otherwise. In contradiction with $T(\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y$.

Having inspected all possible cases, we may now conclude that $y \equiv y'$. ■

Definition 6.3.2.9 (Bisimulation and Bisimulation Model for $\text{PA}_{\text{drt},\varepsilon}^-$ -ID)

Bisimulation for $\text{PA}_{\text{drt},\varepsilon}^-$ -ID and the corresponding bisimulation model are defined in the same way as for $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID. Replace “ $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID” by “ $\text{PA}_{\text{drt},\varepsilon}^-$ -ID” in Definition 6.3.1.6 on page 182 and Definition 6.3.1.7 on page 182.

Definition 6.3.2.10 (Basic Terms of $PA_{\text{drt},\varepsilon}^-$ -ID)

When we speak of basic terms in the context of $PA_{\text{drt},\varepsilon}^-$ -ID, we mean $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic terms as defined in Definition 6.3.1.8 on page 182.

Theorem 6.3.2.11 (Elimination for $PA_{\text{drt},\varepsilon}^-$ -ID)

Let t be a closed $PA_{\text{drt},\varepsilon}^-$ -ID term. Then there is a basic term s such that $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash s = t$.

Proof Use the direct method described in Proof Outline 4.2.1.2 on page 68. We give no details. ■

Theorem 6.3.2.12 (Soundness of $PA_{\text{drt},\varepsilon}^-$ -ID)

The set of closed $PA_{\text{drt},\varepsilon}^-$ -ID terms modulo bisimulation equivalence is a model of the axioms of $PA_{\text{drt},\varepsilon}^-$ -ID.

Proof Use the direct method described in Proof Outline 4.2.2.1 on page 69. We give no details. ■

Theorem 6.3.2.13 (Completeness of $PA_{\text{drt},\varepsilon}^-$ -ID)

The axiom system $PA_{\text{drt},\varepsilon}^-$ -ID is a complete axiomatization of the set of closed $PA_{\text{drt},\varepsilon}^-$ -ID terms modulo bisimulation equivalence.

Proof Use Verhoef's method described in Proof Outline 4.2.3.4 on page 71. We give no details. ■

Proposition 6.3.2.14 (Properties of $PA_{\text{drt},\varepsilon}^-$ -ID, Part I)

Let x be a basic term and $a \in A_\delta$. Then the following properties hold:

- (i). $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash x \parallel \underline{\varepsilon} = x$
- (ii). $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash \underline{\varepsilon} \parallel x = x$
- (iii). $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash \underline{a} \ll x = \underline{a} \cdot x$
- (iv). $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash \underline{\delta} \ll x = \underline{\delta}$
- (v). $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash x \ll \underline{\varepsilon} = x$

Proof

(i). We prove this by induction on the structure of basic terms (Definition 6.3.2.10).

- (a) $x \equiv \underline{\varepsilon}$. Then $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash x \parallel \underline{\varepsilon} = \underline{\varepsilon} \parallel \underline{\varepsilon} = \underline{\varepsilon} \ll \underline{\varepsilon} + \underline{\varepsilon} \ll \underline{\varepsilon} = \underline{\varepsilon} + \underline{\varepsilon} = \underline{\varepsilon} = x$.
- (b) $x \equiv \underline{a} \cdot t$ for some $a \in A_\delta$ and basic term t . Then $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash x \parallel \underline{\varepsilon} = \underline{a} \cdot t \parallel \underline{\varepsilon} = \underline{a} \cdot t \ll \underline{\varepsilon} + \underline{\varepsilon} \ll \underline{a} \cdot t = \underline{a} \cdot (t \parallel \underline{\varepsilon}) + \underline{\delta} = \underline{a} \cdot t + \underline{\delta} = \underline{a} \cdot t = x$.
- (c) $x \equiv s + t$ for some basic terms s and t . Then $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash x \parallel \underline{\varepsilon} = (s + t) \parallel \underline{\varepsilon} = (s + t) \ll \underline{\varepsilon} + \underline{\varepsilon} \ll (s + t) = s \ll \underline{\varepsilon} + t \ll \underline{\varepsilon} + \underline{\varepsilon} \ll s + \underline{\varepsilon} \ll t = s \ll \underline{\varepsilon} + \underline{\varepsilon} \ll s + t \ll \underline{\varepsilon} + \underline{\varepsilon} \ll t = s \parallel \underline{\varepsilon} + t \parallel \underline{\varepsilon} = s + t = x$.
- (d) $x \equiv \underline{\sigma} \cdot t$ for some basic term t . Then $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash x \parallel \underline{\varepsilon} = \underline{\sigma} \cdot t \parallel \underline{\varepsilon} = \underline{\sigma} \cdot t \ll \underline{\varepsilon} + \underline{\varepsilon} \ll \underline{\sigma} \cdot t = \underline{\sigma} \cdot t + \underline{\delta} = \underline{\sigma} \cdot t = x$.

(ii). Using (i), we have $PA_{\text{drt},\varepsilon}^-$ -ID $\vdash \underline{\varepsilon} \parallel x = \underline{\varepsilon} \ll x + x \ll \underline{\varepsilon} = x \ll \underline{\varepsilon} + \underline{\varepsilon} \ll x = x \parallel \underline{\varepsilon} = x$.

- (iii). Using (ii), we have $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{a} \ll x = \underline{a} \cdot \underline{\varepsilon} \ll x = \underline{a} \cdot (\underline{\varepsilon} \ll x) = \underline{a} \cdot x$.
- (iv). Using (iii), we have $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\delta} \ll x = \underline{\delta} \cdot x = \underline{\delta}$.
- (v). We again prove this by induction on the structure of basic terms.
- (a) $x \equiv \underline{\varepsilon}$. Then $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x \ll \underline{\varepsilon} = \underline{\varepsilon} \ll \underline{\varepsilon} = \underline{\varepsilon} = x$.
- (b) $x \equiv \underline{a} \cdot t$ for some $a \in A_\delta$ and basic term t . Then $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x \ll \underline{\varepsilon} = \underline{a} \cdot t \ll \underline{\varepsilon} = \underline{a} \cdot (t \ll \underline{\varepsilon}) = \underline{a} \cdot t = x$.
- (c) $x \equiv s + t$ for some basic terms s and t . Then $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash (s + t) \ll \underline{\varepsilon} = s \ll \underline{\varepsilon} + t \ll \underline{\varepsilon} = s + t = x$.
- (d) $x \equiv \underline{\sigma} \cdot t$ for some basic term t . Then $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash x \ll \underline{\varepsilon} = \underline{\sigma} \cdot t \ll \underline{\varepsilon} = \underline{\sigma} \cdot t = x$.

■

Proposition 6.3.2.15 (Properties of $PA_{\text{drt},\varepsilon}^- \text{-ID}$, Part II)

Let x, y, z be closed $PA_{\text{drt},\varepsilon}^- \text{-ID}$ terms. Then the following properties hold:

- (i). $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\varepsilon} \ll x = \begin{cases} \underline{\varepsilon} & \text{if } x \downarrow \\ \underline{\delta} & \text{if } x \uparrow \end{cases}$
- (ii). $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll \nu_{\text{rel}}(y) = \begin{cases} \underline{\sigma} \cdot x & \text{if } y \downarrow \\ \underline{\delta} & \text{if } y \uparrow \end{cases}$
- (iii). $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll (\nu_{\text{rel}}(y) + \underline{\sigma} \cdot z) = \underline{\sigma}(x \ll z)$
- (iv). $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \nu_{\text{rel}}(x) \ll y = \nu_{\text{rel}}(\nu_{\text{rel}}(x) \ll y)$.
- (v). $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \nu_{\text{rel}}(x) \ll \nu_{\text{rel}}(y) = \nu_{\text{rel}}(\nu_{\text{rel}}(x) \ll \nu_{\text{rel}}(y))$.

Proof By Theorem 6.3.2.11 on the preceding page, we may assume that x, y and z are basic terms.

- (i). We use induction on the structure of basic term x .
- (a) $x \equiv \underline{\varepsilon}$. Then $x \downarrow$, and we have $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\varepsilon} \ll x = \underline{\varepsilon} \ll \underline{\varepsilon} = \underline{\varepsilon}$.
- (b) $x \equiv \underline{a} \cdot t$ for some $a \in A_\delta$ and basic term t . Then $x \uparrow$, and we have $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\varepsilon} \ll x = \underline{\varepsilon} \ll \underline{a} \cdot t = \underline{\delta}$.
- (c) $x \equiv s + t$ for some basic terms s and t . First, suppose that $x \uparrow$. Then, $s \uparrow$ and $t \uparrow$. So, by the induction hypothesis, $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\varepsilon} \ll x = \underline{\varepsilon} \ll (s + t) = \underline{\varepsilon} \ll s + \underline{\varepsilon} \ll t = \underline{\delta} + \underline{\delta} = \underline{\delta}$. Secondly, suppose that $x \downarrow$. Then, $s \downarrow$, or $t \downarrow$, or both. Assume that $s \downarrow$ and $t \uparrow$. Then, by the induction hypothesis $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\varepsilon} \ll x = \underline{\varepsilon} \ll (s + t) = \underline{\varepsilon} \ll s + \underline{\varepsilon} \ll t = \underline{\varepsilon} + \underline{\delta} = \underline{\varepsilon}$. The cases $s \uparrow, t \downarrow$ and $s \downarrow, t \downarrow$ are handled in the same way.
- (d) $x \equiv \underline{\sigma} \cdot t$ for some basic term t . Then $x \uparrow$, and we have $PA_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\varepsilon} \ll x = \underline{\varepsilon} \ll \underline{\sigma} \cdot t = \underline{\delta}$.
- (ii). As y is a basic term, by Corollary 6.3.1.15, we may use induction on the structure of ANF terms

- (a) $y \equiv \underline{\delta}$. Then $y \dagger$, and we have $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll v(y) = \underline{\sigma} \cdot x \ll v(\underline{\delta}) = \underline{\sigma} \cdot x \ll v(\underline{\delta} \cdot \underline{\varepsilon}) = \underline{\sigma} \cdot x \ll \underline{\delta} \cdot \underline{\varepsilon} = \underline{\sigma} \cdot x \ll \underline{\delta} = \underline{\delta}$.
- (b) $y \equiv \underline{\varepsilon}$. Then $y \downarrow$, and we have $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll v(y) = \underline{\sigma} \cdot x \ll v(\underline{\varepsilon}) = \underline{\sigma} \cdot x \ll \underline{\varepsilon} = \underline{\sigma} \cdot x$.
- (c) $y \equiv \underline{a} \cdot s + t$ for some $a \in A_\delta$ and basic terms s and t . Then $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll v(y) = \underline{\sigma} \cdot x \ll v(\underline{a} \cdot s + t) = \underline{\sigma} \cdot x \ll (v(\underline{a} \cdot s) + v(t)) = \underline{\sigma} \cdot x \ll (\underline{a} \cdot s + v(t)) = \underline{\sigma} \cdot x \ll v(t)$. Now, using the induction hypothesis, we get $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll v(t) = \underline{\sigma} \cdot x$ when $t \dagger$, hence $y \dagger$, and $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll n(t) = \underline{\delta}$ when $t \downarrow$, hence $y \dagger$.
- (d) $y \equiv \underline{\sigma} \cdot t$ for some basic term t . Then $y \dagger$, and we have $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll v(y) = \underline{\sigma} \cdot x \ll v(\underline{\sigma} \cdot t) = \underline{\sigma} \cdot x \ll \underline{\delta} = \underline{\delta}$.
- (e) $y \equiv \underline{\sigma} \cdot t + \underline{\varepsilon}$ for some basic term t . Then $y \downarrow$, and we have $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll v(y) = \underline{\sigma} \cdot x \ll v(\underline{\sigma} \cdot t + \underline{\varepsilon}) = \underline{\sigma} \cdot x \ll (v(\underline{\sigma} \cdot t) + v(\underline{\varepsilon})) = \underline{\sigma} \cdot x \ll (\underline{\delta} + \underline{\varepsilon}) = \underline{\sigma} \cdot x \ll \underline{\varepsilon} = \underline{\sigma} \cdot x$.

(iii). As y is a basic term, by Corollary 6.3.1.15, we may use induction on the structure of ANF terms.

- (a) $y \equiv \underline{\delta}$. Then $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll (v(y) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(\underline{\delta}) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (\underline{\delta} + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll \underline{\sigma} \cdot z = \underline{\sigma} \cdot (x \ll z)$.
- (b) $y \equiv \underline{\varepsilon}$. Then $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll (v(y) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(\underline{\varepsilon}) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (\underline{\varepsilon} + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot (x \ll z)$.
- (c) $y \equiv \underline{a} \cdot s + t$ for some $a \in A_\delta$ and basic terms s and t . Then $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll (v(y) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(\underline{a} \cdot s + t) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(\underline{a} \cdot s) + v(t) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (\underline{a} \cdot s + v(t) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(t) + \underline{\sigma} \cdot z)$. Now, using the induction hypothesis, $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll (v(t) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot (x \ll z)$.
- (d) $y \equiv \underline{\sigma} \cdot t$ for some basic term t . Then $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll (v(y) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(\underline{\sigma} \cdot t) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (\underline{\delta} + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll \underline{\sigma} \cdot z = \underline{\sigma} \cdot (x \ll z)$.
- (e) $y \equiv \underline{\sigma} \cdot t + \underline{\varepsilon}$ for some basic term t . Then $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \underline{\sigma} \cdot x \ll (v(y) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(\underline{\sigma} \cdot t + \underline{\varepsilon}) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (v(\underline{\sigma} \cdot t) + v(\underline{\varepsilon}) + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (\underline{\delta} + \underline{\varepsilon} + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot x \ll (\underline{\varepsilon} + \underline{\sigma} \cdot z) = \underline{\sigma} \cdot (x \ll z)$.

(iv). We use induction on the structure of basic term x .

- (a) $x \equiv \underline{\varepsilon}$. Suppose that $y \downarrow$. Using (i), we then have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash v(x) \ll y &= v(\underline{\varepsilon}) \ll y = \underline{\varepsilon} \ll y = \underline{\varepsilon} = v(\underline{\varepsilon}) = v(\underline{\varepsilon} \ll y) = \\ &v(v(\underline{\varepsilon}) \ll y) = v(v(x) \ll y) \end{aligned}$$

The case where $y \dagger$ is handled in the same way.

- (b) $x \equiv \underline{a} \cdot t$ for some $a \in A_\delta$ and basic term t . We then have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash v(x) \ll y &= v(\underline{a} \cdot t) \ll y = \underline{a} \cdot t \ll y = \underline{a} \cdot (t \ll y) = \\ &v(\underline{a} \cdot (t \ll y)) = v(\underline{a} \cdot t \ll y) = v(v(\underline{a} \cdot t) \ll y) = \\ &v(v(x) \ll y) \end{aligned}$$

(c) $x \equiv s + t$ for some basic terms s and t . Using the induction hypothesis, we then have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \nu(x) \parallel y &= \nu(s + t) \parallel y = (\nu(s) + \nu(t)) \parallel y = \\ &= \nu(s) \parallel y + \nu(t) \parallel y = \nu(\nu(s) \parallel y) + \nu(\nu(t) \parallel y) = \\ &= \nu(\nu(s) \parallel y + \nu(t) \parallel y) = \nu((\nu(s) + \nu(t)) \parallel y) = \\ &= \nu(\nu(s + t) \parallel y) = \nu(\nu(x) \parallel y) \end{aligned}$$

(d) $x \equiv \underline{\underline{\sigma}} \cdot t$ for some basic term t . Using Proposition 6.3.2.14(iv), we then have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \nu(x) \parallel y &= \nu(\underline{\underline{\sigma}} \cdot t) \parallel y = \underline{\underline{\delta}} \parallel y = \underline{\underline{\delta}} \cdot \underline{\underline{\varepsilon}} = \nu(\underline{\underline{\delta}} \cdot \underline{\underline{\varepsilon}}) = \\ &= \nu(\underline{\underline{\delta}}) = \nu(\underline{\underline{\delta}} \parallel y) = \nu(\nu(\underline{\underline{\sigma}} \cdot t) \parallel y) = \nu(\nu(x) \parallel y) \end{aligned}$$

(v). Using (iv), we have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \nu(x) \parallel \nu(y) &= \nu(x) \parallel \nu(y) + \nu(y) \parallel \nu(x) = \\ &= \nu(\nu(x) \parallel \nu(y)) + \nu(\nu(y) \parallel \nu(x)) = \\ &= \nu(\nu(x) \parallel \nu(y) + \nu(y) \parallel \nu(x)) = \nu(\nu(x) \parallel \nu(y)) \end{aligned}$$

■

☞ These properties, especially (ii), make clear why the ν/σ -axiomatization style we used successfully for $\text{PA}_{\text{drt}}^- \text{-ID}$ does not work well for $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$: we would need a conditional axiom to define $\underline{\underline{\sigma}} \cdot x \parallel \nu_{\text{rel}}(y)$, where the condition would be $y \downarrow$. Although this is not, as it may seem, a model dependent condition (it can be rephrased as $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x = x + \underline{\underline{\varepsilon}}$), it is still hairy enough for us to reject it.

Theorem 6.3.2.16 (Axioms of Standard Concurrency for $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$)

Let x, y, z be closed $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID}$ terms. Then the following properties hold:

- (i). $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x \parallel y = y \parallel x$
- (ii). $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$
- (iii). $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$

Proof By Theorem 6.3.2.11 on page 192, we may assume that x, y and z are basic terms.

- (i). This follows directly from the axioms: $\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash x \parallel y = x \parallel y + y \parallel x = y \parallel x + x \parallel y = y \parallel x$ (note that this holds for open terms also).
- (ii). We use induction on the structure of basic term x , combined with simultaneous induction with (iii):

(a) $x \equiv \underline{\underline{\varepsilon}}$. Using Proposition 6.3.2.15(i), we then have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z &= (\underline{\underline{\varepsilon}} \parallel y) \parallel z = \begin{cases} \underline{\underline{\varepsilon}} \parallel z & \text{if } y \downarrow \\ \underline{\underline{\delta}} \parallel z & \text{if } y \uparrow \end{cases} = \begin{cases} \underline{\underline{\varepsilon}} & \text{if } y \downarrow, z \downarrow \\ \underline{\underline{\delta}} & \text{otherwise} \end{cases} = \\ &= \begin{cases} \underline{\underline{\varepsilon}} & \text{if } (y \parallel z) \downarrow \\ \underline{\underline{\delta}} & \text{if } (y \parallel z) \uparrow \end{cases} = \underline{\underline{\varepsilon}} \parallel (y \parallel z) = x \parallel (y \parallel z) \end{aligned}$$

- (b) $x \equiv \underline{a} \cdot t$ for some $a \in A_\delta$ and basic term t . Using simultaneous induction with (iii), and using Proposition 6.3.2.14(iii) we then have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z &= (\underline{a} \cdot t \parallel y) \parallel z = \underline{a} \cdot (t \parallel y) \parallel z = \\ &= \underline{a} \cdot ((t \parallel y) \parallel z) = \underline{a} \cdot (t \parallel (y \parallel z)) = \underline{a} \cdot t \parallel (y \parallel z) = x \parallel (y \parallel z) \end{aligned}$$

- (c) $x \equiv s + t$ for some basic terms s and t . Using the induction hypothesis, we then have:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z &= ((s + t) \parallel y) \parallel z = (s \parallel y + t \parallel y) \parallel z = \\ &= (s \parallel y) \parallel z + (t \parallel y) \parallel z = s \parallel (y \parallel z) + t \parallel (y \parallel z) = \\ &= (s + t) \parallel (y \parallel z) = x \parallel (y \parallel z) \end{aligned}$$

- (d) $x \equiv \underline{\sigma} \cdot t$ for some basic term t . We assume, by Lemma 6.3.1.12 on page 183, that either $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \nu(y)$, or that there exists a basic term y' , such that $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \nu(y) + \underline{\sigma} \cdot y'$ and $n(y') < n(y)$, and assume mutatis mutandis the same for z . We now distinguish four cases:

1. $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \nu(y)$ and $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash z = \nu(z)$. Then we have, using Proposition 6.3.2.15:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z &= (\underline{\sigma} \cdot t \parallel \nu(y)) \parallel \nu(z) = \\ &= \begin{cases} \underline{\sigma} \cdot t \parallel \nu(z) & \text{if } y \downarrow \\ \underline{\delta} \parallel \nu(z) & \text{if } y \dagger \end{cases} = \begin{cases} \underline{\sigma} \cdot t \parallel \nu(z) & \text{if } y \downarrow \\ \underline{\delta} & \text{if } y \dagger \end{cases} = \\ &= \begin{cases} \underline{\sigma} \cdot t & \text{if } y \downarrow, z \downarrow \\ \underline{\delta} & \text{otherwise} \end{cases} = \begin{cases} \underline{\sigma} \cdot t & \text{if } (y \parallel z) \downarrow \\ \underline{\delta} & \text{otherwise} \end{cases} = \\ &= \underline{\sigma} \cdot t \parallel \nu(y \parallel z) = \underline{\sigma} \cdot t \parallel \nu(\nu(y) \parallel \nu(z)) = \\ &= \underline{\sigma} \cdot t \parallel (\nu(y) \parallel \nu(z)) = x \parallel (y \parallel z) \end{aligned}$$

2. $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \nu(y)$ and $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash z = \nu(z) + \underline{\sigma} \cdot z'$. Then we have, using Proposition 6.3.2.15:

$$\begin{aligned} \text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash \\ (x \parallel y) \parallel z &= (\underline{\sigma} \cdot t \parallel \nu(y)) \parallel (\nu(z) + \underline{\sigma} \cdot z') = \\ &= \begin{cases} \underline{\sigma} \cdot t \parallel (\nu(z) + \underline{\sigma} \cdot z') & \text{if } y \downarrow \\ \underline{\delta} \parallel (\nu(z) + \underline{\sigma} \cdot z') & \text{if } y \dagger \end{cases} = \begin{cases} \underline{\sigma} \cdot (t \parallel z') & \text{if } y \downarrow \\ \underline{\delta} & \text{if } y \dagger \end{cases} = \\ &= \begin{cases} \underline{\sigma} \cdot t \parallel (\nu(s) + \underline{\sigma} \cdot z') & \text{if } y \downarrow \\ \underline{\sigma} \cdot t \parallel \nu(s) & \text{if } y \dagger \end{cases} = \begin{cases} \underline{\sigma} \cdot t \parallel (\nu(s) + \underline{\sigma} \cdot z') & \text{if } y \downarrow \\ \underline{\sigma} \cdot t \parallel (\nu(s) + \underline{\delta}) & \text{if } y \dagger \end{cases} = \\ &= \underline{\sigma} \cdot t \parallel (\nu(s) + \underline{\sigma} \cdot z' \parallel \nu(y)) = \\ &= \underline{\sigma} \cdot t \parallel (\nu(\nu(y) \parallel z + \nu(z) \parallel y) + \underline{\sigma} \cdot z' \parallel \nu(y)) = \\ &= \underline{\sigma} \cdot t \parallel (\nu(\nu(y) \parallel z) + \nu(\nu(z) \parallel y) + \underline{\sigma} \cdot z' \parallel \nu(y)) = \\ &= \underline{\sigma} \cdot t \parallel (\nu(y) \parallel z + \nu(z) \parallel y + \underline{\sigma} \cdot z' \parallel \nu(y)) = \\ &= \underline{\sigma} \cdot t \parallel (\nu(y) \parallel (\nu(z) + \underline{\sigma} \cdot z') + (\nu(z) + \underline{\sigma} \cdot z') \parallel \nu(y)) = \\ &= \underline{\sigma} \cdot t \parallel (\nu(y) \parallel (\nu(z) + \underline{\sigma} \cdot z')) = \\ &= x \parallel (y \parallel z) \end{aligned}$$

where we define:

$$s \equiv \nu(y) \parallel z + \nu(z) \parallel y$$

and use that if $y \dagger$, then $s \dagger$.

3. $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \nu(y) + \underline{\sigma} \cdot y'$ and $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash z = \nu(z) + \underline{\sigma} \cdot z'$. This case is handled in the same way as the previous case.
4. $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash y = \nu(y) + \underline{\sigma} \cdot y'$ and $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash z = \nu(z) + \underline{\sigma} \cdot z'$. Then we have, using Proposition 6.3.2.15, and simultaneous induction with (iii):

$$\begin{aligned}
\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z &= \\
(\underline{\sigma} \cdot t \parallel (\nu(y) + \underline{\sigma} \cdot y')) \parallel (\nu(z) + \underline{\sigma} \cdot z') &= \\
\underline{\sigma} \cdot (t \parallel y') \parallel (\nu(z) + \underline{\sigma} \cdot z') &= \\
\underline{\sigma} \cdot ((t \parallel y') \parallel z') &= \\
\underline{\sigma} \cdot (t \parallel (y' \parallel z')) &= \\
\underline{\sigma} \cdot t \parallel (\nu(\nu(y) \parallel z + \nu(z) \parallel y) + \underline{\sigma} \cdot (y' \parallel z')) &= \\
\underline{\sigma} \cdot t \parallel (\nu(\nu(y) \parallel z + \nu(z) \parallel y) + \underline{\sigma} \cdot (y' \parallel z' + z' \parallel y')) &= \\
\underline{\sigma} \cdot t \parallel \left(\begin{array}{c} \nu(\nu(y) \parallel z) + \underline{\sigma} \cdot (y' \parallel z') \\ \nu(\nu(z) \parallel y) + \underline{\sigma} \cdot (z' \parallel y') \end{array} \right) &= \\
\underline{\sigma} \cdot t \parallel \left(\begin{array}{c} \nu(y) \parallel z + \underline{\sigma} \cdot y' \parallel (\nu(z) + \underline{\sigma} \cdot z') \\ \nu(z) \parallel y + \underline{\sigma} \cdot z' \parallel (\nu(y) + \underline{\sigma} \cdot y') \end{array} \right) &= \\
\underline{\sigma} \cdot t \parallel \left(\begin{array}{c} (\nu(y) + \underline{\sigma} \cdot y') \parallel (\nu(z) + \underline{\sigma} \cdot z') \\ (\nu(z) + \underline{\sigma} \cdot z') \parallel (\nu(y) + \underline{\sigma} \cdot y') \end{array} \right) &= \\
\underline{\sigma} \cdot t \parallel ((\nu(y) + \underline{\sigma} \cdot y') \parallel (\nu(z) + \underline{\sigma} \cdot z')) &= \\
x \parallel (y \parallel z) &
\end{aligned}$$

(iii). We use simultaneous induction with (ii). Using (i), we then have:

$$\begin{aligned}
\text{PA}_{\text{drt},\varepsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z &= \\
(x \parallel y) \parallel z + z \parallel (x \parallel y) &= \\
(x \parallel y + y \parallel x) \parallel z + z \parallel (x \parallel y) &= \\
(x \parallel y) \parallel z + (y \parallel x) \parallel z + z \parallel (y \parallel x) &= \\
x \parallel (y \parallel z) + y \parallel (x \parallel z) + (z \parallel y) \parallel x &= \\
x \parallel (y \parallel z) + y \parallel (z \parallel x) + (z \parallel y) \parallel x &= \\
x \parallel (y \parallel z) + (y \parallel z) \parallel x + (z \parallel y) \parallel x &= \\
x \parallel (y \parallel z) + (y \parallel z + z \parallel y) \parallel x &= \\
x \parallel (y \parallel z) + (y \parallel z) \parallel x &= \\
x \parallel (y \parallel z) &
\end{aligned}$$

■

Corollary 6.3.2.17 (Commutativity and Associativity of the Merge)

For closed terms, the free merge operator of $PA_{\text{drt},\varepsilon}^-$ -ID is commutative and associative.

Proof This follows directly from Theorem 6.3.2.16 on page 195. ■

6.3.3 $ACP_{\text{drt},\varepsilon}^-$ -ID

In this section, we modify the process algebra $PA_{\text{drt},\varepsilon}^-$ -ID by extending the free merge to a merge, where the axioms for the communication merge are based on the inductive definition of basic terms. The resulting process algebra is called $ACP_{\text{drt},\varepsilon}^-$ -ID.

Definition 6.3.3.1 (Signature of $ACP_{\text{drt},\varepsilon}^-$ -ID)

The signature of $ACP_{\text{drt},\varepsilon}^-$ -ID consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *time-unit delay constant* $\underline{\sigma}$, the *undelayable empty process constant* $\underline{\varepsilon}$, the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *“now” operator* ν_{rel} , *merge operator* \parallel , the *left merge operator* \llcorner , the *communication merge operator* \lrcorner , and the *encapsulation operator* ∂_H .

Definition 6.3.3.2 (Axioms of $ACP_{\text{drt},\varepsilon}^-$ -ID)

The process algebra $ACP_{\text{drt},\varepsilon}^-$ -ID is axiomatized by the axioms of $PA_{\text{drt},\varepsilon}^-$ -ID given in Definition 6.3.2.2 on page 187, *minus* Axiom DRTEM1, *plus* Axioms DRTECM1-DRTECM9, DRTECF, and DRTED1-DRTED5 shown in Table 6.5 on the facing page: $ACP_{\text{drt},\varepsilon}^-$ -ID = A1-A5 + DRTE1-DRTE4 + TF + DCSE1-DCSE4 + DRTEM2-DRTEM12 + DRTECM1-DRTECM9 + DRTECF + DRTED1-DRTED6.

☞ Axioms DRTECM1, DRTECM2, DRTECM8, DRTECM9, DRTECF, and DRTED1-DRTED4 all correspond to identical axioms in ACP_{drt}^- -ID, so we will not elaborate on them.

Axiom DRTECM3 expresses that the time-unit delay constant distributes over the communication merge (due to the fact that time steps do not communicate and weak time factorization). Axioms DRTECM4 and DRTECM5 express that if one side of a communication merge can only do a time step, and the other side cannot do a time step, then the communication merge collapses to deadlock.

Axioms DRTECM6 and DRTECM7 express that the undelayable empty process does not communicate: if either side of a communication merge consist of just an undelayable empty process, the whole communication merge again collapses to deadlock.

Finally, Axioms DRTED5 and DRTED6 express that the time-unit delay constant and the undelayable empty process are immune to encapsulation.

Definition 6.3.3.3 (Semantics of $ACP_{\text{drt},\varepsilon}^-$ -ID)

The semantics of $ACP_{\text{drt},\varepsilon}^-$ -ID are given by the term-deduction system $T(ACP_{\text{drt},\varepsilon}^-$ -ID), induced by the deduction rules for $PA_{\text{drt},\varepsilon}^-$ -ID given in Definition 6.3.2.7 on page 190, and the additional deduction rules for $ACP_{\text{drt},\varepsilon}^-$ -ID shown in Table 6.6 on page 200.

☞ These deduction rules are entirely straightforward, so we will not discuss them.

Theorem 6.3.3.4 (Time Determinism for $ACP_{\text{drt},\varepsilon}^-$ -ID)

Let x , y , and y' be closed $ACP_{\text{drt},\varepsilon}^-$ -ID terms. Then we have:

$$T(ACP_{\text{drt},\varepsilon}^-$$
-ID) $\models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y' \implies y \equiv y'$

$x \parallel y = x \parallel y + y \parallel x + x \mid y$	DRTECM1
$\underline{a} \cdot x \mid \underline{b} \cdot y = (\underline{a} \mid \underline{b}) \cdot (x \parallel y)$	DRTECM2
$\underline{\sigma} \cdot x \mid \underline{\sigma} \cdot y = \underline{\sigma} \cdot (x \mid y)$	DRTECM3
$\underline{\sigma} \cdot x \mid \underline{a} \cdot y = \underline{\delta}$	DRTECM4
$\underline{a} \cdot x \mid \underline{\sigma} \cdot y = \underline{\delta}$	DRTECM5
$x \mid \underline{\varepsilon} = \underline{\delta}$	DRTECM6
$\underline{\varepsilon} \mid x = \underline{\delta}$	DRTECM7
$(x + y) \mid z = x \mid z + y \mid z$	DRTECM8
$x \mid (y + z) = x \mid y + x \mid z$	DRTECM9
$\underline{a} \mid \underline{b} = \underline{c} \quad \text{if } \gamma(a, b) = c$	DRTECF
$\partial_H(\underline{a}) = \underline{a} \quad \text{if } a \notin H$	DRTED1
$\partial_H(\underline{a}) = \underline{\delta} \quad \text{if } a \in H$	DRTED2
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	DRTED3
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	DRTED4
$\partial_H(\underline{\sigma}) = \underline{\sigma}$	DRTED5
$\partial_H(\underline{\varepsilon}) = \underline{\varepsilon}$	DRTED6

Table 6.5: Additional axioms for $\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}$.

Proof We proceed in the manner outlined in Theorem 6.3.1.5 on page 181. As the only new deduction rules regarding σ -transitions are for the communication merge and the encapsulation, we do not repeat the cases (i)–(viii) already listed in Theorem 6.3.1.5 on page 181 and Theorem 6.3.2.8 on page 190.

- (ix). $x \equiv s \mid t$ for closed $\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}$ terms s and t . We proceed by case distinction on the transitions of s and t .
- (a) $T(\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s', t \xrightarrow{\sigma} t'$ (where, by the induction hypothesis, s' and t' are unique). Then $y \equiv y' \equiv s' \mid t'$.
 - (b) Otherwise. In contradiction with $T(\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y$.
- (x). $x \equiv \partial_H(s)$ for a closed $\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}$ terms s . We proceed by case distinction on the transitions of s .
- (a) $T(\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}) \models s \xrightarrow{\sigma} s'$ (where s' is unique). Then $y \equiv y' \equiv \partial_H(s')$.
 - (b) Otherwise. In contradiction with $T(\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}) \models x \xrightarrow{\sigma} y$.

Having inspected all possible cases, we may now conclude that $y \equiv y'$. ■

Definition 6.3.3.5 (Bisimulation and Bisimulation Model for $\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}$)

Bisimulation for $\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}$ and the corresponding bisimulation model are defined in the same way as for $\text{BPA}_{\text{drt}, \varepsilon}^- \text{-ID}$. Replace “ $\text{BPA}_{\text{drt}, \varepsilon}^- \text{-ID}$ ” by “ $\text{ACP}_{\text{drt}, \varepsilon}^- \text{-ID}$ ” in Definition 6.3.1.6 on page 182 and Definition 6.3.1.7 on page 182.

$$\begin{array}{c}
\frac{x \xrightarrow{a} x', y \xrightarrow{b} y', \gamma(a, b) = c}{x \parallel y \xrightarrow{c} x' \parallel y'} \quad \frac{x \xrightarrow{a} x', y \xrightarrow{b} y', \gamma(a, b) = c}{x \mid y \xrightarrow{c} x' \parallel y'} \quad \frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x \mid y \xrightarrow{\sigma} x' \mid y'} \\
\\
\frac{x \xrightarrow{a} x', a \notin H}{\partial_H(x) \xrightarrow{a} \partial_H(x')} \quad \frac{x \downarrow}{\partial_H(x) \downarrow} \quad \frac{x \xrightarrow{\sigma} x'}{\partial_H(x) \xrightarrow{\sigma} \partial_H(x')}
\end{array}$$

Table 6.6: Additional deduction rules for $ACP_{\text{drt}, \varepsilon}^-$ -ID.**Definition 6.3.3.6 (Basic Terms of $ACP_{\text{drt}, \varepsilon}^-$ -ID)**

When we speak of basic terms in the context of $ACP_{\text{drt}, \varepsilon}^-$ -ID, we mean $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ -basic terms as defined in Definition 6.3.1.8 on page 182.

Theorem 6.3.3.7 (Elimination for $ACP_{\text{drt}, \varepsilon}^-$ -ID)

Let t be a closed $ACP_{\text{drt}, \varepsilon}^-$ -ID term. Then there is a basic term s such that $BPA_{\text{drt}, \varepsilon}^-$ -ID $\vdash s = t$.

Proof Use the direct method described in Proof Outline 4.2.1.2 on page 68. We give no details. ■

Theorem 6.3.3.8 (Soundness of $ACP_{\text{drt}, \varepsilon}^-$ -ID)

The set of closed $ACP_{\text{drt}, \varepsilon}^-$ -ID terms modulo bisimulation equivalence is a model of the axioms of $ACP_{\text{drt}, \varepsilon}^-$ -ID.

Proof Use the direct method described in Proof Outline 4.2.2.1 on page 69. We give no details. ■

Theorem 6.3.3.9 (Completeness of $ACP_{\text{drt}, \varepsilon}^-$ -ID)

The axiom system $ACP_{\text{drt}, \varepsilon}^-$ -ID is a complete axiomatization of the set of closed $ACP_{\text{drt}, \varepsilon}^-$ -ID terms modulo bisimulation equivalence.

Proof Use Verhoef's method described in Proof Outline 4.2.3.4 on page 71. We give no details. ■

Proposition 6.3.3.10 (Properties of $ACP_{\text{drt}, \varepsilon}^-$ -ID)

Let x be a basic term and $a \in A_\delta$. Then the following properties hold:

- (i). $ACP_{\text{drt}, \varepsilon}^-$ -ID $\vdash x \parallel \underline{\varepsilon} = x$
- (ii). $ACP_{\text{drt}, \varepsilon}^-$ -ID $\vdash \underline{\varepsilon} \parallel x = x$
- (iii). $ACP_{\text{drt}, \varepsilon}^-$ -ID $\vdash \underline{a} \parallel x = \underline{a} \cdot x$
- (iv). $ACP_{\text{drt}, \varepsilon}^-$ -ID $\vdash \underline{\delta} \parallel x = \underline{\delta}$
- (v). $ACP_{\text{drt}, \varepsilon}^-$ -ID $\vdash x \parallel \underline{\underline{\varepsilon}} = x$

Proof

(i). We prove this by induction on the structure of basic terms.

- (a) $x \equiv \underline{\underline{\epsilon}}$. Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel \underline{\underline{\epsilon}} = \underline{\underline{\epsilon}} \parallel \underline{\underline{\epsilon}} = \underline{\underline{\epsilon}} \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel \underline{\underline{\epsilon}} \mid \underline{\underline{\epsilon}} = \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} + \underline{\underline{\delta}} = \underline{\underline{\epsilon}} = x$.
- (b) $x \equiv \underline{\underline{a}} \cdot t$ for $a \in A_\delta$ and a basic term t . Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel \underline{\underline{\epsilon}} = \underline{\underline{a}} \cdot t \parallel \underline{\underline{\epsilon}} = \underline{\underline{a}} \cdot t \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel \underline{\underline{a}} \cdot t + \underline{\underline{a}} \cdot t \mid \underline{\underline{\epsilon}} = \underline{\underline{a}} \cdot (t \parallel \underline{\underline{\epsilon}}) + \underline{\underline{\delta}} + \underline{\underline{\delta}} = \underline{\underline{a}} \cdot t + \underline{\underline{\delta}} + \underline{\underline{\delta}} = \underline{\underline{a}} \cdot t = x$.
- (c) $x \equiv s + t$ for basic terms s and t . Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel \underline{\underline{\epsilon}} = (s + t) \parallel \underline{\underline{\epsilon}} = (s + t) \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel (s + t) + (s + t) \mid \underline{\underline{\epsilon}} = s \parallel \underline{\underline{\epsilon}} + t \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel s + \underline{\underline{\epsilon}} \parallel t + s \mid \underline{\underline{\epsilon}} + t \mid \underline{\underline{\epsilon}} = s \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel s + s \mid \underline{\underline{\epsilon}} + t \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel t + t \mid \underline{\underline{\epsilon}} = s \parallel \underline{\underline{\epsilon}} + t \parallel \underline{\underline{\epsilon}} = s + t = x$.
- (d) $x \equiv \underline{\underline{\sigma}} \cdot t$ for a basic term t . Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel \underline{\underline{\epsilon}} = \underline{\underline{\sigma}} \cdot t \parallel \underline{\underline{\epsilon}} = \underline{\underline{\sigma}} \cdot t \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel \underline{\underline{\sigma}} \cdot t + \underline{\underline{\sigma}} \cdot t \mid \underline{\underline{\epsilon}} = \underline{\underline{\sigma}} \cdot t + \underline{\underline{\delta}} + \underline{\underline{\delta}} = \underline{\underline{\sigma}} \cdot t = x$.

(ii). Using (i), we have $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash \underline{\underline{\epsilon}} \parallel x = \underline{\underline{\epsilon}} \parallel x + x \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \mid x = \underline{\underline{\epsilon}} \parallel x + x \parallel \underline{\underline{\epsilon}} + \underline{\underline{\delta}} = x \parallel \underline{\underline{\epsilon}} + \underline{\underline{\epsilon}} \parallel x + x \mid \underline{\underline{\epsilon}} = x \parallel \underline{\underline{\epsilon}} = x$.

(iii). Using (ii), we have $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash \underline{\underline{a}} \parallel x = \underline{\underline{a}} \cdot \underline{\underline{\epsilon}} \parallel x = \underline{\underline{a}} \cdot (\underline{\underline{\epsilon}} \parallel x) = \underline{\underline{a}} \cdot x$.

(iv). Using (iii), we have $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash \underline{\underline{\delta}} \parallel x = \underline{\underline{\delta}} \cdot x = \underline{\underline{\delta}}$.

(v). We again prove this by induction on the structure of basic terms.

- (a) $x \equiv \underline{\underline{\epsilon}}$. Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel \underline{\underline{\epsilon}} = \underline{\underline{\epsilon}} \parallel \underline{\underline{\epsilon}} = \underline{\underline{\epsilon}} = x$.
- (b) $x \equiv \underline{\underline{a}} \cdot t$ for $a \in A_\delta$ and a basic term t . Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel \underline{\underline{\epsilon}} = \underline{\underline{a}} \cdot t \parallel \underline{\underline{\epsilon}} = \underline{\underline{a}} \cdot (t \parallel \underline{\underline{\epsilon}}) = \underline{\underline{a}} \cdot t = x$.
- (c) $x \equiv s + t$ for basic terms s and t . Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash (s + t) \parallel \underline{\underline{\epsilon}} = s \parallel \underline{\underline{\epsilon}} + t \parallel \underline{\underline{\epsilon}} = s + t = x$.
- (d) $x \equiv \underline{\underline{\sigma}} \cdot t$ for a basic term t . Then $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel \underline{\underline{\epsilon}} = \underline{\underline{\sigma}} \cdot t \parallel \underline{\underline{\epsilon}} = \underline{\underline{\sigma}} \cdot t = x$.

■

Theorem 6.3.3.11 (Axioms of Standard Concurrency for $ACP_{\text{drt},\epsilon}^- \text{-ID}$)

Let x, y, z be closed $ACP_{\text{drt},\epsilon}^- \text{-ID}$ terms. Then the following properties hold:

- (i). $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \mid y = y \mid x$
- (ii). $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel y = y \parallel x$
- (iii). $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash (x \mid y) \mid z = x \mid (y \mid z)$
- (iv). $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$
- (v). $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \mid (y \parallel z) = (x \mid y) \parallel z$
- (vi). $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$

Proof By Theorem 6.3.3.7 assume that $x, y,$ and z are basic terms. Prove (i) by structural induction on both x and y . Then, using (i), we have that $ACP_{\text{drt},\epsilon}^- \text{-ID} \vdash x \parallel y = x \parallel y + y \parallel x + x \mid y = y \parallel x + x \parallel y + y \mid x = y \parallel x$, which proves (ii). Items (iii)–(vi) are finally proven together, using simultaneous induction in the manner of the proof of item (ii) and (iii) of Theorem 6.3.2.16 on page 195. We give no details. ■

Corollary 6.3.3.12 (Commutativity and Associativity of the Merge)

For closed terms, the merge and communication merge operators of $ACP_{drt,\varepsilon}^-$ -ID are commutative and associative.

Proof This follows directly from Theorem 6.3.3.11 on the page before. ■

6.3.4 Immediate Deadlock versus the Empty Process

In this section, we make a small detour into (the absence of) the process algebra $BPA_{drt,\varepsilon}^-$. This process algebra should be like $BPA_{drt,\varepsilon}^-$ -ID with immediate deadlock added. As we will argue, the addition of the immediate deadlock to $BPA_{drt,\varepsilon}^-$ -ID is not very well conceivable, i.e., appears to lead inescapably to inconsistencies.

Let us naively try to introduce the immediate deadlock like we did for BPA_{drt}^- . We then get the axioms shown in Table 6.7.

$$\begin{array}{ll} \underline{\sigma} \cdot \dot{\delta} = \underline{\delta} & \text{DRTESID} \\ x + \dot{\delta} = x & \text{A6ID} \\ \dot{\delta} \cdot x = \dot{\delta} & \text{A7ID} \end{array}$$

Table 6.7: Axioms for immediate deadlock with empty process

Here Axioms A6ID and A7ID express the deadlock property of the immediate deadlock, and Axiom DRTESID expresses the fact that the immediate deadlock prohibits the further passage of time: stopping time at the beginning of the next time-slice (the process $\underline{\sigma} \cdot \dot{\delta}$) is the same as stopping time at the end of the current time-slice (the process $\underline{\delta}$).

The introduction of Axiom A6ID now makes it necessary to drop Axiom A6: as $\dot{\delta}$ now plays the role of the zero element for the choice, the equality $x + \underline{\delta} = x$ does not hold anymore for $x = \dot{\delta}$, as we have $\underline{\delta} + \dot{\delta} = \underline{\delta}$ by Axiom A6ID. To compensate for this, we will have to introduce axioms to the effect that $x + \underline{\delta} = x$ for $x \neq \dot{\delta}$. So, for example, we could introduce the axioms shown in Table 6.8.

$$\begin{array}{ll} \underline{a} + \underline{\delta} = \underline{a} & \text{DRTE5} \\ \underline{\sigma} + \underline{\delta} = \underline{\sigma} & \text{DRTE6} \\ \underline{\varepsilon} + \underline{\delta} = \underline{\varepsilon} & \text{DRTE7} \end{array}$$

Table 6.8: Weakened axioms for undelayable deadlock with empty process

Unfortunately, this leads already to an inconsistency, as we now have:

$$\dot{\delta} = \underline{\varepsilon} \cdot \dot{\delta} = (\underline{\varepsilon} + \underline{\delta}) \cdot \dot{\delta} = \underline{\varepsilon} \cdot \dot{\delta} + \underline{\delta} \cdot \dot{\delta} = \dot{\delta} + \underline{\delta} = \underline{\delta}$$

Hence, $\dot{\delta} = \underline{\underline{\delta}}$, and the whole exercise is in vain.

What to do? We have several options. We could for example weaken Axioms DRTE3 and DRTE4 so that x is not allowed to be $\dot{\delta}$ anymore. This could be achieved by weakening Axioms DRTE3 and DRTE4 to Axioms DRTE3ID and DRTE4ID from Table 6.9.

$$\begin{aligned} (x + \underline{\underline{\delta}}) \cdot \underline{\underline{\varepsilon}} &= x + \underline{\underline{\delta}} && \text{DRTE3ID} \\ \underline{\underline{\varepsilon}} \cdot (x + \underline{\underline{\delta}}) &= x + \underline{\underline{\delta}} && \text{DRTE4ID} \end{aligned}$$

Table 6.9: Weakened axioms for empty process and sequential composition.

However, this does not get us far, as we now have to decide what the process $\underline{\underline{\varepsilon}} \cdot \dot{\delta}$ should be. As shown above, we cannot let it be $\dot{\delta}$, as that leads to $\underline{\underline{\delta}} = \dot{\delta}$. Then, the next logical candidate is $\underline{\underline{\delta}}$. However, putting $\underline{\underline{\varepsilon}} \cdot \dot{\delta} = \underline{\underline{\delta}}$ leads to another inconsistency, as we then have:

$$\underline{\underline{a}} \cdot \dot{\delta} = (\underline{\underline{a}} \cdot \underline{\underline{\varepsilon}}) \cdot \dot{\delta} = \underline{\underline{a}} \cdot (\underline{\underline{\varepsilon}} \cdot \dot{\delta}) = \underline{\underline{a}} \cdot \underline{\underline{\delta}}$$

So, $\underline{\underline{a}} \cdot \dot{\delta} = \underline{\underline{a}} \cdot \underline{\underline{\delta}}$, and that is not what we want either.

Our last resort now is to introduce, next to $\dot{\delta}$, a brand new constant $\dot{\varepsilon}$, the immediate empty process, that intuitively denotes something like an “immediate successful termination option”. The idea is that $\dot{\varepsilon}$ can terminate (like $\underline{\underline{\varepsilon}}$), but cannot idle (like $\dot{\delta}$). This $\dot{\varepsilon}$ would then be a true unit element for the sequential composition. However, the role of the undelayable actions would be seriously changed, as we now need to consider the idling behavior within a time-slice. So, for example, the process $\underline{\underline{a}}$ now differs from the process $\underline{\underline{a}} \cdot \underline{\underline{\varepsilon}}$, as the second can still idle after it has executed an a , while the first cannot. If we fail to make this distinction, we run into the same problems as before. As a result, we have to construct quite different term-deduction rules, leading to a different model. Since one of our design goals was to build on existing discrete-time process algebras, and not invent totally different ones, we will not further explore the $\dot{\varepsilon}$ option sketched here. (Note that the above considerations are not specifically bound to discrete-time process algebra; the incompatibility also arises in untimed process algebra. The reason that we treat this problem in this chapter, and not in Chapter 2, is that we felt that due to the technical nature of the problem, it would not fit in with the introductory character of that chapter.)

Taking into account the above, we conclude that in the current framework $\underline{\underline{\varepsilon}}$ and $\dot{\delta}$ do not combine in any useful way, and we will not consider $\dot{\delta}$ any further in this chapter.

This is very unfortunate, as $\dot{\delta}$ could potentially be quite useful. Consider for example Lemma 6.3.1.12 on page 183. If we would have a consistent way to define a process algebra $\text{BPA}_{\text{drt}, \varepsilon}^-$, for all terms t there would be a term s such that $t + \underline{\underline{\delta}} = \nu(t) + \underline{\underline{\sigma}} \cdot s$ (in those cases where $t = \nu(t)$, we would take $s \equiv \dot{\delta}$, as then $\nu(t) + \underline{\underline{\sigma}} \cdot \dot{\delta} = \nu(t) + \underline{\underline{\delta}} = t + \underline{\underline{\delta}}$). So, Lemma 6.3.1.12 would only have one case instead of two, and as a result the proof on page 196 would only need to examine *one* case under item (i)(d), instead of *four* as it does now.

In this way, the presence of δ would enable us to collapse quite a few of the case distinctions with which our proofs are riddled. Strangely enough, in the conclusion of Chapter 5 we argued (on equally valid grounds), that indeed the presence of $\underline{\underline{\varepsilon}}$ would collapse a lot of the case distinctions made there. It seems very ironical you cannot have them both.

6.4 Process Algebras with Delayable Actions

Now, we extend the process algebras of Section 6.3 with *delayable actions*. We define the process algebra $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$, and step-by-step extend it with the free merge and the merge. For each extension we give an axiomatization, an operational semantics, and a description of all concepts that are introduced. Furthermore, we give the considerations that have led us to construct these algebras in the way we have done.

6.4.1 $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$

In this section, we define the process algebra $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$, which is basically the process algebra $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$ extended with delayable actions (note the absence of the superscript “-”).

Definition 6.4.1.1 (Signature of $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$)

The signature of $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$ consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *delayable actions* $\{a \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *delayable deadlock constant* δ , the *time-unit delay constant* $\underline{\underline{\sigma}}$, the *undelayable empty process constant* $\underline{\underline{\varepsilon}}$, the *delayable empty process constant* ε , the *alternative composition operator* $+$, the *sequential composition operator* \cdot , and the “now” operator ν_{rel} .

Remark 6.4.1.2 (The Delayable Empty Process)

The *delayable empty process* ε in the signature of $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$ is the delayable counterpart of $\underline{\underline{\varepsilon}}$: it can terminate successfully in the current time-slice, but it can also move on to a following time-slice.

Definition 6.4.1.3 (Axioms of $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$)

The process algebra $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$ is axiomatized by the axioms of $\text{BPA}_{\text{drt},\varepsilon}^- \text{-ID}$ given in Definition 6.3.1.3 on page 179, and Axioms DEP and DA shown in Table 6.10: $\text{BPA}_{\text{drt},\varepsilon}\text{-ID} = \text{A1-A5} + \text{DRTE1-DRTE4} + \text{TF} + \text{DCSE1-DCSE4} + \text{DEP} + \text{DA}$.

$$\begin{array}{ll} \varepsilon = \underline{\underline{\varepsilon}} + \underline{\underline{\sigma}} \cdot \varepsilon & \text{DEP} \\ a = \varepsilon \cdot \underline{a} & \text{DA} \end{array}$$

Table 6.10: Additional axioms for $\text{BPA}_{\text{drt},\varepsilon}\text{-ID}$.

☞ Axiom DEP defines the ε : the choice between terminating now, or moving on to the next time-slice. Axiom DA then uses ε to define the delayable actions: a delayable action corresponds to moving to some time-slice, followed by the execution of an a .

Definition 6.4.1.4 (Recursion Principle for ε)

Besides the axioms mentioned in Definition 6.4.1.3, the system $\text{BPA}_{\text{drt},\varepsilon}^+$ -ID also contains the *recursion principle* RSP(DEP) shown in Table 6.11.

$$y = x + \underline{\underline{\sigma}} \cdot y \quad \Rightarrow \quad y = \varepsilon \cdot x \quad \text{RSP(DEP)}$$

Table 6.11: Recursive specification principle for delayable empty process.

☞ Compare RSP(DEP) with RSP(USD) from Table 3.11 on page 56. Intuitively, we have the equality $[x]^\omega = \varepsilon \cdot \nu_{\text{rel}}(x)$.

Remark 6.4.1.5 (The Recursion Principle RSP(DEP))

The recursion principle RSP(DEP) will be used to derive equalities between terms that contain delayable actions. As it turns out, RSP(DEP) is very powerful: for every new operator we add to $\text{BPA}_{\text{drt},\varepsilon}^+$ -ID, we only have to add axioms to eliminate this operator from terms that contain only *undelayable* actions. The elimination of the new operator from terms that also contain *delayable* actions is then possible using RSP(DEP).

The price to be paid for this power, is the fact that RSP(DEP) is formulated as a conditional axiom. Hence, we lose the strict equationality of our process algebra. If so desired, it is possible to maintain strict equationality, but that requires the addition of new axioms to deal with delayable actions for every new operator added. For a description and examples on how to do this in a structured manner, see Section 5.5.

In Example 6.4.1.6 we show how to apply RSP(DEP) to derive simple equalities in $\text{BPA}_{\text{drt},\varepsilon}^+$ -ID. In Example 6.4.2.10 on page 208 we show how to apply it to derive equalities involving a newly introduced operator, namely the free merge.

Example 6.4.1.6 (Use of RSP(DEP))

First, we show how to derive the equality $a = \varepsilon \cdot a$ in $\text{BPA}_{\text{drt},\varepsilon}^+$ -ID. To begin with, we derive $a = \varepsilon \cdot \underline{\underline{a}} = (\underline{\underline{\varepsilon}} + \underline{\underline{\sigma}} \cdot \varepsilon) \cdot \underline{\underline{a}} = \underline{\underline{\varepsilon}} \cdot \underline{\underline{a}} + \underline{\underline{\sigma}} \cdot \varepsilon \cdot \underline{\underline{a}} = \underline{\underline{a}} + \underline{\underline{\sigma}} \cdot a$. Using this equality, we can then derive $a = \underline{\underline{a}} + \underline{\underline{\sigma}} \cdot a = \underline{\underline{a}} + \underline{\underline{\sigma}} \cdot a + \underline{\underline{\sigma}} \cdot a = a + \underline{\underline{\sigma}} \cdot a$. Applying RSP(DEP), we now get the desired result $a = \varepsilon \cdot a$. Note that the converse, namely $a = a \cdot \varepsilon$, does not hold, as $a \cdot \varepsilon$ can still idle after it has done an a , while a cannot.

Secondly, we show how to derive the equality $\varepsilon \cdot x + \varepsilon \cdot y = \varepsilon \cdot (x + y)$. Applying the axioms, we get $\varepsilon \cdot x + \varepsilon \cdot y = (\underline{\underline{\varepsilon}} + \underline{\underline{\sigma}} \cdot \varepsilon) \cdot x + (\underline{\underline{\varepsilon}} + \underline{\underline{\sigma}} \cdot \varepsilon) \cdot y = \underline{\underline{\varepsilon}} \cdot x + \underline{\underline{\sigma}} \cdot \varepsilon \cdot x + \underline{\underline{\varepsilon}} \cdot y + \underline{\underline{\sigma}} \cdot \varepsilon \cdot y = x + y + \underline{\underline{\sigma}} \cdot (\varepsilon \cdot x + \varepsilon \cdot y)$. Applying RSP(DEP), we now get the desired result $\varepsilon \cdot x + \varepsilon \cdot y = \varepsilon \cdot (x + y)$.

Thirdly, we show how to derive the equality $\underline{\underline{\sigma}} \cdot \varepsilon = \varepsilon \cdot \underline{\underline{\sigma}}$. Applying the axioms, we get $\underline{\underline{\sigma}} \cdot \varepsilon = \underline{\underline{\sigma}} \cdot (\underline{\underline{\varepsilon}} + \underline{\underline{\sigma}} \cdot \varepsilon) = \underline{\underline{\sigma}} \cdot \underline{\underline{\varepsilon}} + \underline{\underline{\sigma}} \cdot \underline{\underline{\sigma}} \cdot \varepsilon = \underline{\underline{\sigma}} + \underline{\underline{\sigma}} \cdot \underline{\underline{\sigma}} \cdot \varepsilon$. Applying RSP(DEP), we now get the desired result $\underline{\underline{\sigma}} \cdot \varepsilon = \varepsilon \cdot \underline{\underline{\sigma}}$.

Definition 6.4.1.7 (Semantics of $\text{BPA}_{\text{drt},\varepsilon}$ -ID)

The semantics of $\text{BPA}_{\text{drt},\varepsilon}$ -ID are given by the term-deduction system $T(\text{BPA}_{\text{drt},\varepsilon}\text{-ID})$, induced by the deduction rules for $\text{BPA}_{\text{drt},\varepsilon}^-$ -ID given in Definition 6.3.1.4 on page 180, and the additional deduction rules for $\text{BPA}_{\text{drt},\varepsilon}$ -ID shown in Table 6.12 on the next page.

☞ The new deduction rules are very simple: a , δ , and ε can do a time-step to themselves, a can do an a -step to $\underline{\underline{a}}$, and ε has the option to terminate.

$$a \xrightarrow{a} \underline{\underline{\varepsilon}} \quad a \xrightarrow{\sigma} a \quad \delta \xrightarrow{\sigma} \delta \quad \varepsilon \xrightarrow{\sigma} \varepsilon \quad \varepsilon \downarrow$$

Table 6.12: Additional deduction rules for $BPA_{\text{drt},\varepsilon}\text{-ID}$.**Theorem 6.4.1.8 (Time Determinism for $BPA_{\text{drt},\varepsilon}\text{-ID}$)**

Let x , y , and y' be closed $BPA_{\text{drt},\varepsilon}\text{-ID}$ terms. Then we have:

$$T(BPA_{\text{drt},\varepsilon}\text{-ID}) \models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y' \implies y \equiv y'$$

Proof Next to the cases treated in Theorem 6.3.1.5 on page 181, we only have to examine the cases $x \equiv a$ for $a \in A_\delta$ and $x \equiv \varepsilon$. For all these cases we have that $T(BPA_{\text{drt},\varepsilon}\text{-ID}) \models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y'$ implies that $y \equiv y' \equiv x$, and we are done. ■

Definition 6.4.1.9 (Bisimulation and Bisimulation Model for $BPA_{\text{drt},\varepsilon}\text{-ID}$)

Bisimulation for $BPA_{\text{drt},\varepsilon}\text{-ID}$ and the corresponding bisimulation model are defined in the same way as for $BPA_{\text{drt},\varepsilon}^-\text{-ID}$. Replace “ $BPA_{\text{drt},\varepsilon}^-\text{-ID}$ ” by “ $BPA_{\text{drt},\varepsilon}\text{-ID}$ ” in Definition 6.3.1.6 on page 182 and Definition 6.3.1.7 on page 182.

Definition 6.4.1.10 (Basic Terms of $BPA_{\text{drt},\varepsilon}\text{-ID}$)

We define $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic terms inductively as follows:

- (i). The constant $\underline{\underline{\varepsilon}}$ is a $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic term,
- (ii). if $a \in A_\delta$ and t is a $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic term, then $\underline{\underline{a}} \cdot t$ is a $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic term,
- (iii). if t and s are $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic terms, then $t + s$ is a $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic term,
- (iv). if t is a $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic term, then $\underline{\underline{\sigma}} \cdot t$ and $\varepsilon \cdot t$ are $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic terms.

From now on, when we speak of basic terms in the context of $BPA_{\text{drt},\varepsilon}\text{-ID}$, we mean $(\underline{\underline{\sigma}}, \underline{\underline{\delta}}, \underline{\underline{\varepsilon}}, \delta, \varepsilon)$ -basic terms.

Theorem 6.4.1.11 (General Form of Basic Terms of $BPA_{\text{drt},\varepsilon}\text{-ID}$)

Modulo the commutativity and associativity of the $+$, all basic terms t of $BPA_{\text{drt},\varepsilon}^-\text{-ID}$ are of the form:

$$t \equiv \sum_{i < m} \underline{\underline{a}}_i \cdot s_i + \sum_{j < n} \underline{\underline{\sigma}} \cdot u_j + \sum_{k < p} \varepsilon \cdot v_k + \sum_{l < q} \underline{\underline{\varepsilon}}$$

for $m, n, p, q \in \mathbb{N}$, $m + n + p + q \geq 1$, $a_i \in A_\delta$, and basic terms s_i , u_j , and v_k .

Proof Trivial, by inspection of the definition of basic terms, Definition 6.4.1.10. Observe that the general form of basic terms is closed under the formation rules given in Definition 6.4.1.10. ■

Theorem 6.4.1.12 (Elimination for $BPA_{\text{drt},\varepsilon}^+\text{-ID}$)

Let t be a closed $BPA_{\text{drt},\varepsilon}\text{-ID}$ term. Then there is a basic term s such that $BPA_{\text{drt},\varepsilon}^+\text{-ID} \vdash s = t$.

Proof Use the lexicographical path ordering method described in Proof Outline 4.2.1.1 on page 68. We give no details. ■

Theorem 6.4.1.13 (Soundness of $BPA_{drt,\epsilon}^+$ -ID)

The set of closed $BPA_{drt,\epsilon}$ -ID terms modulo bisimulation equivalence is a model of the axioms of $BPA_{drt,\epsilon}^+$ -ID.

Proof Use the direct method described in Proof Outline 4.2.2.1 on page 69. Furthermore, prove the soundness of the recursion principle RSP(DEP) separately. We give no details. ■

Theorem 6.4.1.14 (Completeness of $BPA_{drt,\epsilon}^+$ -ID)

The axiom system $BPA_{drt,\epsilon}^+$ -ID is a complete axiomatization of the set of closed $BPA_{drt,\epsilon}$ -ID terms modulo bisimulation equivalence.

Proof Use the direct method described in Proof Outline 4.2.3.1 on page 70. We give no details. ■

6.4.2 $PA_{drt,\epsilon}$ -ID

In this section, we define the process algebra $PA_{drt,\epsilon}$ -ID, which is basically the process algebra $BPA_{drt,\epsilon}$ -ID extended with the free merge. Adding the free merge to $BPA_{drt,\epsilon}$ -ID to get $PA_{drt,\epsilon}$ -ID is entirely similar to adding the free merge in the case without delayable actions (treated Section 6.3.2 on page 187), so we contend ourselves with only giving one example to illustrate the new process algebra.

Definition 6.4.2.1 (Signature of $PA_{drt,\epsilon}$ -ID)

The signature of $PA_{drt,\epsilon}$ -ID consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *delayable actions* $\{a \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *delayable deadlock constant* δ , the *time-unit delay constant* $\underline{\sigma}$, the *undelayable empty process constant* $\underline{\epsilon}$, the *delayable empty process constant* ϵ , the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *“now” operator* ν_{rel} , the *free merge operator* \parallel , and the *left merge operator* \llcorner .

Definition 6.4.2.2 (Axioms of $PA_{drt,\epsilon}$ -ID)

The process algebra $PA_{drt,\epsilon}$ -ID is axiomatized by the axioms of $PA_{drt,\epsilon}^-$ -ID given in Definition 6.3.2.2 on page 187, and the Axioms DEP and DA that are shown in Table 6.10 on page 204: $PA_{drt,\epsilon}$ -ID = A1–A5 + DRTE1–DRTE4 + TF + DCSE1–DCSE4 + DEP + DA + DRTEM1–DRTEM12.

Definition 6.4.2.3 (Semantics of $PA_{drt,\epsilon}$ -ID)

The semantics of $PA_{drt,\epsilon}$ -ID are given by the term-deduction system $T(PA_{drt,\epsilon}$ -ID), induced by the deduction rules for $PA_{drt,\epsilon}^-$ -ID given in Definition 6.3.2.7 on page 190, and the additional deduction rules for $BPA_{drt,\epsilon}$ -ID shown in Table 6.12 on the facing page.

Theorem 6.4.2.4 (Time Determinism for $PA_{drt,\epsilon}$ -ID)

Let x , y , and y' be closed $PA_{drt,\epsilon}$ -ID terms. Then we have:

$$T(PA_{drt,\epsilon}\text{-ID}) \models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y' \implies y \equiv y'$$

Proof Proven in the same way as Theorem 6.4.1.8 on the preceding page, extending it with the extra cases given in the proof of Theorem 6.3.2.8 on page 190. We give no details. ■

Definition 6.4.2.5 (Bisimulation and Bisimulation Model for $PA_{drt,\varepsilon}$ -ID)

Bisimulation for $PA_{drt,\varepsilon}$ -ID and the corresponding bisimulation model are defined in the same way as for $BPA_{drt,\varepsilon}$ -ID. Replace “ $BPA_{drt,\varepsilon}$ -ID” by “ $PA_{drt,\varepsilon}$ -ID” in Definition 6.3.1.6 on page 182 and Definition 6.3.1.7 on page 182.

Definition 6.4.2.6 (Basic Terms of $PA_{drt,\varepsilon}$ -ID)

When we speak of basic terms in the context of $PA_{drt,\varepsilon}$ -ID, we mean $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon}, \delta, \varepsilon)$ -basic terms as defined in Definition 6.4.1.10 on page 206.

Theorem 6.4.2.7 (Elimination for $PA_{drt,\varepsilon}^+$ -ID)

Let t be a closed $PA_{drt,\varepsilon}$ -ID term. Then there is a basic term s such that $PA_{drt,\varepsilon}^+$ -ID $\vdash s = t$.

Proof Use the direct method described in Proof Outline 4.2.1.2 on page 68. We give no details. ■

Theorem 6.4.2.8 (Soundness of $PA_{drt,\varepsilon}^+$ -ID)

The set of closed $PA_{drt,\varepsilon}$ -ID terms modulo bisimulation equivalence is a model of the axioms of $PA_{drt,\varepsilon}^+$ -ID.

Proof Use the direct method described in Proof Outline 4.2.2.1 on page 69. We give no details. ■

Theorem 6.4.2.9 (Completeness of $PA_{drt,\varepsilon}^+$ -ID)

The axiom system $PA_{drt,\varepsilon}^+$ -ID is a complete axiomatization of the set of closed $PA_{drt,\varepsilon}$ -ID terms modulo bisimulation equivalence.

Proof Use Verhoef’s method described in Proof Outline 4.2.3.4 on page 71. We give no details. ■

Example 6.4.2.10 (Use of RSP(DEP) with $PA_{drt,\varepsilon}$ -ID)

We show how to derive the equality $a \parallel b = a \cdot b + b \cdot a$. Applying the axioms, we get $a \parallel b = a \parallel b + b \parallel a = (\underline{a} + \underline{\sigma} \cdot a) \parallel b + (\underline{b} + \underline{\sigma} \cdot b) \parallel a = \underline{a} \parallel b + \underline{\sigma} \cdot a \parallel (\underline{b} + \underline{\sigma} \cdot b) + \underline{b} \parallel a + \underline{\sigma} \cdot b \parallel (\underline{a} + \underline{\sigma} \cdot a) = \underline{a} \cdot b + \underline{\sigma} \cdot (a \parallel b) + \underline{b} \cdot a + \underline{\sigma} \cdot (b \parallel a) = \underline{a} \cdot b + \underline{b} \cdot a + \underline{\sigma} \cdot (a \parallel b)$. Applying RSP(DEP), and using $\varepsilon \cdot (x + y) = \varepsilon \cdot x + \varepsilon \cdot y$ (see Example 6.4.1.6 on page 205), we now get $a \parallel b = \varepsilon \cdot (\underline{a} \cdot b + \underline{b} \cdot a) = \varepsilon \cdot \underline{a} \cdot b + \varepsilon \cdot \underline{b} \cdot a = a \cdot b + b \cdot a$.

Proposition 6.4.2.11 (Properties of $PA_{drt,\varepsilon}$ -ID)

Let x be a basic term and $a \in A_\delta$. Then the following properties hold:

- (i). $PA_{drt,\varepsilon}$ -ID $\vdash x \parallel \underline{\varepsilon} = x$
- (ii). $PA_{drt,\varepsilon}$ -ID $\vdash \underline{\varepsilon} \parallel x = x$
- (iii). $PA_{drt,\varepsilon}$ -ID $\vdash \underline{a} \parallel x = \underline{a} \cdot x$
- (iv). $PA_{drt,\varepsilon}$ -ID $\vdash \underline{\delta} \parallel x = \underline{\delta}$
- (v). $PA_{drt,\varepsilon}$ -ID $\vdash x \parallel \underline{\varepsilon} = x$

Proof In the same manner as Proposition 6.3.2.14 on page 192. We give no details. ■

Theorem 6.4.2.12 (Axioms of Standard Concurrency for $PA_{drt,\varepsilon}^+$ -ID)

Let x, y, z be closed $PA_{drt,\varepsilon}$ -ID terms. Then the following properties hold:

- (i). $PA_{drt,\varepsilon}^+$ -ID $\vdash x \parallel y = y \parallel x$
- (ii). $PA_{drt,\varepsilon}^+$ -ID $\vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$
- (iii). $PA_{drt,\varepsilon}^+$ -ID $\vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$

Proof In the same manner as Theorem 6.3.2.16 on page 195. We give no details. ■

Corollary 6.4.2.13 (Commutativity and Associativity of the Merge)

For closed terms, the free merge operator of $PA_{drt,\varepsilon}^+$ -ID is commutative and associative.

Proof This follows directly from Theorem 6.4.2.12. ■

6.4.3 ACP_{drt, ε} -ID

In this section, we define the process algebra $ACP_{drt,\varepsilon}$ -ID, which is basically the process algebra $PA_{drt,\varepsilon}$ -ID with the free merge modified to merge. Again, adding the merge is entirely similar to adding the merge in the case without delayable actions (treated in Section 6.3.3 on page 198).

Definition 6.4.3.1 (Signature of $ACP_{drt,\varepsilon}$ -ID)

The signature of $ACP_{drt,\varepsilon}$ -ID consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *delayable actions* $\{a \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *delayable deadlock constant* δ , the *time-unit delay constant* $\underline{\sigma}$, the *undelayable empty process constant* $\underline{\varepsilon}$, the *delayable empty process constant* ε , the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the “now” operator ν_{rel} , the *merge operator* \parallel , the *left merge operator* \parallel , the *communication merge operator* $|$, and the *encapsulation operator* ∂_H .

Definition 6.4.3.2 (Axioms of $ACP_{drt,\varepsilon}$ -ID)

The process algebra $ACP_{drt,\varepsilon}$ -ID is axiomatized by the axioms of $ACP_{drt,\varepsilon}^-$ -ID given in Definition 6.3.3.2 on page 198, and the Axioms DEP and DA shown in Table 6.10 on page 204: $ACP_{drt,\varepsilon}$ -ID = A1-A5 + DRTE1-DRTE4 + TF + DCSE1-DCSE4 + DEP + DA + DRTEM2-DRTEM12 + DRTECM1-DRTECM9 + DRTECF + DRTED1-DRTED6.

Definition 6.4.3.3 (Semantics of $ACP_{drt,\varepsilon}$ -ID)

The semantics of $ACP_{drt,\varepsilon}$ -ID are given by the term-deduction system $T(ACP_{drt,\varepsilon}$ -ID), induced by the deduction rules for $ACP_{drt,\varepsilon}^-$ -ID given in Definition 6.3.3.3 on page 198, and the additional deduction rules for $BPA_{drt,\varepsilon}$ -ID shown in Table 6.12 on page 206.

Theorem 6.4.3.4 (Time Determinism for $ACP_{drt,\varepsilon}$ -ID)

Let x, y , and y' be closed $ACP_{drt,\varepsilon}$ -ID terms. Then we have:

$$T(ACP_{drt,\varepsilon}$$
-ID) $\models x \xrightarrow{\sigma} y, x \xrightarrow{\sigma} y' \implies y \equiv y'$

Proof Proven in the same way as Theorem 6.4.1.8 on page 206, extending it with the extra cases given in the proofs of Theorem 6.3.2.8 on page 190 and Theorem 6.3.3.4 on page 198. We give no details. ■

Definition 6.4.3.5 (Bisimulation and Bisimulation Model for $ACP_{\text{drt},\varepsilon}\text{-ID}$)

Bisimulation for $ACP_{\text{drt},\varepsilon}\text{-ID}$ and the corresponding bisimulation model are defined in the same way as for $BPA_{\text{drt},\varepsilon}\text{-ID}$. Replace “ $BPA_{\text{drt},\varepsilon}\text{-ID}$ ” by “ $ACP_{\text{drt},\varepsilon}\text{-ID}$ ” in Definition 6.3.1.6 on page 182 and Definition 6.3.1.7 on page 182.

Definition 6.4.3.6 (Basic Terms of $ACP_{\text{drt},\varepsilon}\text{-ID}$)

When we speak of basic terms in the context of $ACP_{\text{drt},\varepsilon}\text{-ID}$, we mean $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon}, \delta, \varepsilon)$ -basic terms as defined in Definition 6.4.1.10 on page 206.

Theorem 6.4.3.7 (Elimination for $ACP_{\text{drt},\varepsilon}^+\text{-ID}$)

Let t be a closed $ACP_{\text{drt},\varepsilon}\text{-ID}$ term. Then there is a basic term s such that $ACP_{\text{drt},\varepsilon}^+\text{-ID} \vdash s = t$.

Proof Use the direct method described in Proof Outline 4.2.1.2 on page 68. We give no details. ■

Theorem 6.4.3.8 (Soundness of $ACP_{\text{drt},\varepsilon}^+\text{-ID}$)

The set of closed $ACP_{\text{drt},\varepsilon}\text{-ID}$ terms modulo bisimulation equivalence is a model of the axioms of $ACP_{\text{drt},\varepsilon}^+\text{-ID}$.

Proof Use the direct method described in Proof Outline 4.2.2.1 on page 69. We give no details. ■

Theorem 6.4.3.9 (Completeness of $ACP_{\text{drt},\varepsilon}^+\text{-ID}$)

The axiom system $ACP_{\text{drt},\varepsilon}^+\text{-ID}$ is a complete axiomatization of the set of closed $ACP_{\text{drt},\varepsilon}\text{-ID}$ terms modulo bisimulation equivalence.

Proof Use Verhoef’s method described in Proof Outline 4.2.3.4 on page 71. We give no details. ■

Proposition 6.4.3.10 (Properties of $ACP_{\text{drt},\varepsilon}\text{-ID}$)

Let x be a basic term and $a \in A_\delta$. Then the following properties hold:

- (i). $ACP_{\text{drt},\varepsilon}\text{-ID} \vdash x \parallel \underline{\varepsilon} = x$
- (ii). $ACP_{\text{drt},\varepsilon}\text{-ID} \vdash \underline{\varepsilon} \parallel x = x$
- (iii). $ACP_{\text{drt},\varepsilon}\text{-ID} \vdash \underline{a} \ll x = \underline{a} \cdot x$
- (iv). $ACP_{\text{drt},\varepsilon}\text{-ID} \vdash \underline{\delta} \ll x = \underline{\delta}$
- (v). $ACP_{\text{drt},\varepsilon}\text{-ID} \vdash x \ll \underline{\varepsilon} = x$

Proof In the same manner as Proposition 6.3.3.10 on page 200. We give no details. ■

Theorem 6.4.3.11 (Axioms of Standard Concurrency for $ACP_{\text{drt},\varepsilon}^+\text{-ID}$)

Let x, y, z be closed $ACP_{\text{drt},\varepsilon}\text{-ID}$ terms. Then the following properties hold:

- (i). $ACP_{\text{drt},\varepsilon}^+\text{-ID} \vdash x \mid y = y \mid x$
- (ii). $ACP_{\text{drt},\varepsilon}^+\text{-ID} \vdash x \parallel y = y \parallel x$
- (iii). $ACP_{\text{drt},\varepsilon}^+\text{-ID} \vdash (x \mid y) \mid z = x \mid (y \mid z)$

$$(iv). \text{ACP}_{drt,\varepsilon}^+ \text{-ID} \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$$

$$(v). \text{ACP}_{drt,\varepsilon}^+ \text{-ID} \vdash x \mid (y \parallel z) = (x \mid y) \parallel z$$

$$(vi). \text{ACP}_{drt,\varepsilon}^+ \text{-ID} \vdash (x \parallel y) \parallel z = x \parallel (y \parallel z)$$

Proof In the same manner as Theorem 6.3.3.11 on page 201. We give no details. ■

Corollary 6.4.3.12 (Commutativity and Associativity of the Merge)

For closed terms, the merge and communication merge operators of $\text{ACP}_{drt,\varepsilon}^+ \text{-ID}$ are commutative and associative.

Proof This follows directly from Theorem 6.4.3.11 on the preceding page. ■

6.5 Embeddings

The following embeddings hold between the process algebras given in this chapter, and between them and the process algebras given in previous chapters:

- (i). $\text{BPA}_{\delta\varepsilon} \subseteq \text{BPA}_{drt,\varepsilon}^- \text{-ID}$
- (ii). $\text{PA}_{\varepsilon} \subseteq \text{PA}_{drt,\varepsilon}^- \text{-ID}$
- (iii). $\text{ACP}_{\varepsilon} \subseteq \text{ACP}_{drt,\varepsilon}^- \text{-ID}$
- (iv). $\text{BPA}_{\varepsilon} \subseteq \text{BPA}_{drt,\varepsilon} \text{-ID}$
- (v). $\text{PA}_{\varepsilon} \subseteq \text{PA}_{drt,\varepsilon} \text{-ID}$
- (vi). $\text{ACP}_{\varepsilon} \subseteq \text{ACP}_{drt,\varepsilon} \text{-ID}$
- (vii). $\text{BPA}_{drt}^- \text{-ID} \subseteq \text{BPA}_{drt,\varepsilon}^- \text{-ID}$
- (viii). $\text{PA}_{drt}^- \text{-ID} \subseteq \text{PA}_{drt,\varepsilon}^- \text{-ID}$
- (ix). $\text{ACP}_{drt}^- \text{-ID} \subseteq \text{ACP}_{drt,\varepsilon}^- \text{-ID}$
- (x). $\text{BPA}_{drt,\varepsilon}^- \text{-ID} \subseteq \text{BPA}_{drt,\varepsilon} \text{-ID}$
- (xi). $\text{PA}_{drt,\varepsilon}^- \text{-ID} \subseteq \text{PA}_{drt,\varepsilon} \text{-ID}$
- (xii). $\text{ACP}_{drt,\varepsilon}^- \text{-ID} \subseteq \text{ACP}_{drt,\varepsilon} \text{-ID}$
- (xiii). $\text{BPA}_{drt,\varepsilon}^- \text{-ID} \subseteq \text{PA}_{drt,\varepsilon}^- \text{-ID} \subseteq \text{ACP}_{drt,\varepsilon}^- \text{-ID}$
- (xiv). $\text{BPA}_{drt,\varepsilon} \text{-ID} \subseteq \text{PA}_{drt,\varepsilon} \text{-ID} \subseteq \text{ACP}_{drt,\varepsilon} \text{-ID}$

The embeddings of (i)–(iii) are achieved by projecting a onto \underline{a} for $a \in A_{\delta\varepsilon}$, and everything else onto itself.

The embeddings of (iv)–(vi) can be achieved in two different ways: either by projecting the untimed process a onto the undelayable process \underline{a} for $a \in A_{\delta\varepsilon}$, and everything else onto itself, or by projecting the untimed process a onto the delayable process $a \cdot \varepsilon$ for $a \in A_{\delta\varepsilon}$, and everything else onto itself. Note that projecting the untimed process a

onto the delayable process a for $a \in A_{\delta\varepsilon}$ will *not* work, as the untimed empty process ε is a unit element for the sequential composition in $BPA_{\delta\varepsilon}$, whereas the delayable empty process ε is not a unit element for the sequential composition in $BPA_{\text{drt},\varepsilon}\text{-ID}$; we have $BPA_{\delta\varepsilon} \vdash a = a \cdot \varepsilon$, but $BPA_{\text{drt},\varepsilon}\text{-ID} \not\vdash a = a \cdot \varepsilon$. By projecting a onto $a \cdot \varepsilon$ we do get a proper embedding, as $BPA_{\text{drt},\varepsilon}\text{-ID} \vdash (a \cdot \varepsilon) = (a \cdot \varepsilon) \cdot (\varepsilon \cdot \varepsilon)$ does indeed hold.

The embeddings of (vii)–(ix) are achieved by projecting $\sigma_{\text{rel}}(x)$ onto $\underline{\underline{\sigma}} \cdot x$, and everything else onto itself.

The embeddings of (x)–(xiv) are achieved by projecting everything onto itself.

6.6 Conclusions

We have successfully introduced the empty process in the context of discrete-time process algebra with relative timing. In doing so, we found that there is not much room for choice: the constraints of the unit-element property with respect to sequential composition and merge, associativity of the merge, time determinism, and taking $BPA_{\text{drt}}^- \text{-ID}$ as a basis almost completely determine which course to take. We also found that the empty process cannot straightforwardly be combined with the immediate deadlock process of BAETEN AND BERGSTRA [24, 25].

The axioms we have given lead to a sound and complete axiomatization of our bisimulation model. For closed terms, the axioms of standard concurrency are derivable.

As the behavior of the empty process is not always into accordance with one's first intuition (see Remark 6.3.2.4 on page 188 and Remark 6.3.2.5 on page 189), one should be very careful when verifying protocols, to make sure that the protocol that is coded in process algebra, is indeed the same as the one that is supposed to be under study.

The discrete-time empty process makes for a worthwhile addition to process algebra. It can potentially be very useful in giving a formal semantics to specification languages such as SDL and MSC. It also extends the class of processes that can be finitely specified.

The usefulness of the empty process with respect to real-life protocol verification remains to be determined.

7

Fischer's Protocol

7.1 Introduction

In this chapter we present a simple, yet non-trivial, protocol that relies heavily on time, namely Fischer's Protocol for mutual exclusion (introduced by FISCHER [79], see LAMPORT [124]). We will try to prove the correctness of this protocol using the discrete-time process algebra $ACP_{\text{drt},\tau}$, which is like ACP_{drt} , but has an additional constant $\underline{\tau}$ that denotes a so-called *undelayable silent action*.

The whole point of this exercise lies in the following two questions:

- How well suited is discrete-time process algebra to verify non-trivial systems?
- With non-trivial verifications in mind, what modifications to it are desirable?

7.2 The Protocol

In this section we give a short history of Fischer's Protocol and discuss the proof requirements. After that an informal description of the protocol is given, together with an informal correctness argument.

7.2.1 History

The protocol we examine is a mutual exclusion protocol, first proposed by FISCHER [79], and later studied by, among others, LAMPORT [124], SCHNEIDER, BLOOM, AND MARZULLO [184], ABADI AND LAMPORT [3], and JANSSEN, *et al.* [107]. None of these studies uses algebraic methods to prove correctness; they all rely on some form of temporal logic or Floyd-Hoare logic.

Instead of using atomic test-and-set instructions or semaphores, as is often done to assure mutual exclusion, Fischer's Protocol only assumes atomic reads and writes to a shared variable. (Note that when the first mutual exclusion protocols were developed in the late 1960s all mutual exclusion protocols were of the "shared variable kind", see for example DIJKSTRA [77], KNUTH, [118], DE BRUIJN [56], and LAMPORT [123]. Later on

researchers have more concentrated on the “semaphore kind” of protocol.) Mutual exclusion in Fischer's Protocol is guaranteed by carefully placing bounds on the execution times of the instructions, leading to a protocol which is very simple, but nevertheless relies heavily and non-trivially on timing aspects. This makes it an ideal candidate for the purpose we have in mind, namely to try to verify, using process algebra, a not too difficult protocol which still has quite intricate timing aspects.

7.2.2 Proof Requirements

What does one need to prove in the case of a mutual exclusion protocol? Strange as it may seem, this is not all that clear. In the literature one sees requirements like:

- The actual property of mutual exclusion: only one component may be in its critical section at any time,
- Symmetry between the components,
- No assumptions about the execution times of statements (obviously not satisfied in our case!),
- Liveness: there should always be some process that is able to proceed,
- No starvation: it may not be that a component is permanently prohibited from entering its critical section,
- Various kinds of fairness: each component should get its fair share (in various senses) of being allowed to proceed into its critical section,
- Loosely connectedness: when one component deadlocks (outside its critical section), this should not affect the progress of the other components,
- Minimal overhead: the protocol should make a decision as soon it has enough information to do so,

and even more requirements. Some of these requirements are related (for example: symmetry guarantees most kinds of fairness), and each paper about mutual exclusion seems to have its own favorite subset of which ones to prove.

In the case of Fischer's Protocol we choose, mostly following DIJKSTRA [77] (the earliest paper on mutual exclusion), to prove the following three properties:

- Actual mutual exclusion between the two critical sections,
- Symmetry between the two components,
- No starvation.

We will not try to formalize these properties algebraically, as they do not lend themselves easily to this. This is more due to the shortcomings of the (current) algebraic approach than it is to unwillingness on our part; note for example that many of the above properties *can* indeed be very easily formalized using temporal logic. We will return to this subject in our conclusions.

7.2.3 First Informal Description

We will now describe the protocol in an informal way, giving an informal correctness argument. Assume the existence of a shared variable x , to which atomic reads and writes are possible. Initially x equals zero. In Table 7.1 we give Fischer's Protocol expressed in pseudo code. There are two components, running in parallel. The angle brackets (" \langle ", " \rangle ") denote atomicity, the assignment operator (" $:=$ ") denotes the assignment of a value to a variable, and the equality symbol (" $=$ ") denotes the testing of the equality of two expressions.

Component 1:	Component 2:
<pre> repeat repeat await $\langle x = 0 \rangle$; $\langle x := 1 \rangle$; $\langle \textit{delay} \rangle$; until $\langle x = 1 \rangle$; <i>critical section 1</i>; $\langle x := 0 \rangle$; until false; </pre>	<pre> repeat repeat await $\langle x = 0 \rangle$; $\langle x := 2 \rangle$; $\langle \textit{delay} \rangle$; until $\langle x = 2 \rangle$; <i>critical section 2</i>; $\langle x := 0 \rangle$; until false; </pre>

Table 7.1: Fischer's Protocol, first informal version.

The protocol proceeds as follows. Initially, the value of the shared variable is 0. When Component 1 observes that x is 0, it will write the value 1 to x . After that, it waits for some time, and if x then still has the value 1, it is safe to enter the critical section. Component 2 works in a similar way (using 2 instead of 1), and both components run in parallel.

The mutual exclusion property of the protocol is based on the following observation. The *delay* operation causes Component 1 to wait sufficiently long so that, if Component 2 had read the value of x in its **await** statement before the Component 1 executed its $x := 1$ assignment, then Component 2 will have completed the following $x := 2$ statement. Therefore, it can never happen that Component 1 falls through its **until** statement, entering *critical section 1*, while Component 2 is still about to execute its $x := 2$ assignment. This guarantees mutual exclusion. By symmetry, the argument also holds the other way around.

7.2.4 Second Informal Description

Let us try to make the reasoning from the previous section a bit more solid by exactly indicating the possible durations of the statements. First of all, the **await** statement may take anywhere between 0 and ∞ (infinity) time units after x becomes 0. The assignments $x := i$ are supposed to take between a and a' time units, and the *delay* statements between d and d' time units, for fixed non-negative values $a \leq a'$ and $d \leq d'$ over some totally-ordered time domain. Furthermore, assume that $a' < d$, i.e. the delay always takes longer than an assignment. For simplicity's sake, the read actions $x = i$ are supposed to take 0 time units, and the critical section may take any time, including 0 time units. Writing

$\langle action \rangle_t^{t'}$ for an atomic action that happens between t and t' time units after it has been enabled, we arrive at the protocol of Table 7.2.

Component 1:	Component 2:
<pre> repeat repeat await $\langle x = 0 \rangle_0^\infty$; $\langle x := 1 \rangle_a^{a'}$; $\langle delay \rangle_d^{d'}$; until $\langle x = 1 \rangle_0^0$; critical section 1; $\langle x := 0 \rangle_a^{a'}$; until false; </pre>	<pre> repeat repeat await $\langle x = 0 \rangle_0^\infty$; $\langle x := 2 \rangle_a^{a'}$; $\langle delay \rangle_d^{d'}$; until $\langle x = 2 \rangle_0^0$; critical section 2; $\langle x := 0 \rangle_a^{a'}$; until false; </pre>

Table 7.2: Fischer's Protocol, second informal version.

Remember we assumed that $0 \leq a \leq a' < d \leq d' < \infty$. Now we have that if Component 2 falls through its **await** statement, it will complete its $x := 2$ assignment within at most a' time units. If Component 1 would have happened to complete its $x := 1$ assignment just after Component 2 fell through its **await**, it will take Component 1 at least d time units to complete its *delay*. As $a' < d$, when Component 1 reaches the **until** $\langle x = 1 \rangle$ statement, Component 2 will have completed its $x := 2$ assignment. Therefore, the value of x has stabilized, and Component 2 can safely enter its critical section.

As a final remark: note that Fischer's Protocol can be trivially generalized to any number $n > 2$ of components. This generalization, however, we will not examine.

7.3 Adding Silent Actions

In this section we extend the process algebra ACP_{drt} with some new features, in order to make it suitable for the verification of Fischer's Protocol.

7.3.1 Abstraction

Before we can start with the verification of Fischer's Protocol, we need to introduce the concept of *silent actions*. The idea behind silent actions is that often the identity of certain actions, sometimes called *internal actions*, is not relevant for the correctness of a protocol. In the case of Fischer's Protocol, for example, the assignments to the variable x are irrelevant for the mutual exclusion properties. So, although the variable x is needed to structure the internal behavior of the protocol, we are not interested in it when the external behavior (i.e., the mutual exclusion properties) of the protocol is concerned.

In terms of process algebra, we express this by introducing a special constant $\underline{\tau}$, the so-called *undelayable silent action*, and renaming all internal actions to this constant by means of the so-called *abstraction operator*, denoted τ_I . We then use the special properties of the undelayable silent action to simplify the resulting process. In this way, it is

possible to disregard the internal behavior of the protocol, while still maintaining all relevant aspects of the external behavior. For more information on abstraction, and several examples, see BAETEN AND WEIJLAND [38].

7.3.2 $ACP_{drt,\tau}$

In this section we extend the process algebra ACP_{drt} with silent actions, resulting in the process algebra $ACP_{drt,\tau}$. The definitions we give are taken from BAETEN, BERGSTRA, AND RENIERS [29]. As our concern (contrary to previous chapters) is not with theoretical results, but with practical applicability, we will not give any theorems, or motivations for the axioms. For these issues, we refer to [29]. We will restrict ourselves to definitions only.

Definition 7.3.2.1 (Signature of $ACP_{drt,\tau}$)

The signature of $ACP_{drt,\tau}$ consists of the *undelayable actions* $\{\underline{a} \mid a \in A\}$, the *delayable actions* $\{a \mid a \in A\}$, the *undelayable deadlock constant* $\underline{\delta}$, the *delayable deadlock constant* δ , the *immediate deadlock constant* $\dot{\delta}$, the *undelayable silent action* $\underline{\tau}$, the *delayable silent action* τ , the *alternative composition operator* $+$, the *sequential composition operator* \cdot , the *time-unit delay operator* σ_{rel} , the “*now*” *operator* ν_{rel} , the *unbounded start delay operator* $\lfloor \rfloor^\omega$, the *merge operator* \parallel , the *left merge operator* \llcorner , the *communication merge operator* \lrcorner , the *encapsulation operator* ∂_H , and the *abstraction operator* τ_I .

Definition 7.3.2.2 (Axioms of $ACP_{drt,\tau}$)

The process algebra $ACP_{drt,\tau}$ is axiomatized by the axioms of ACP_{drt} that are given in Definition 5.3.3.2 on page 152 and Axioms DRTB1–DRTB4 and DRTT1–DRTT6 shown in Table 7.3: $ACP_{drt,\tau} = A1\text{--}A5 + A6ID + A7ID + DRT1\text{--}DRT5 + DR\text{TSID} + DCS1\text{--}DCS4 + DCSID + ATS + USD + DR\text{TM2ID}\text{--}DR\text{TM3ID} + DR\text{TM4} + DR\text{TM5ID} + DR\text{TM6} + DR\text{TCM1}\text{--}DR\text{TCM5} + DR\text{TCM6ID}\text{--}DR\text{TCM7ID} + DR\text{TCM12}\text{--}DR\text{TCM13} + DR\text{TCF} + DR\text{TD1}\text{--}DR\text{TD6} + DR\text{TMID1}\text{--}DR\text{TMID4} + DRTB1\text{--}DRTB4 + DRTT1\text{--}DRTT6$.

$x \cdot (\underline{\tau} \cdot (\nu_{rel}(y) + z + \underline{\delta})) + \nu_{rel}(y) = x \cdot (\nu_{rel}(y) + z + \underline{\delta})$	DRTB1
$x \cdot (\underline{\tau} \cdot (\nu_{rel}(y) + z + \underline{\delta})) + z = x \cdot (\nu_{rel}(y) + z + \underline{\delta})$	DRTB2
$x \cdot (\sigma_{rel}(\underline{\tau} \cdot (y + \underline{\delta})) + \nu_{rel}(z)) = x \cdot (\sigma_{rel}(y + \underline{\delta})) + \nu_{rel}(z)$	DRTB3
$x \cdot (\tau \cdot \lfloor y + z + \underline{\delta} \rfloor^\omega + \lfloor y \rfloor^\omega) = x \cdot \lfloor y + z + \underline{\delta} \rfloor^\omega$	DRTB4
$\tau_I(\underline{a}) = \underline{a}$ if $a \notin I$	DRTT1
$\tau_I(\underline{a}) = \underline{\tau}$ if $a \in I$	DRTT2
$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$	DRTT3
$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$	DRTT4
$\tau_I(\sigma_{rel}(x)) = \sigma_{rel}(\tau_I(x))$	DRTT5
$\tau_I(\dot{\delta}) = \dot{\delta}$	DRTT6

Table 7.3: Additional axioms for $ACP_{drt,\tau}$.

Remark 7.3.2.3 (Axioms of $ACP_{drt,\tau}$)

In the axioms of $ACP_{drt,\tau}$ we have that a in an axiom ranges over $A_{\delta\tau}$ instead of over A_δ .

Definition 7.3.2.4 (Semantics of $ACP_{drt,\tau}$)

The semantics of $ACP_{drt,\tau}$ are given by the term-deduction system $T(ACP_{drt,\tau})$ induced by the deduction rules for ACP_{drt} given in Definition 5.3.3.3 on page 153, and the additional deduction rules for $ACP_{drt,\tau}$ shown in Table 7.4.

$\frac{x \xrightarrow{a} x', a \notin I}{\tau_I(x) \xrightarrow{a} \tau_I(x')}$	$\frac{x \xrightarrow{a} x', a \in I}{\tau_I(x) \xrightarrow{\tau} \tau_I(x')}$
$\frac{x \xrightarrow{a} \surd, a \notin I}{\tau_I(x) \xrightarrow{a} \surd}$	$\frac{x \xrightarrow{a} \surd, a \in I}{\tau_I(x) \xrightarrow{\tau} \surd}$
$\frac{x \xrightarrow{\sigma} x'}{\tau_I(x) \xrightarrow{\sigma} \tau_I(x')}$	$\frac{ID(x)}{ID(\tau_I(x))}$

Table 7.4: Additional deduction rules for $ACP_{drt,\tau}$.

Remark 7.3.2.5 (Semantics of $ACP_{drt,\tau}$)

In the deduction rules of $ACP_{drt,\tau}$ we have that a in a deduction rule ranges over A_τ instead of over A .

Definition 7.3.2.6 (Symbol for a Chain of τ 's)

We will write $x \xRightarrow{\tau} y$ to indicate that x can reach y by doing zero or more τ -transitions. Formally, $\xRightarrow{\tau}$ denotes the transitive, reflexive closure of $\xrightarrow{\tau}$.

Definition 7.3.2.7 (Bisimulation for $ACP_{drt,\tau}$, Part I)

Bisimulation for $ACP_{drt,\tau}$ is defined as follows; a binary relation R on closed $ACP_{drt,\tau}$ terms is a *branching tail bisimulation* if the following transfer conditions hold for all closed $ACP_{drt,\tau}$ terms p and q :

- (i). If $R^S(p, q)$ and $T(ACP_{drt,\tau}) \models p \xrightarrow{a} p'$, where $a \in A$, then there exist closed terms q' and q'' , such that $T(ACP_{drt,\tau}) \models q \xRightarrow{\tau} q' \xrightarrow{a} q''$, $R^S(p, q')$, and $R^S(p', q'')$,
- (ii). if $R^S(p, q)$ and $T(ACP_{drt,\tau}) \models p \xrightarrow{\tau} p'$, then there exists a closed term q' , such that $T(ACP_{drt,\tau}) \models q \xRightarrow{\tau} q'$, $R^S(p, q')$, and $R^S(p', q')$,
- (iii). if $R^S(p, q)$ and $T(ACP_{drt,\tau}) \models p \xrightarrow{\sigma} p'$, then there exist closed terms q' and q'' , such that $T(ACP_{drt,\tau}) \models q \xRightarrow{\tau} q' \xrightarrow{\sigma} q''$, $R^S(p, q')$, and $R^S(p', q'')$,
- (iv). if $R^S(p, q)$ and $T(ACP_{drt,\tau}) \models p \xrightarrow{a} \surd$, where $a \in A$, then there exists a closed term q' , such that $T(ACP_{drt,\tau}) \models q \xRightarrow{\tau} q' \xrightarrow{a} \surd$ and $R^S(p, q')$.
- (v). if $R^S(p, q)$ and $T(ACP_{drt,\tau}) \models p \xrightarrow{\tau} \surd$, then there exists a closed term q' , such that $T(ACP_{drt,\tau}) \models q \xRightarrow{\tau} q' \xrightarrow{\tau} \surd$ and $R^S(p, q')$,

(vi). if $R^S(p, q)$ and $T(\text{ACP}_{\text{drt}, \tau}) \models \text{ID}(p)$, then $T(\text{ACP}_{\text{drt}, \tau}) \models \text{ID}(q)$.

Two $\text{ACP}_{\text{drt}, \tau}$ terms p and q are *branching tail bisimilar* if there exists a *branching tail bisimulation* relation R such that $R(p, q)$.

Definition 7.3.2.8 (Bisimulation for $\text{ACP}_{\text{drt}, \tau}$, Part II)

If R is a branching tail bisimulation, and for certain $\text{ACP}_{\text{drt}, \tau}$ terms r and s we have $R^S(r, s)$, then we say that the pair (r, s) satisfies the *root condition* if the following transfer conditions hold:

- (i). If $T(\text{ACP}_{\text{drt}, \tau}) \models r \xrightarrow{a} r'$, where $a \in A$, then there exists a closed term s' , such that $T(\text{ACP}_{\text{drt}, \tau}) \models s \xrightarrow{a} s'$ and $R^S(r', s')$,
- (ii). if $T(\text{ACP}_{\text{drt}, \tau}) \models r \xrightarrow{\tau} r'$, then there exists a closed term s' , such that $T(\text{ACP}_{\text{drt}, \tau}) \models s \xrightarrow{\tau} s'$ and $R^S(r', s')$,
- (iii). if $T(\text{ACP}_{\text{drt}, \tau}) \models r \xrightarrow{\sigma} r'$, then there exists a closed term s' , such that $T(\text{ACP}_{\text{drt}, \tau}) \models s \xrightarrow{\sigma} s'$ and $R^S(r', s')$,
- (iv). if $T(\text{ACP}_{\text{drt}, \tau}) \models r \xrightarrow{a} \surd$, where $a \in A$, then $T(\text{ACP}_{\text{drt}, \tau}) \models s \xrightarrow{a} \surd$,
- (v). if $T(\text{ACP}_{\text{drt}, \tau}) \models r \xrightarrow{\tau} \surd$, then that $T(\text{ACP}_{\text{drt}, \tau}) \models s \xrightarrow{\tau} \surd$.

Two $\text{ACP}_{\text{drt}, \tau}$ terms p and q are *rooted branching tail bisimilar*, notation $p \sim_{\text{ACP}_{\text{drt}, \tau}} q$, if there exists a *branching tail bisimulation* relation R such that $R(p, q)$, and for all $\text{ACP}_{\text{drt}, \tau}$ terms r and s we have that if $T(\text{ACP}_{\text{drt}, \tau}) \models p \xrightarrow{\sigma} r, q \xrightarrow{\sigma} s$ and $R(r, s)$, the pair (r, s) satisfies the root condition.

Definition 7.3.2.9 (Bisimulation Model for $\text{ACP}_{\text{drt}, \tau}$)

The bisimulation model for $\text{ACP}_{\text{drt}, \tau}$ is defined in the same way as for BPA. Replace “BPA” by “ $\text{ACP}_{\text{drt}, \tau}$ ” in Definition 2.3.1.16 on page 12.

Remark 7.3.2.10 (Elimination, Soundness, and Completeness of $\text{ACP}_{\text{drt}, \tau}$)

We suppose that the soundness property for $\text{ACP}_{\text{drt}, \tau}$ holds, although we have not tried to prove this. The elimination and completeness properties do *not* hold; we need at least add a recursion principle (or corresponding axioms) before we can properly derive equalities containing delayable actions. For more information on soundness and completeness issues involving abstraction, see BAETEN, BERGSTRA, AND RENIERS [29].

7.4 Specification

Having introduced the relevant operators for $\text{ACP}_{\text{drt}, \tau}$, we are now ready to give a formal specification of Fischer’s Protocol FP_{drt} , using $\text{ACP}_{\text{drt}, \tau}$ in Table 7.5 on the next page. We specify the special case where $a = a' = 0$ and $d = d' = 1$.

When we look at this specification, we see a number of processes (A, B, V, \dots) that are defined in terms of each other. These processes should be considered special constants, that are added to the signature in an ad hoc manner, for the purpose of this one verification. Likewise, the equations of this specification should be considered ad hoc axioms for these new constants. In this way, we can mimic a limited form of *recursion*, without

$$\begin{array}{ll}
A = A_0 & B = B_0 \\
A_0 = r(x = 0) \cdot A_1 & B_0 = r(x = 0) \cdot B_1 \\
A_1 = \underline{s(x := 1)} \cdot A_2 & B_1 = \underline{s(x := 2)} \cdot B_2 \\
A_2 = \sigma_{\text{rel}}(A_3) & B_2 = \sigma_{\text{rel}}(B_3) \\
A_3 = (\underline{r(x = 0)} + \underline{r(x = 2)}) \cdot A_0 + \underline{r(x = 1)} \cdot A_4 & B_3 = (\underline{r(x = 0)} + \underline{r(x = 1)}) \cdot B_0 + \underline{r(x = 2)} \cdot B_4 \\
A_4 = \underline{\text{EnterCS}_1} \cdot A_5 & B_4 = \underline{\text{EnterCS}_2} \cdot B_5 \\
A_5 = \underline{\text{LeaveCS}_1} \cdot A_6 & B_5 = \underline{\text{LeaveCS}_2} \cdot B_6 \\
A_6 = \underline{s(x := 0)} \cdot A_0 & B_6 = \underline{s(x := 0)} \cdot B_0
\end{array}$$

$$\begin{array}{l}
V = V_0 \\
V_0 = (\underline{r(x := 0)} + \underline{s(x = 0)}) \cdot V_0 + \underline{r(x := 1)} \cdot V_1 + \underline{r(x := 2)} \cdot V_2 + \sigma_{\text{rel}}(V_0) \\
V_1 = (\underline{r(x := 1)} + \underline{s(x = 1)}) \cdot V_1 + \underline{r(x := 0)} \cdot V_0 + \underline{r(x := 2)} \cdot V_2 + \sigma_{\text{rel}}(V_1) \\
V_2 = (\underline{r(x := 2)} + \underline{s(x = 2)}) \cdot V_2 + \underline{r(x := 0)} \cdot V_0 + \underline{r(x := 1)} \cdot V_1 + \sigma_{\text{rel}}(V_2)
\end{array}$$

$$\gamma(r(\alpha), s(\alpha)) = c(\alpha) \text{ for } \alpha \in \{x = i, x := i \mid i \in \{0, 1, 2\}\}$$

$$H = \{r(\alpha), s(\alpha) \mid \alpha \in \{x = i, x := i \mid i \in \{0, 1, 2\}\}\}$$

$$\text{FP}_{\text{drt}} = \partial_H(A \parallel B \parallel V)$$

Table 7.5: Fischer's Protocol in $\text{ACP}_{\text{drt}, \tau}$.

the need for formally introducing a full recursion apparatus (which would be quite complicated in the case of $\text{ACP}_{\text{drt}, \tau}$). This method, of course, only works when the equations are reasonably well-behaved, as is the case here.

The specification can be understood intuitively in the following way. There are three processes running concurrently, namely A , B , and V . The processes A and B model “Component 1” and “Component 2” of Table 7.1 on page 215 respectively, and the process V models the variable x .

The process V can be in one of three states: V_0 , V_1 , or V_2 , corresponding to the possible values of x . In any state V_i , V is capable of sending the message $x = i$, signaling that the value of x is currently i , after which V will continue in state V_i . Furthermore, V is in any state V_i capable of receiving the message $x := j$ for any j , indicating that x is being assigned with the value j . After such an assignment V will continue in state V_j . Finally, V is always capable of letting time pass. The process V constructed in this way behaves as a “variable server”: if process A or B wants to assign a value i to x it performs the action $s(x := i)$. If it wants to check if x has the value i , it performs the action $r(x = i)$. (The idea to construct the variable server in this way was taken from NIEUWLAND [156].)

The process A is constructed as follows. First (in state A_0) it waits for an undetermined amount of time till x is 0 ($r(x = 0)$). Then (in state A_1) it sets x to 1 ($s(x := 1)$). After that (in state A_2), it waits till the end of the time slice ($\sigma_{\text{rel}}(A_3)$). When it has arrived in state A_3 , it will examine the contents of x , and either jump back to A_0 (if $x = 0$ or $x = 2$), or continue with state A_4 (if $x = 1$). Thereafter, it enters its critical section, leaves it again, and resets x back to 0 in state A_6 , after which it repeats the whole procedure

over again. The process B is constructed in the same way as A .

The entire protocol, FP_{drt} , now consists of the processes A , B , and V running concurrently, with all $r(\dots)$ and $s(\dots)$ actions encapsulated, thus forcing these actions to communicate to $c(\dots)$ actions. Note that the assignment takes no time ($a = a' = 0$) and the delay takes one time unit ($d = d' = 1$).

7.5 Verification

In order to prove the protocol FP_{drt} correct, we first rewrite the equations of Table 7.5 on the preceding page into an equivalent *linear* system of equations. These equations should only contain the operators $+$, \cdot , and σ , and only undelayable actions. With the equations in this format, we can easily construct the process graph of FP_{drt} .

Finding this linear specification for FP_{drt} is very easy (at least in principle): we repeatedly apply the axioms to expand the merge operators, and throw away the summands that are encapsulated. Starting with FP_{drt} , which we will call X_0 , we expand every state we encounter, where new states we reach are numbered using process variables X_i . In this way, we arrive at the 32 processes X_0, \dots, X_{31} given below:

$$\begin{aligned}\text{FP}_{\text{drt}} &= \partial_H(A \parallel B \parallel V) \\ &= \partial_H(A_0 \parallel B_0 \parallel V_0) \\ &= X_0\end{aligned}$$

$$\begin{aligned}X_0 &= \partial_H(A_0 \parallel B_0 \parallel V_0) \\ &= \underline{c(x=0)} \cdot \partial_H(A_1 \parallel B_0 \parallel V_0) + \underline{c(x=0)} \cdot \partial_H(A_0 \parallel B_1 \parallel V_0) + \\ &\quad \sigma(\partial_H(A_0 \parallel B_0 \parallel V_0)) \\ &= \underline{c(x=0)} \cdot X_1 + \underline{c(x=0)} \cdot X_2 + \sigma(X_0)\end{aligned}$$

$$\begin{aligned}X_1 &= \partial_H(A_1 \parallel B_0 \parallel V_0) \\ &= \underline{c(x:=1)} \cdot \partial_H(A_2 \parallel B_0 \parallel V_1) + \underline{c(x=0)} \cdot \partial_H(A_1 \parallel B_1 \parallel V_0) \\ &= \underline{c(x:=1)} \cdot X_{23} + \underline{c(x=0)} \cdot X_3\end{aligned}$$

$$\begin{aligned}X_2 &= \partial_H(A_0 \parallel B_1 \parallel V_0) \\ &= \underline{c(x=0)} \cdot \partial_H(A_1 \parallel B_1 \parallel V_0) + \underline{c(x:=2)} \cdot \partial_H(A_0 \parallel B_2 \parallel V_2) \\ &= \underline{c(x=0)} \cdot X_3 + \underline{c(x:=2)} \cdot X_4\end{aligned}$$

$$\begin{aligned}X_3 &= \partial_H(A_1 \parallel B_1 \parallel V_0) \\ &= \underline{c(x:=1)} \cdot \partial_H(A_2 \parallel B_1 \parallel V_1) + \underline{c(x:=2)} \cdot \partial_H(A_1 \parallel B_2 \parallel V_2) \\ &= \underline{c(x:=1)} \cdot X_9 + \underline{c(x:=2)} \cdot X_{10}\end{aligned}$$

$$\begin{aligned}
X_4 &= \partial_H(A_0 \parallel B_2 \parallel V_2) \\
&= \sigma(\partial_H(A_0 \parallel B_3 \parallel V_2)) \\
&= \sigma(X_5)
\end{aligned}$$

$$\begin{aligned}
X_5 &= \partial_H(A_0 \parallel B_3 \parallel V_2) \\
&= \underline{\underline{c(x = 2)}} \cdot \partial_H(A_0 \parallel B_4 \parallel V_2) \\
&= \underline{\underline{c(x = 2)}} \cdot X_6
\end{aligned}$$

$$\begin{aligned}
X_6 &= \partial_H(A_0 \parallel B_4 \parallel V_2) \\
&= \underline{\underline{\text{EnterCS}_2}} \cdot \partial_H(A_0 \parallel B_5 \parallel V_2) \\
&= \underline{\underline{\text{EnterCS}_2}} \cdot X_7
\end{aligned}$$

$$\begin{aligned}
X_7 &= \partial_H(A_0 \parallel B_5 \parallel V_2) \\
&= \underline{\underline{\text{LeaveCS}_2}} \cdot \partial_H(A_0 \parallel B_6 \parallel V_2) \\
&= \underline{\underline{\text{LeaveCS}_2}} \cdot X_8
\end{aligned}$$

$$\begin{aligned}
X_8 &= \partial_H(A_0 \parallel B_6 \parallel V_2) \\
&= \underline{\underline{c(x := 0)}} \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \underline{\underline{c(x := 0)}} \cdot X_0
\end{aligned}$$

$$\begin{aligned}
X_9 &= \partial_H(A_2 \parallel B_1 \parallel V_1) \\
&= \underline{\underline{c(x := 2)}} \cdot \partial_H(A_2 \parallel B_2 \parallel V_2) \\
&= \underline{\underline{c(x := 2)}} \cdot X_{24}
\end{aligned}$$

$$\begin{aligned}
X_{10} &= \partial_H(A_1 \parallel B_2 \parallel V_2) \\
&= \underline{\underline{c(x := 1)}} \cdot \partial_H(A_2 \parallel B_2 \parallel V_1) \\
&= \underline{\underline{c(x := 1)}} \cdot X_{11}
\end{aligned}$$

$$\begin{aligned}
X_{11} &= \partial_H(A_2 \parallel B_2 \parallel V_1) \\
&= \sigma(\partial_H(A_3 \parallel B_3 \parallel V_1)) \\
&= \sigma(X_{12})
\end{aligned}$$

$$\begin{aligned}
X_{12} &= \partial_H(A_3 \parallel B_3 \parallel V_1) \\
&= \underline{\underline{c(x = 1)}} \cdot \partial_H(A_4 \parallel B_3 \parallel V_1) + \underline{\underline{c(x = 1)}} \cdot \partial_H(A_3 \parallel B_0 \parallel V_1) \\
&= \underline{\underline{c(x = 1)}} \cdot X_{13} + \underline{\underline{c(x = 1)}} \cdot X_{14}
\end{aligned}$$

$$\begin{aligned}
X_{13} &= \partial_H(A_4 \parallel B_3 \parallel V_1) \\
&= \underline{\text{EnterCS}_1} \cdot \partial_H(A_5 \parallel B_3 \parallel V_1) + \underline{c(x=1)} \cdot \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= \underline{\text{EnterCS}_1} \cdot X_{18} + \underline{c(x=1)} \cdot X_{15}
\end{aligned}$$

$$\begin{aligned}
X_{14} &= \partial_H(A_3 \parallel B_0 \parallel V_1) \\
&= \underline{c(x=1)} \cdot \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= \underline{c(x=1)} \cdot X_{15}
\end{aligned}$$

$$\begin{aligned}
X_{15} &= \partial_H(A_4 \parallel B_0 \parallel V_1) \\
&= \underline{\text{EnterCS}_1} \cdot \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \underline{\text{EnterCS}_1} \cdot X_{16}
\end{aligned}$$

$$\begin{aligned}
X_{16} &= \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \underline{\text{LeaveCS}_1} \cdot \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= \underline{\text{LeaveCS}_1} \cdot X_{17}
\end{aligned}$$

$$\begin{aligned}
X_{17} &= \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= \underline{c(x:=0)} \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \underline{c(x:=0)} \cdot X_0
\end{aligned}$$

$$\begin{aligned}
X_{18} &= \partial_H(A_5 \parallel B_3 \parallel V_1) \\
&= \underline{\text{LeaveCS}_1} \cdot \partial_H(A_6 \parallel B_3 \parallel V_1) + \underline{c(x=1)} \cdot \partial_H(A_5 \parallel B_0 \parallel V_1) \\
&= \underline{\text{LeaveCS}_1} \cdot X_{19} + \underline{c(x=1)} \cdot X_{16}
\end{aligned}$$

$$\begin{aligned}
X_{19} &= \partial_H(A_6 \parallel B_3 \parallel V_1) \\
&= \underline{c(x:=0)} \cdot \partial_H(A_0 \parallel B_3 \parallel V_0) + \underline{c(x=1)} \cdot \partial_H(A_6 \parallel B_0 \parallel V_1) \\
&= \underline{c(x:=0)} \cdot X_{20} + \underline{c(x=1)} \cdot X_{17}
\end{aligned}$$

$$\begin{aligned}
X_{20} &= \partial_H(A_0 \parallel B_3 \parallel V_0) \\
&= \underline{c(x=0)} \cdot \partial_H(A_1 \parallel B_3 \parallel V_0) + \underline{c(x=0)} \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) \\
&= \underline{c(x=0)} \cdot X_{21} + \underline{c(x=0)} \cdot X_0
\end{aligned}$$

$$\begin{aligned}
X_{21} &= \partial_H(A_1 \parallel B_3 \parallel V_0) \\
&= \underline{c(x:=1)} \cdot \partial_H(A_2 \parallel B_3 \parallel V_1) + \underline{c(x=0)} \cdot \partial_H(A_1 \parallel B_0 \parallel V_0) \\
&= \underline{c(x:=1)} \cdot X_{22} + \underline{c(x=0)} \cdot X_1
\end{aligned}$$

$$\begin{aligned}
X_{22} &= \partial_H(A_2 \parallel B_3 \parallel V_1) \\
&= \underline{\underline{c(x=1)}} \cdot \partial_H(A_2 \parallel B_0 \parallel V_1) \\
&= \underline{\underline{c(x=1)}} \cdot X_{23}
\end{aligned}$$

$$\begin{aligned}
X_{23} &= \partial_H(A_2 \parallel B_0 \parallel V_1) \\
&= \sigma(\partial_H(A_3 \parallel B_0 \parallel V_1)) \\
&= \sigma(X_{14})
\end{aligned}$$

$$\begin{aligned}
X_{24} &= \partial_H(A_2 \parallel B_2 \parallel V_2) \\
&= \sigma(\partial_H(A_3 \parallel B_3 \parallel V_2)) \\
&= \sigma(X_{25})
\end{aligned}$$

$$\begin{aligned}
X_{25} &= \partial_H(A_3 \parallel B_3 \parallel V_2) \\
&= \underline{\underline{c(x=2)}} \cdot \partial_H(A_0 \parallel B_3 \parallel V_2) + \underline{\underline{c(x=2)}} \cdot \partial_H(A_3 \parallel B_4 \parallel V_2) \\
&= \underline{\underline{c(x=2)}} \cdot X_5 + \underline{\underline{c(x=2)}} \cdot X_{26}
\end{aligned}$$

$$\begin{aligned}
X_{26} &= \partial_H(A_3 \parallel B_4 \parallel V_2) \\
&= \underline{\underline{c(x=2)}} \cdot \partial_H(A_0 \parallel B_4 \parallel V_2) + \underline{\underline{\text{EnterCS}_2}} \cdot \partial_H(A_3 \parallel B_5 \parallel V_2) \\
&= \underline{\underline{c(x=2)}} \cdot X_6 + \underline{\underline{\text{EnterCS}_2}} \cdot X_{27}
\end{aligned}$$

$$\begin{aligned}
X_{27} &= \partial_H(A_3 \parallel B_5 \parallel V_2) \\
&= \underline{\underline{c(x=2)}} \cdot \partial_H(A_0 \parallel B_5 \parallel V_2) + \underline{\underline{\text{LeaveCS}_2}} \cdot \partial_H(A_3 \parallel B_6 \parallel V_2) \\
&= \underline{\underline{c(x=2)}} \cdot X_7 + \underline{\underline{\text{LeaveCS}_2}} \cdot X_{28}
\end{aligned}$$

$$\begin{aligned}
X_{28} &= \partial_H(A_3 \parallel B_6 \parallel V_2) \\
&= \underline{\underline{c(x=2)}} \cdot \partial_H(A_0 \parallel B_6 \parallel V_2) + \underline{\underline{c(x:=0)}} \cdot \partial_H(A_3 \parallel B_0 \parallel V_0) \\
&= \underline{\underline{c(x=2)}} \cdot X_8 + \underline{\underline{c(x:=0)}} \cdot X_{29}
\end{aligned}$$

$$\begin{aligned}
X_{29} &= \partial_H(A_3 \parallel B_0 \parallel V_0) \\
&= \underline{\underline{c(x=0)}} \cdot \partial_H(A_0 \parallel B_0 \parallel V_0) + \underline{\underline{c(x=0)}} \cdot \partial_H(A_3 \parallel B_1 \parallel V_0) \\
&= \underline{\underline{c(x=0)}} \cdot X_0 + \underline{\underline{c(x=0)}} \cdot X_{30}
\end{aligned}$$

$$\begin{aligned}
X_{30} &= \partial_H(A_3 \parallel B_1 \parallel V_0) \\
&= \underline{\underline{c(x=0)}} \cdot \partial_H(A_0 \parallel B_1 \parallel V_0) + \underline{\underline{c(x:=2)}} \cdot \partial_H(A_3 \parallel B_2 \parallel V_2) \\
&= \underline{\underline{c(x=0)}} \cdot X_2 + \underline{\underline{c(x:=2)}} \cdot X_{31}
\end{aligned}$$

$$\begin{aligned}
 X_{31} &= \partial_H(A_3 \parallel B_2 \parallel V_2) \\
 &= \underline{\underline{c(x = 2)}} \cdot \partial_H(A_0 \parallel B_2 \parallel V_2) \\
 &= \underline{\underline{c(x = 2)}} \cdot X_4
 \end{aligned}$$

From this linear system we then construct, using the deduction rules of $ACP_{drt,\tau}$, the process graph of FP_{drt} . This process graph is shown in Table 7.6. For clarity, we do not label the $c(\dots)$ edges, and abbreviate the four “EnterCS” and “LeaveCS” actions by E1, E2, L1, and L2.

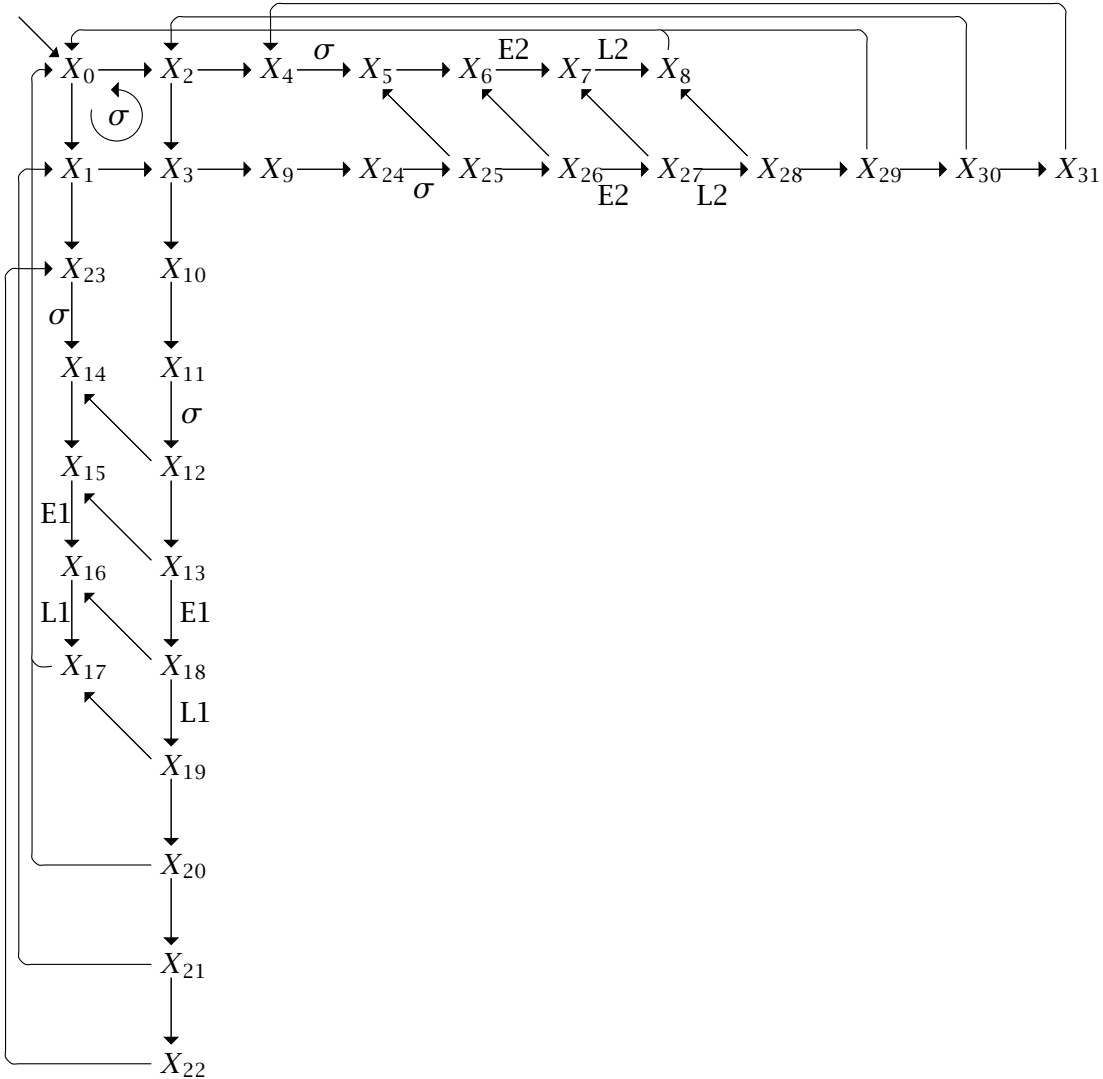


Table 7.6: The process graph of FP_{drt} .

Now define the set I of internal actions as:

$$I = \{c(\alpha) \mid \alpha \in \{x = i, x := i \mid i \in \{0, 1, 2\}\}\}$$

(i.e. all communication actions) and rename these actions into τ , yielding the process graph of $\tau_I(FP_{drt})$. On this graph we compute the maximal rooted branching tail bisimulation between the graph and itself. This gives us the equivalence classes X_A, \dots, X_H

given below:

$$\begin{array}{ll}
 X_A = \{X_0, X_8, X_{17}, X_{19}, X_{20}, X_{28}, X_{29}\} & X_E = \{X_{16}, X_{18}\} \\
 X_B = \{X_1, X_2, X_3, X_{21}, X_{30}\} & X_F = \{X_4, X_9, X_{24}, X_{31}\} \\
 X_C = \{X_{10}, X_{11}, X_{22}, X_{23}\} & X_G = \{X_5, X_6, X_{25}, X_{26}\} \\
 X_D = \{X_{12}, X_{13}, X_{14}, X_{15}\} & X_H = \{X_7, X_{27}\}
 \end{array}$$

When we divide out this equivalence relation we arrive at the reduced process graph shown in Table 7.7, where the unlabeled edges are τ -edges. On this graph we will now

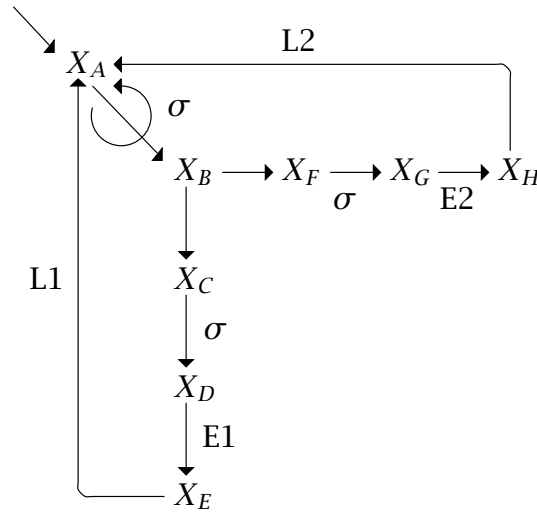


Table 7.7: The reduced process graph of $\tau_I(\text{FP}_{\text{drt}})$.

be able to check all required properties very easily. The three requirements for mutual exclusion we chose in Section 7.2.2 are satisfied for the following reasons:

- **Actual mutual exclusion:** It can be easily seen from the graph of Table 7.7 that an EnterCS_1 action is always immediately followed by a LeaveCS_1 action, and the same holds for EnterCS_2 and LeaveCS_2 . Therefore, it cannot be the case that both components are in their critical section at the same time.
- **Symmetry:** As the graph is symmetrical with respect to the paths from the root, through the critical sections of the components, back to the root, it is clear that the protocol is symmetrical with respect to the components.
- **No starvation:** Whichever state the protocol is in, there is always a path leading to each component's critical section. Therefore, assuming fairness, it cannot be the case that one component is permanently prohibited from entering its critical section.

This completes the proof of the correctness of FP_{drt} for the special case where $a = a' = 0$ and $d = d' = 1$. The proof for the general case where $0 < a < a' < d < d' < 2a$ proceeds along the same lines, although the calculations get much more complicated. In VEREIJKEN [191] this proof is written out in full, not using discrete-time process algebra, but using real-time process algebra.

7.6 Conclusions

As we have seen, Fischer’s protocol can be proven correct quite satisfactorily using techniques from process algebra. This however does not mean too much; it is an almost trivial protocol, that has been solved time and time again using different kinds of formalisms. But it is not all that bad either; see for example SCHNEIDER, BLOOM, AND MARZULLO [184] where an (incomplete) proof is given of Fischer’s protocol. When it would be written out in full detail, that proof would be about as long and tedious as ours is (SCHNEIDER [183]), and the same probably holds for a detailed temporal logic proof (ABADI [2]).

Looking at our proof one observes that, unfortunately, only the first part of it is purely algebraic. In the second part, we need to fall back to model-dependent reasoning, and that is disappointing. Furthermore, although the proof is conceptually very clear and easy, the inner workings required a lot of bothersome and failure-prone computations. It seems valid to doubt whether all these calculations were really necessary. Much of their complexity results from the fact that the theories we used are based on bisimulation semantics, which preserves very many moments of choice, while that was not at all required for the proof we were constructing. Then, the so-called *algebraic advantage*, i.e. the ability to calculate with processes without having to write out the entire state space, is almost absent. This is also disappointing; although the algebraic advantage manifests itself very clearly when verifying protocols that do not exhibit much internal parallelism, such as the *Alternating Bit Protocol* (see for example BAETEN AND WEIJLAND [38]), it seems to be lost in the verification of Fischer’s protocol, which has very much internal parallelism. As a result, we get the worst of both worlds: the state explosion from naive model checking, and the complicated term rewriting from discrete-time process algebra.

Let us however not become too pessimistic. It has become clear that discrete-time process algebra can indeed be applied to protocols with intricate timing aspects. It would be unrealistic to expect that these process algebras, which were designed without much regard for their practical application, would in their unmodified form be splendidly suited for real-life verifications. There is a lot of room for tuning these theories towards more practicality. We see at least three directions in which we would like to proceed, preferably in all three at the same time.

- First of all, it would be nice to have a discrete-time process algebra that does not lean on bisimulation semantics only. It is simply not always a good idea to preserve all internal moments of choice. If we had a theory based on, say, ready semantics or failure semantics, we would probably gain some of our algebraic advantage back.

But maybe abandoning bisimulation semantics altogether would be too crude. A more sophisticated and subtle approach could be to abstract only from those internal moments of choice we really want to abstract from. This could be implemented by introducing a special choice operator next to the ordinary choice. For example, the delayed choice of BAETEN AND MAUW [34], the partial choice of BAETEN AND BERGSTRA [20], or the delayed choice with abstraction of D’ARGENIO AND MAUW [68].

- Secondly, it might be profitable to augment process algebra with a (limited) form of temporal logic. Looking at the linearization process of FP_{drt} , it is clear that much

of the calculations involved are needed to keep track of which time-slice we are in. Or, on a conceptually higher level: the calculations get complicated because we do not have adequate tools for denoting the precise flow of time. When working with a hybrid process algebra-temporal logic theory (still predominantly algebraic!), these complications would probably not have arisen. See for example BAETEN, BERGSTRA, AND BOL [28], where this approach is investigated.

- Thirdly, we might just as well admit that verifications involving time are difficult, and probably will remain so for some years to come. Therefore, it may be advisable to have automated tools at our disposal. One could for example imagine a “process algebra calculator” which, like an ordinary arithmetic calculator, could assist in performing complex calculations. This way we could put the human back into the driver's seat of the verification process, instead of having him stumble in the dark while juggling complex process terms. Assuming we can indeed exploit our algebraic advantage, this approach could be very interesting. A combination of easy process term manipulation and powerful algebraic techniques could lead to verifications that are both short and easy to construct.

Concluding, we can say that the verification of Fischer's Protocol by process algebraic means as given in this chapter is not very straightforward, and not very algebraic. But nonetheless, there are several promising directions for future research that could lead to new, algebraically oriented theories that may perform much better for real-life verifications.

8

Related Work

8.1 Introduction

In this chapter, we examine a number of algebraically oriented timed process formalisms that have been published by others. For each formalism, we give a short indication of its foundations, the nature of the time domain (discrete, dense, unspecified, etc.), the nature of the semantics, and the special features it supports. Whenever the formalism under consideration has interesting connections with the work presented in this thesis, we delve somewhat deeper into it, and discuss the topic with some technical depth. On the whole, however, the emphasis is more on giving a general overview than on discussing technical details.

In choosing which formalisms to discuss, we tried to be quite comprehensive; rather than restricting ourselves to a number of closely related formalisms, we have opted to treat a large number of formalisms, even when that meant we needed to be very concise in some cases. Still, we make no claims as to the completeness of our overview. The field of timed process algebras is so large that we cannot even hope to have covered every formalism described in the literature.

8.2 ACP-Style Timed Process Algebras

In this section, we look at a number of formalisms that find their common roots in the *Algebra of Communicating Processes* (see BERGSTRA AND KLOP [45] and BAETEN AND WEIJLAND [38]).

8.2.1 ACP with Real-Time Steps (Groote)

In GROOTE [87, 88] we find an ACP-style timed process algebra that uses a discrete time domain (the author uses “real time”, somewhat misleadingly, in the sense of “explicit time”, as opposed to the “implicit time” that can be achieved by cleverly encoding time in untimed process algebra). The semantics of the process algebra is given using Plotkin-style rules (see PLOTKIN [166]), leading to a weak-bisimulation model. Among the features of the process algebra are the *empty process*, the *silent action*, and *recursion*.

In these articles, the author introduces a special constant t that is conceptually very much like our constant $\underline{\sigma}$ from Chapter 6. He rejects (on philosophical grounds) the time-factorization principle, which makes the axioms somewhat simpler, and the operational semantics a great deal simpler. The empty process defined is a unit element with respect to the sequential composition but *not* with respect to the merge (he has, for example (in our notation) that $\underline{\varepsilon} \parallel \underline{\sigma} = \underline{\delta}$, and as a result of that, $\underline{a} \parallel \underline{\sigma} \cdot \underline{b} = \underline{a} \cdot \underline{\delta}$).

The interaction between t and the merge is encoded in the communication merge: the author uses (in our notation) $\underline{\sigma} \cdot x \parallel y = \underline{\delta}$, and then needs to use $\underline{\sigma} \cdot x \mid \underline{\sigma} \cdot y = \underline{\sigma} \cdot (x \parallel y)$ to let t propagate through the merge. As a result, he has no free merge, and therefore no PA-like process algebra.

Using recursion, the author defines a process Δ (“delay”), that is both in its behavior and definition very much like our delayable empty process ε .

The author ends with an example of a verification of a *manufacturing workcell*.

8.2.2 Two-Phase Discrete-Time Process Algebra (Baeten and Bergstra)

In BAETEN AND BERGSTRA [21, 24] we find a number of ACP-style timed process algebras that use a discrete time domain. The semantics of the process algebras is given using Plotkin-style rules, leading to a strong-bisimulation model.

These papers have laid the foundation on which the process algebras of Chapters 3 to 7 of this thesis were developed.

The authors define process algebras BPA_{drt} , PA_{drt} , and ACP_{drt} that are very similar to the correspondingly named process algebras in this thesis. Next to a bisimulation model, they also define a graph model and a projective-limit model.

In addition to the relative-time process algebras, as treated in this thesis, the authors also define *absolute-time* process algebras, in which all timing is related to a global clock instead of to the current time slice as is done in *relative time*. We give an example, where, following the authors, we write $cts(a)$ to denote an action a in the current time-slice (i.e., a relative-time undelayable action \underline{a}), and $fts(a)$ to denote an action a in the first time-slice (in the absolute sense). We have operators $\sigma_{rel}(x)$ and $\sigma_{abs}(x)$ that denote the process x in the next time slice, in the relative sense and absolute sense respectively.

First, the relative time example. Here we have the following equality:

$$\sigma_{rel}(cts(a)) \cdot \sigma_{rel}(cts(b)) \cdot \sigma_{rel}(cts(c)) = \sigma_{rel}(cts(a) \cdot \sigma_{rel}(cts(b) \cdot \sigma_{rel}(cts(c))))$$

As can be seen, the σ_{rel} operator behaves in a relative fashion: the a executes in the second time-slice, the b in the third, and the c in the fourth, because the σ_{rel} operators push their arguments one time-slice forward in time relative to the time-slice in which they occur.

Then, the corresponding absolute time example. Here we have this equality:

$$\sigma_{abs}(fts(a)) \cdot \sigma_{abs}(fts(b)) \cdot \sigma_{abs}(fts(c)) = \sigma_{abs}(fts(a) \cdot fts(b) \cdot fts(c))$$

We see that the σ_{abs} operator behaves in an absolute fashion: the a , b , and c all execute in the second time slice, because the σ_{abs} operators push their argument one time-slice forward in time absolutely, i.e., with respect to the very first time-slice, and irrespective to the time-slice in which they occur.

All process algebras treated, like all discrete-time process algebras in this thesis, use *two-phase notation*: they treat actions and time separately. So, when we write $\sigma_{\text{rel}}(\text{cts}(a))$ to denote a in the second relative time-slice, we have distinct notations for time (namely: σ_{rel}) and actions (namely: $\text{cts}(a)$). Similarly, in the structured operational semantics, we treat time transitions separately from action transitions. In Section 8.2.3 we will see an approach in which time and actions are integrated into one single notation: the *timestamped notation*.

The authors end with a method to combine relative timing and absolute timing in one process algebra: this is called *parametric timing*. As they show, the immediate deadlock δ is needed to make this combination work. So, the immediate deadlock finds additional justification for its existence in the combination of relative and absolute timing in one process algebra.

8.2.3 Timestamped Discrete-Time Process Algebra (Baeten and Bergstra)

In BAETEN AND BERGSTRA [25] we find a number of ACP-style timed process algebras that use a discrete time domain. The semantics of the process algebras is given using Plotkin-style rules, leading to a strong-bisimulation model. Process algebras are given that use relative timing, absolute timing, and parametric timing.

The process algebras the authors introduce are similar to the process algebras discussed in Section 8.2.2, except that the actions are now in *timestamped notation* instead of *two-phase notation*. In timestamped notation, time and actions are integrated into one single notation as follows. To denote an action a in the n th time-slice, the authors write $\underline{a}[n]$ in relative timing, and $\underline{a}(n)$ in absolute timing. Relating timestamped notation with two-phase notation, we then conceptually have the following equalities:

$$\begin{aligned}\underline{a}[0] &= \underline{a}(0) = \delta \\ \underline{a}[n+1] &= \sigma_{\text{rel}}^n(\text{cts}(a)) \text{ for } n \in \mathbb{N} \\ \underline{a}(n+1) &= \sigma_{\text{abs}}^n(\text{fts}(a)) \text{ for } n \in \mathbb{N}\end{aligned}$$

Using timestamped notation, we do not need the σ_{rel} and σ_{abs} operators anymore as constructors. This makes process expression much easier to read. We give a few examples of derivable equalities.

First, we look at an “expired” action in absolute time:

$$\begin{aligned}(\underline{a}(1) + \underline{b}(3)) \cdot \underline{c}(2) &= \underline{a}(1) \cdot \underline{c}(2) + \underline{b}(3) \cdot \delta \\ (\underline{a}[1] + \underline{b}[3]) \cdot \underline{c}[2] &= \underline{a}[1] \cdot \underline{c}[2] + \underline{b}[3] \cdot \underline{c}[2]\end{aligned}$$

In the first equality, the c action becomes either enabled in the first time-slice or in the third time-slice. In the latter case, it has already expired, because it needs to execute in the second time-slice. In such a case, the action turns into the immediate deadlock δ . In relative time, expired actions do not occur, as the time-slice in which an action needs to execute is expressed relatively to the current time-slice. So, the right-hand side of the second equality does not contain immediate deadlock.

Next, we look at the behavior of absolute time and relative time with respect to the merge:

$$\begin{aligned}\underline{a}(1) \cdot \underline{b}(3) \parallel \underline{c}(2) \cdot \underline{d}(4) &= \underline{a}(1) \cdot \underline{c}(2) \cdot \underline{b}(3) \cdot \underline{d}(4) \\ \underline{a}[1] \cdot \underline{b}[3] \parallel \underline{c}[2] \cdot \underline{d}[4] &= \underline{a}[1] \cdot \underline{c}[2] \cdot \underline{b}[2] \cdot \underline{d}[3]\end{aligned}$$

In the first equality absolute time is used, and therefore the timestamps on the right-hand side are the same as the time-stamps on the left-hand side. In the second equality, timing is relative, and therefore the timestamps change; the d action should execute in the fourth time-slice after c executes, which is the third time-slice after b executes in the second time-slice after c executes. Therefore, $\underline{d}[4]$ on the left-hand side becomes $\underline{d}[3]$ on the right-hand side.

Finally, we look at an equality involving parametric time:

$$\underline{a}(2) \cdot \underline{b}[3] \cdot \underline{c}(5) \cdot \underline{d}[7] = \underline{a}(2) \cdot \underline{b}(4) \cdot \underline{c}(5) \cdot \underline{d}(11)$$

Here, the relative-time timestamps of the left-hand side have been converted to absolute-time timestamps on the right-hand side.

The authors end with a simple example involving *message passing*, and a discussion regarding applications.

8.2.4 Real-Time Process Algebra (Baeten and Bergstra)

In BAETEN AND BERGSTRA [16] we find a number of ACP-style timed process algebras that use a dense time domain. The semantics of the process algebras is given using Plotkin-style rules, leading to a strong-bisimulation model. A projective limit model is also given.

The authors define several timestamped process algebras, where the timestamps range over the non-negative real numbers $\mathbb{R}^{\geq 0}$. Absolute, relative, and parametric timing are used. To denote the action a at the moment $t \in \mathbb{R}^{\geq 0}$ they write $a(t)$ in absolute timing and $a[t]$ in relative timing. Actions do not take time, but only one action can occupy a given point in time, so the process $a(1) \cdot b(1)$ can only execute an a at time 1, and then deadlocks.

An interesting construct the authors use is called *integration*, which is used to denote an infinite summation over a subset of the time domain. For example:

$$\int_{v \in [3,5)} a(v)$$

denotes the process that executes an a between time 3 (inclusively) and time 5 (exclusively). Since they do not put any restriction on the form of the integration set, the integration construct provides an unusually large expressivity.

The authors end with two simple example specifications: a timed *FIFO Queue*, and a timed version of the *Alternating Bit Protocol*.

8.2.5 Real-Time Process Algebra with Infinitesimals (Baeten and Bergstra)

In BAETEN AND BERGSTRA [22] we find a number of ACP-style timed process algebras that use a dense time domain. The semantics of the process algebras is given using Plotkin-style rules, leading to a strong-bisimulation model.

The process algebras defined by the authors are closely related to the process algebras discussed in Section 8.2.4. The main difference is that this time they use the non-negative *non-standard real numbers* (also called *surreal numbers*, see CONWAY [66] and KNUTH [119]) as their time domain. In this time domain, so-called *infinitesimals* are used, which should intuitively be looked upon as infinitely small numbers. Using this time domain,

actions can execute infinitely close to each other: where in the process algebras of Section 8.2.4 we had that the process $a(1) \cdot b(1)$ could only execute an a at time 1, and then would deadlock, we here have that the process $a(1) \cdot b(1)$ can execute an a at time 1, and then, infinitely close afterward, execute a b , again at time 1. Such actions, that not only take no time, but can also occupy the same point in the time domain, are called *urgent actions* or *immediate actions*.

The authors end by giving translations from the process algebras *Temporal CCS* (see Section 8.3.1), *Wang's Timed CCS* (see Section 8.3.2), *Chen's Timed CCS* (see Section 8.3.3), and the *Algebra of Timed Processes* (see Section 8.6.1) into their process algebras.

8.2.6 Real-Space Process Algebra (Baeten and Bergstra)

In BAETEN AND BERGSTRA [15, 18] we find a number of ACP-style timed process algebras that use a dense time domain. The semantics of the process algebras is given using Plotkin-style rules, leading to a strong-bisimulation model.

The process algebras defined by the authors are, again, closely related to the process algebras discussed in Section 8.2.4. This time they extend their actions not only with a *temporal coordinate*, but also with a *spatial coordinate*. So, in a manner of speaking, all actions are space-time stamped with an element from the space-time domain. This is done as follows: by $a(x, t)$ they denote the action a at the moment t in time (where t ranges over the non-negative real numbers: $t \in \mathbb{R}^{\geq 0}$) at the location x in space (where x is a triple whose components range over the real numbers: $x \in \mathbb{R}^3$). Using this kind of actions, it becomes possible to describe processes that represent objects moving through space.

The authors introduce so-called *multi-actions*, denoted $a_1(x_1) \& \dots \& a_n(x_n)$ to denote the simultaneous execution (in time) of the actions a_1 to a_n at mutually different locations in space. Integration is added, in the same way as discussed in Section 8.2.4, but only over the temporal coordinate, *not* over the spatial coordinate. Further features introduced are a *state operator*, *asynchronous communication*, a *priority operator*, and *process creation*.

In [15] the authors distinguish two different versions of their formalism: one for *classical space-time* (i.e. *Newtonian*), in which relativistic aspects play no rôle, and one for *relativistic space-time* (i.e. *Einsteinian*), in which they do. In [18] they withdraw the relativistic variant, as they feel that the relevance of it is not fully established.

The authors end with a large number of example specifications, among which *communication via a static intermediate station*, a version of the *Concurrent Alternating Bit Protocol*, *communication via a mobile intermediate station*, several versions of the *Positive Acknowledgment with Retransmission Protocol*, *data transmission via a mobile intermediate station using an unreliable medium*, a *message handler*, a *faulty queue for asynchronous message transfer*, and *user interaction with a computer and a printer*.

8.2.7 Real-Time Process Algebra (Klusener)

In KLUSENER [117] we find a number of ACP-style timed process algebras that use a dense time domain. The semantics of the process algebras is given using Plotkin-style rules, leading to several different kinds of bisimulation models.

The author takes the absolute-time process algebras of Section 8.2.4, and studies the approach taken there in more detail. He restricts the format of the integration construct to so-called *prefix integration*, and also places restrictions on the form of the integration set. In this way, it becomes possible to give a complete axiomatization of the model, which is theoretically impossible when unrestricted integration is used.

The author adds *abstraction*, and studies models based on *branching bisimulation*, *delay bisimulation*, and *weak bisimulation*. He also adds *guarded recursion*, and uses this to specify and verify a version of the *Positive Acknowledgment with Retransmission Protocol*. Finally, he introduces *urgent actions*, and studies several models of the resulting process algebras.

The author ends by giving translations from the process algebras *Temporal CCS* (see Section 8.3.1), *Wang's Timed CCS* (see Section 8.3.2), *Chen's Timed CCS* (see Section 8.3.3), the *Timed Calculus* (see Section 8.5.1), the *Algebra of Timed Processes* (see Section 8.6.1), and the *Temporal Process Language* (see Section 8.6.2) into his process algebras.

8.2.8 Algebra of Timed Frames (Bergstra, Fokkink, and Middelburg)

In BERGSTRA, FOKKINK, AND MIDDELBURG [42] we find an ACP-like algebraic theory that use a discrete time domain. The semantics of the algebra follow by considering so-called σ -bisimulation as an equivalence relation on so-called *frames*.

This formalism is different from the other ACP-like formalisms we have encountered in the fact that its central elements are *frames* instead of *processes*. Here, *frames* are structures that are built from *states* and *transitions*. A *frame algebra* is defined that induces an algebraic equivalence between these frames. A special form of bisimulation, called σ -bisimulation, is used to define an operational equivalence between frames.

An interesting detail is that although the authors adhere to the *weak time factorization* principle, this is not reflected in the algebra. Instead, they use their σ -bisimulation to fold σ -transitions onto each other, thus enforcing weak time factorization. As a result, the definition of σ -bisimulation is more complicated than our bisimulation definition are. By doing this, σ -transitions can be treated exactly like action transitions in the axioms, which can therefore be simpler.

8.2.9 Timed μ CRL (Groote)

In GROOTE [89] we find a description of the timed specification language *Timed μ CRL*, that is based on the (untimed) specification language μ CRL (GROOTE AND PONSE [91]). The time domain is left unspecified, and can be chosen at will, provided it is totally ordered and contains a smallest element. An operational semantics is given using Plotkin-style rules, leading to a strong-bisimulation model.

The author introduces the notations $x \cdot t$ (“ x at t ”), to denote the action a at time t , and $x \ll y$ (“ x before y ”), to denote that part of the process x that starts before y must perform an action. These are the only constructions involving time. Only absolute time is used, on the grounds that Timed μ CRL should be a small, clean language. All actions are *urgent*: the process $a \cdot 1 \cdot b \cdot 1$ can execute an a at time 1, and immediately afterward execute a b , still at time 1.

Relevant for this thesis is also the fact that the author gives an example specification of *Fischer's Protocol for Mutual Exclusion* (see Chapter 7). Due to the flexibility of his spec-

ification language, he can leave open the time domain, the number of components, and the execution durations of reading and writing from/to the shared variable. No attempt at verification is made.

8.3 CCS-Style Timed Process Algebras

In this section, we look at a number of formalisms that find their common roots in the *Calculus of Communicating Systems* (see MILNER [142]).

8.3.1 Temporal CCS (Moller and Tofts)

In MOLLER AND TOFTS [145] and MOLLER [144] we find a timed process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model.

The process algebra described, called *Temporal Calculus of Communicating Systems*, or *Temporal CCS* for short, consists of the framework of CCS, with a few simple discrete-time timing constructs added. The result is an elegant, simple process algebra that has a lot in common with the discrete-time process algebras described in Chapters 3 to 5 of this thesis.

The main construct to introduce time is the *temporal prefix operator*, denoted $(t).P$, where $t \in \mathbb{N}^+$. This is the process that idles for t time-slices, and then continues as P , which can be expressed in our notation as $\sigma_{\text{rel}}^t(P)$. Then, there is the *delay operator*, denoted $\delta.P$, the process that can idle for an arbitrary number (including zero) of time-slices, and then continues as the part of P that does not idle. This corresponds exactly to our process $\lfloor P \rfloor^\omega$. An interesting point is that Temporal CCS has two choice operators: the *weak choice*, denoted $P \oplus Q$, that obeys weak time factorization, and the *strong choice*, denoted $P + Q$, that obeys strong time factorization. Finally, there is the *nil process*, denoted $\mathbf{0}$, which can neither do an action, nor let time progress (this corresponds to our undelayable deadlock $\underline{\delta}$).

In Temporal CCS, all actions are undelayable: the process $a.P$ (in our notation: $\underline{a}.P$) must execute an a in the first time-slice. A delayable action can be coded as $\delta.a.P$ (in our notation: $\lfloor \underline{a}.P \rfloor^\omega$). As a shorthand for $\delta.a.P$ and $\delta.\mathbf{0}$, the notations $\underline{a}.P$ and $\underline{\mathbf{0}}$ are introduced, just like we introduced the delayable action a and the delayable deadlock δ .

The authors examine a subcalculus of Temporal CCS, called *Loose Temporal CCS*, or ℓ TCCS for short, in which all actions and deadlocks are made delayable. In ℓ TCCS, the weak choice and the strong choice collapse into the same operation, and there is no longer need for the delay operator.

Finally, a parallel composition operator is defined and axiomatized. Since no auxiliary operators, like for the example a left merge, are introduced, the axiomatization is both complex and infinite.

8.3.2 Timed CCS (Wang)

In WANG [201, 202, 203] we find a timed process algebra that uses an unspecified time domain, although in all examples the non-negative real numbers $\mathbb{R}^{\geq 0}$ are used. The se-

mantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model and a weak-bisimulation model.

The process algebra described is called *Timed Calculus of Communicating Systems*, or *Wang's Timed CCS* for short. The process algebra features two main constructs to introduce time. The first, called *time event prefix*, and denoted $\epsilon(d).P$, is the process that idles for an amount $d > 0$ of time, and then becomes P . The second, called *generalized action prefix*, and denoted $\mu@t.P$ (where t is a time variable that may be free in P), is the process that idles for an indeterminate amount d of time, executes the action μ , and then becomes $P[d/t]$ (i.e., P where d is substituted for t).

ACETO AND JEFFREY [6] give a complete axiomatization for open terms with finite-state recursion in Wang's Timed CCS, and in WANG [204], probabilities are introduced in the context of Wang's Timed CCS.

8.3.3 Timed CCS (Chen)

In CHEN [62] we find a timed process algebra that uses an unspecified time domain. An operational semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model and a weak-bisimulation model. Also a denotational semantics is given, based on *timed synchronization trees*.

The process algebra described is called *Timed Calculus of Communicating Systems*, or *Chen's Timed CCS* for short. The main construct to introduce time is the *action prefix operator*, denoted $\alpha(t)_e^{e'}.P$, where e and e' are *time expressions*, and t is a time variable that may be free in P . The process $\alpha(t)_e^{e'}.P$ is the process that idles for an amount d of time, where $0 \leq e \leq d \leq e' \leq \infty$, executes the action α , and then becomes $P[d/t]$ (i.e., P where d is substituted for t).

The author examines, among other things, decidability and completeness issues of the process algebras that result when the initially unspecified time domain is instantiated with a number of discrete and dense time domains.

8.3.4 Timed Probabilistic CCS (Hansson)

In HANSSON [92, 93] we find a timed process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model.

The process algebra described is called *Timed Probabilistic Calculus of Communicating Systems*, or *TPCCS* for short. The main construct to introduce time is the *binary time-out operator*, denoted $N \triangleright P$ (where N and P are TPCCS processes), which can be expressed as $\nu_{\text{rel}}(N) + \sigma_{\text{rel}}(P)$ in our notation. Note that this is the same operator as the $[N](P)$ operator of ATP (see Section 8.6.1).

The interesting feature of TPCCS is the presence of *probabilities*: next to a non-deterministic choice, there is also a probabilistic choice in which every summand has a weight associated with it, that represents the probability of that summand being chosen.

In his dissertation [93], the author also introduces a temporal probabilistic logic, which is shown to possess an expressive power that is equivalent to bisimulation. Finally, a large number of examples is specified and verified in extensive detail.

8.3.5 Linear Timed CCS (Jeffrey)

In JEFFREY [111] we find a timed process algebra that uses an arbitrary totally ordered monoid as its time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model.

The process algebra defined, called *Linear Timed Calculus of Communicating Systems*, or *LTCCS* for short, is based on an abstract time domain, for which any totally ordered monoid can be substituted. As the author shows, many (timed) process algebras from the literature can be mimicked by LTCCS, by properly instantiating the time domain, and building the appropriate operators from the the primitives of LTCCS. Examples include TPL (see Section 8.6.2), by taking $(\mathbb{N}, +, 0)$ as the monoid, Wang's Timed CCS (see Section 8.3.2), by taking $(\mathbb{R}^{\geq 0}, +, 0)$ as the monoid, and untimed CCS, by taking $(\{0\}, +, 0)$ as the monoid.

The main construct to introduce time is the $\varepsilon t:P$ operator (where t is an element of the time domain, and P an LTCCS process), which idles for an amount t of time, and then behaves as P .

8.3.6 CCS with Interval Time (Daniels)

In DANIELS [67] we find a timed process algebra that uses a dense time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model, a timed-weak bisimulation model, and a weak bisimulation model.

The process algebra defined, called *Calculus of Communicating Systems with Interval Time*, or *CCSiT* for short, uses the real numbers \mathbb{R} as its time domain. The main construct to introduce time is the *prefix operator*, denoted $\mu@i.P$ (where μ is an action, i an open or closed interval from \mathbb{R} , and P a CCSiT process), which is the process that can execute a μ at any moment $t \in i$, and then behaves as P .

Note the similarity between the $\mu@i.P$ operator of CCSiT and the more general integration operator of Real-Time Process Algebra (see Section 8.2.4). In terms of this integration operator, $\mu@i.P$ can be expressed as $(\int_{t \in i} \mu[t]) \cdot P$.

8.3.7 Constraint-Oriented Real-Time Process Calculus (Fidge)

In FIDGE [78] we find a timed process algebra that can either use a discrete or a dense time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a bisimulation model.

The process algebra defined, called *Constraint-Oriented Real-Time Process Calculus*, is based on untimed CCS, using time extensions in the style of *TIC* (see Section 8.5.1). As the time domain, either the natural numbers \mathbb{N} , or the non-negative real numbers $\mathbb{R}^{\geq 0}$ can be used. The main constructs to introduce time are the *prefix operator*, denoted $\alpha^r.E$ (where α is an action, r an element of the time domain, and E a process), which is the process that idles for an amount r of time, executes an action α , and then behaves as E , and a generalization of this operator, denoted $\alpha^R.E$ (where α is an action, R a subset of the time domain, and E a process), which is the process that idles for a non-deterministic amount $r \in R$ of time, executes an action α , and then behaves as E .

8.4 CSP-Style Timed Process Algebras

In this section, we look at a number of formalisms that find their common roots in *Communicating Sequential Processes* (see HOARE [98]).

8.4.1 Timed CSP (Reed and Roscoe)

In REED AND ROSCOE [175, 176] we find a timed process algebra that uses a dense time domain. A denotational semantics of the process algebra is given, leading to the *timed stability model*.

The process algebra described, called *Timed Communicating Sequential Processes*, or *Timed CSP* for short, uses the non-negative real numbers $\mathbb{R}^{\geq 0}$ as its time domain. The main construct to introduce time is denoted $\text{WAIT } t$ (where $t \in \mathbb{R}^{\geq 0}$), the process that idles for an amount t of time, and then terminates.

Over the years, Timed CSP has undergone a lot of changes. For a comprehensive introduction into Timed CSP, and an overview of its history, see REED, ROSCOE, AND SCHNEIDER [177], DAVIES, *et al.* [70], or DAVIES AND SCHNEIDER [73]. Here, we suffice by mentioning a number of relevant papers.

First, in the dissertation of JONES [113] we find an early attempt at providing CSP with a timed model. In HUIZING, GERTH, AND DE ROEVER [100] and GERTH AND BOUCHER [82] a denotational semantics of Timed CSP is developed, leading to a timed failures model. Then, in the dissertation of REED [174], a variety of semantic models for Timed CSP is given, and in the dissertations of SCHNEIDER [185] and DAVIES [74], we find proof theory for Timed CSP. An operational semantics of Timed CSP is given in SCHNEIDER [186], and LOWE [133, 134] adds probabilities and priorities to Timed CSP. A large number of case studies can be found in DAVIES, *et al.* [70], and DAVIES AND SCHNEIDER [72] show how to verify the *Alternating Bit Protocol* using Timed CSP.

8.4.2 Discrete Timed CSP (Jeffrey)

In JEFFREY [109] we find a timed process algebra that uses a discrete time domain. A denotational semantics of the process algebra is given, using *timed traces*, *timed refusals*, *timed failures*, and *metric spaces*.

The process algebra described is called *Discrete Timed Communicating Sequential Processes*, or DCSP for short. There is no construct to introduce time explicitly. Instead, time is implicitly introduced by the rule that if the environment refuses to cooperate in any action that is offered to it, then the remaining part of the process moves on to the next time-slice. By this rule, the process that offers no action at all and then continues as P , is equivalent to P in the next time-slice. Using this coding trick, the author recursively defines the $\text{WAIT } t$ construct of Timed CSP (see Section 8.4.1), such that for any $t \in \mathbb{N}$, the process $\text{WAIT } t$ is the process that idles for t time-slices, and then terminates.

8.5 LOTOS-Style Timed Process Algebras

In this section, we look at a number of formalisms that find their common roots in LOTOS (see ISO [101]).

8.5.1 Timed Calculus (Quemada, de Frutos, and Azcorra)

In QUEMADA AND FERNANDEZ [171], QUEMADA, AZCORRA, AND DE FRUTOS [170], and QUEMADA, DE FRUTOS, AND AZCORRA [172] we find a process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model and a weak-bisimulation model.

The process algebra described is called *Timed Calculus*, or *TIC* for short. The main constructs to introduce time are the *action prefix operator*, denoted $at;B$ (where $t \in \mathbb{N}$, and B is a TIC process), which is the process that executes an a in the $(t + 1)$ th time-slice, and then behaves as B , and an extension of it, the *timed choice operator*, denoted $aT;B$ (where T is an interval from \mathbb{N} , and B is a TIC process), which is the process that for any $t \in T$ can execute an a in the $(t + 1)$ th time-slice, and then behave as B .

In [172] the authors give two expansion theorems, consider internal actions, and give two examples: a *stop and wait protocol* and a *railroad crossing*.

8.5.2 U-LOTOS / T-LOTOS (Bolognesi and Lucidi)

In BOLOGNESI AND LUCIDI [50, 51] we find two process algebras that use a discrete time domain. The semantics of the process algebras is given using Plotkin-style rules.

The two process algebras described, called *U-LOTOS* and *T-LOTOS*, are inspired by ideas from *Temporal CCS* (see Section 8.3.1). The first one, *U-LOTOS*, has two main constructs to introduce time: the *time prefix operator*, denoted $(t).B$ (where $t \in \mathbb{N}$, and B is a U-LOTOS process), which is the process that idles for t time-slices, and then behaves as B , and the *urgency of actions operator*, denoted $\text{asap } G \text{ in } B$ (where G is a set of *gates*, and B is a U-LOTOS process), which is the process that behaves like B , except that as soon as an action at some gate in the set G is ready for execution, it is immediately executed. The second process algebra, *T-LOTOS*, generalizes the *urgency of actions operator* to a *timer operator*, denoted $\text{timer } a(t_1, t_2) \text{ in } B$ (where $t_1, t_2 \in \mathbb{N}$, and B is a T-LOTOS process), which is the process that behaves like B , except that as soon as the action a is enabled, it must be executed in the closed interval $[t_1, t_2]$ of future time-slices. So, the U-LOTOS process $\text{asap } a \text{ in } B$ is equivalent to the T-LOTOS process $\text{timer } a(0, 0) \text{ in } B$.

8.5.3 TLOTOS (Leduc)

In LEDUC [128] we find a process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model.

The process algebra described, called *TLOTOS*, is inspired by the *Algebra of Timed Processes* (see Section 8.6.1). Let P and Q be TLOTOS processes, and $d \in \mathbb{N}^+$, then the main constructs to introduce time are the *start delay operator*, denoted $[P]^d(Q)$, the *execution delay operator*, denoted $[P]^d(Q)$, and the *unbounded start delay*, denoted $[P]^\omega$. For each of these three operators, the semantics is identical to that of the corresponding operator in ATP (see Section 8.6.1 for a full description).

8.5.4 Timed LOTOS / ET-LOTOS (Leduc and Léonard)

In LEDUC AND LÉONARD [129] and LÉONARD AND LEDUC [131] we find two process algebras that use an unspecified time domain. The semantics of the process algebras is given using Plotkin-style rules.

The process algebra described in [129], is called *Timed LOTOS*. The main construct to introduce time is the *delay operator*, denoted $\Delta^{[d_1, d_2]}P$ (where d_1 and d_2 are elements of the time domain, and P is a Timed LOTOS process), which is the process that idles for a non-deterministic amount $t \in [d_1, d_2]$ of time, and then behaves as P . On the basis of this operator, a number of other timing operators are introduced.

In [131], the authors extend Timed LOTOS with several new features, leading to a new process algebra called *Enhanced Timed LOTOS*, or *ET-LOTOS* for short. The main new construct is the *extended action prefix operator*, denoted $g @ t \{d\}; A$ (where g is an action, t is a variable that may be free in A , d is an element from the time domain, and A is an ET-LOTOS process), which is the process that offers an action g for a period d of time; if in this period the g is executed after an amount t' of time, then after that, the process behaves as $A[t'/t]$ (i.e., A where t' is substituted for t), or else, the process becomes a delayable deadlock. The authors end by giving a case study of the use of ET-LOTOS.

Finally, in BRYANS, DAVIES, AND SCHNEIDER [57], a denotational semantics for ET-LOTOS is developed.

8.5.5 TE-LOTOS (Davies, Bryans, and Schneider)

In DAVIES, BRYANS, AND SCHNEIDER [71] we find a timed process algebra that uses a dense time domain. A denotational semantics of the process algebra is given, leading to the *timed observations model*.

The process algebra described is called *Timed Enhanced LOTOS*, or *TE-LOTOS* for short. The main constructs to introduce time are the *action prefix operator*, denoted $a\{t \text{ in } d^-.d^+\}; P$ (where a is an action, $d^-, d^+ \in \mathbb{R}^{\geq 0}$, and P is a TE-LOTOS process), which is the process that can execute an action a after any amount $t \in [d^-, d^+]$ of idling, and then behave as P , or else becomes a delayable deadlock, and the *delay operator*, denoted $\text{Wait}(d); P$ (where $d \in \mathbb{R}^{\geq 0}$, and P is a TE-LOTOS process), which is the process that idles for an amount d of time, and after that behaves as P .

8.5.6 LOTOS-T (Miguel, Fernández, and Vidaller)

In MIGUEL, FERNÁNDEZ, AND VIDALLER [140] we find a process algebra that uses an unspecified time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model and a weak-bisimulation model.

The process algebra described is called *LOTOS-T*. The main constructs to introduce time are the *timed action prefix*, denoted $a\{t\}; B$ (where a is an action, t is an element from the time domain, and B is a LOTOS-T process), which is the process that idles for an amount t of time, and then can execute an action a , or else becomes a delayable deadlock, and the *timed successful termination*, denoted $\text{exit}\{t\}$ (where t is an element from the time domain), which is the process that idles for an amount t of time, and then can successfully terminate, or else becomes a delayable deadlock.

8.5.7 LOTOS/T (Nakata, Higashino, and Taniguchi)

In NAKATA, HIGASHINO, AND TANIGUCHI [151] we find a process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model and a weak-bisimulation model.

The process algebra described is called *LOTOS/T*. The main construct to introduce time is the *time-constrained action prefix*, denoted $a[P(t, \bar{x})]; E$, where a is an action, t is a special variable that denotes the absolute time, \bar{x} denotes the vector of all variables except t , $P(t, \bar{x})$ is a predicate over a certain subset of the Presburger Arithmetic, and E is a LOTOS/T process. The meaning of this construct is a process that can under certain conditions execute an action a at a certain moment in time, and then behaves as E . For a full definition of the predicates used, and their semantics, see [151]. Here, we only repeat a simple example given by the authors, in order to sketch the flavor of the formalism. Consider the following process, in which t is the special variable that denotes the absolute time:

$$a[2 \leq t \leq 3 \wedge x_0 = t]; b[t = x_0 + 3]; \text{stop}$$

This is the process that executes the action a between time 2 and 3, and then executes the action b three time-units later. The assignment $x_0 = t$ is used to record the time of execution of a , and then later on x_0 is referred to in the second predicate in order to specify that the b should execute 3 time-units after the a .

8.6 Other Timed Process Algebras

In this section, we look at a number of formalisms that cannot be easily classified as being timed extensions of existing untimed formalisms.

8.6.1 Algebra of Timed Processes (Nicollin and Sifakis)

In NICOLLIN AND SIFAKIS [154] we find a timed process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model.

The process algebra described, called *Algebra of Timed Processes*, or *ATP* for short, is closely related to ACP: ATP borrows several features from untimed ACP, and discrete-time ACP, in turn, borrows from ATP. Since ATP is so closely related to the process algebras described in this thesis, we will examine it in some detail.

We begin with the signature. ATP has the following constants and operators (let P and Q be ATP processes, α an action, and X a free variable):

- The *deadlock constant*, denoted 0 , which is our $\underline{\delta}$.
- A countable set of *actions*, denoted \mathcal{A} . There is one special action, denoted χ , that represents the progress of time. The set of *asynchronous actions*, i.e. $\mathcal{A} - \{\chi\}$, is denoted by \mathcal{A}^α . A single action, by itself, does *not* denote a process in ATP. In our terminology, \mathcal{A} would be called the *alphabet* instead of the set of actions (see also Remark 2.3.1.4 on page 7).

- The *prefixing operator*, denoted αP , for $\alpha \in \mathcal{A}^\alpha$, which can be expressed as $\underline{\alpha} \cdot P$ in our notation. In our terminology, all actions in ATP are undelayable. Like, e.g., CCS, ATP has no general sequential composition, but only action prefix. As the deadlock, 0, functions as a starting point to build processes, all processes end in deadlock. So, no distinction is made between successful and unsuccessful termination, as is done in ACP.
- The *alternative choice operator*, denoted $P \oplus Q$, which is almost like our $P + Q$, except that ATP uses *strong* time factorization.
- The *unit-delay operator*, denoted $[P](Q)$, which can be expressed as $\nu_{\text{rel}}(P) + \sigma_{\text{rel}}(Q)$ in our notation. ATP has no operator corresponding to our $\sigma_{\text{rel}}(x)$, however, this process can be expressed as $[0](x)$ in ATP. Due to the use of strong time factorization, ATP can express our process $\nu_{\text{rel}}(x)$ as $0 \oplus x$. Note that 0 is not a proper zero element for the alternative choice operator in ATP.
- The *recursion operator*, denoted $\text{rec } X \cdot P$. As we have no recursion, we will not look at ATP recursion.
- The *parallel composition operator*, denoted $P \parallel Q$, which is the ACP merge operator.
- The *left merge operator*, denoted $P \ll Q$, which is the ACP left merge operator.
- The *communication merge operator*, denoted $P | Q$, which is the ACP communication merge operator.
- The *encapsulation operator*, denoted $\partial_H(P)$, which is the ACP encapsulation operator.

In connection with Chapter 5, it is interesting to see how the merge operators are axiomatized. In Table 8.1 the ATP axioms for the merge and left merge operators are shown (where P, Q, R, P_1, P_2, Q_1 , and Q_2 are ATP processes, and “ \equiv ” is used instead of “ $=$ ”). Then, in Table 8.2 on the next page, the same axioms interpreted in ACP notation are shown, *where the alternative composition should be viewed as a strong time-factorization alternative composition*.

$$\begin{array}{rcl}
 P \parallel Q \equiv P \ll Q \oplus Q \ll P \oplus P | Q & [\parallel 1] \\
 (P \oplus Q) \ll R \equiv P \ll R \oplus Q \ll R & [\ll 1] \\
 0 \ll P \equiv 0 & [\ll 2] \\
 \alpha P \ll Q \equiv \alpha(P \parallel Q) & [\ll 3] \\
 [P](Q) \ll (0 \oplus R) \equiv P \ll (0 \oplus R) & [\ll 4] \\
 [P_1](Q_1) \ll [P_2](Q_2) \equiv [P_1 \ll P_2](Q_1 \ll Q_2) & [\ll 5]
 \end{array}$$

Table 8.1: Axioms for the ATP merge and left merge operators.

$x \parallel y = x \ll y + y \ll x + x \mid y$	ATPCM1
$(x + y) \ll z = x \ll z + y \ll z$	ATPLM1
$\underline{\underline{\delta}} \ll x = \underline{\underline{\delta}}$	ATPLM2
$\underline{\underline{a}} \cdot x \ll y = \underline{\underline{a}} \cdot (x \parallel y)$	ATPLM3
$(\nu(x) + \sigma(y)) \ll \nu(z) = x \ll \nu(z)$	ATPLM4
$(\nu(x_1) + \sigma(y_1)) \ll (\nu(x_2) + \sigma(y_2)) = \nu(x_1 \ll x_2) + \sigma(y_1 \ll y_2)$	ATPLM5

Table 8.2: Axioms for the ATP merge and left merge operators (interpreted in ACP notation, assuming strong time factorization).

When we study these axioms more closely, we see that Axioms ATPCM1, ATPLM1, and ATPLM3 are identical to our Axioms DRTCM1, DRTM4, and DRTM3 respectively, and furthermore, that Axiom ATPLM2 is an instantiation of our Axiom DRTM2.

The remaining axioms, Axioms ATPLM4 and ATPLM5, bear a conceptual correspondence to our Axioms DRTM5 and Axioms DRTM6 (see Table 5.2 on page 106), which can be viewed as follows. Both axiomatizations are based on splitting the right argument of the left merge into a ν_{rel} -part and a σ_{rel} -part. This leads in both axiomatizations to the distinction of two cases: the case in which the right argument does not have a σ_{rel} -part (their Axiom ATPLM4, our Axiom DRTM5), and the case in which the argument does have a σ_{rel} -part (their Axiom ATPLM5, our Axiom DRTM6). The difference between their axioms and our axioms is in the left argument of the left merge. They split this argument also into a ν_{rel} -part and a σ_{rel} -part, while we assume it has only a σ_{rel} -part, and leave the handling of a possible ν_{rel} -part to a distributivity axiom, namely Axiom DRTCM4, in the assurance that ν_{rel} is an operator that can be eliminated on basic terms. The motivation for these choices is clear: in ATP, processes are by nature split into $\nu_{\text{rel}}/\sigma_{\text{rel}}$ parts, due to the use of the $[P](Q)$ operator to construct processes that can do a time step; in ACP, we have a $\sigma_{\text{rel}}(x)$ operator to construct such processes, so we can have somewhat simpler axioms.

Note that Axiom ATPLM5 is *not* a valid equality in our process algebras, say in PA_{drt}^- -ID. This is due to the use of strong time factorization in ATP, and weak time factorization in ACP. Examine the following two examples. In strong time factorization, using Axiom ATPLM5, we have this equality:

$$(\nu(\underline{\underline{a}}) + \sigma(\underline{\underline{b}})) \ll (\nu(\underline{\underline{c}}) + \sigma(\underline{\underline{d}})) = \underline{\underline{a}} \cdot \underline{\underline{c}} + \sigma(\underline{\underline{b}} \cdot \underline{\underline{d}})$$

whereas in weak time factorization we have the following equality:

$$(\nu(\underline{\underline{a}}) + \sigma(\underline{\underline{b}})) \ll (\nu(\underline{\underline{c}}) + \sigma(\underline{\underline{d}})) = \underline{\underline{a}} \cdot (\underline{\underline{c}} + \sigma(\underline{\underline{d}})) + \sigma(\underline{\underline{b}} \cdot \underline{\underline{d}})$$

As can be seen, in weak time factorization the subterm $\sigma(\underline{\underline{d}})$ becomes enabled after the a has been executed, while in strong time factorization the $\sigma(\underline{\underline{d}})$ disappears, because the subterm $\underline{\underline{c}}$, which is in a choice context with $\sigma(\underline{\underline{d}})$, cannot do a time step.

Next to the ATP operators already discussed, the authors also introduce a number of auxiliary operators, called *delay operators*, all of which can be eliminated. Most of these operators are parameterized by a number $d \in \mathbb{N}^+$, where d represents the number of time-slices before something special happens. We describe them shortly (let P and Q be ATP processes):

- The *timeout at d operator*, denoted $P \triangleright^d Q$, is the process that waits for P to execute an action in any of the first d time-slices, and then continues as P , or else becomes Q in the $(d + 1)$ th time-slice.
- The *start-delay within d operator*, denoted $\lfloor P \rfloor^d(Q)$, is the process that can start executing P in any of the first d time-slices, or else becomes Q in the $(d + 1)$ th time-slice. The difference with the previous operator is that the start-delay operator can delay the execution of P , even if P cannot delay its own execution, while the timeout operator cannot delay the execution of P if P cannot delay its own execution. For $d = 1$, the start-delay operator is identical to the unit-delay operator: $\lfloor P \rfloor^1(Q) = \lfloor P \rfloor(Q)$.
- The *unbounded start-delay operator*, denoted $\lfloor P \rfloor^\omega$, is the process that can delay the execution of the first action of P for an arbitrary number of time-slices. This operator is like our $\lfloor x \rfloor^\omega$ operator, but differs from it at a crucial point: while in ACP we have that $\lfloor \sigma_{\text{rel}}(x) \rfloor^\omega = \delta$, in ATP we have that $\lfloor \sigma_{\text{rel}}(x) \rfloor^\omega = \sigma_{\text{rel}}(\lfloor x \rfloor^\omega)$. The ATP unbounded start-delay operator is not identical to the iterated delay $\sigma_{\text{rel}}^*(x)$ of ACP (see Remark 3.2.4.4 on page 55) either: for that operator, we have that $\sigma_{\text{rel}}^*(\sigma_{\text{rel}}(x)) = \sigma_{\text{rel}}(\sigma_{\text{rel}}^*(x + \sigma_{\text{rel}}(x)))$.
- The *execution delay within d operator*, denoted $\lceil P \rceil^d(Q)$, is the process that behaves as P for the first d time-slices, and then continues as Q .

The authors end with a comparison of the Algebra of Timed Process with the process algebras *ACP with Real-Time Steps* (see Section 8.2.1), the *Real-Time Process Algebra* (see Section 8.2.4), *Temporal CCS* (see Section 8.3.1), *Wang's Timed CCS* (see Section 8.3.2), *Timed CSP* (see Section 8.4.1), and the *Temporal Process Language* (see Section 8.6.2).

8.6.2 Temporal Process Language (Hennessy and Regan)

In HENNESSY AND REGAN [95, 96] we find a timed process algebra that uses a discrete time domain. An operational semantics of the process algebra using Plotkin-style rules, and a semantic theory based on testing are given.

The process algebra described, called *Temporal Process Language*, or *TPL* for short, is related to ATP (see Section 8.6.1), the main difference being, in our terminology, that in TPL all actions are delayable, whereas in ATP all actions are undelayable. The elementary timing construct of TPL in [95] is action prefix with a special action σ denoting time: $\sigma.t$, the process that waits for one time-slice, and then becomes t (it is from this notation that our $\sigma_{\text{rel}}(x)$ derives). Later, in [96], the $\lfloor p \rfloor(q)$ operator of ATP was added, in order to facilitate the axiomatization of the parallel composition. The $\sigma.t$ operator remained, but is now little more than syntactic sugar: $\sigma.t$ can be expressed by $\lfloor \text{nil} \rfloor(t)$ (where nil denotes a delayable deadlock) in the new algebra. Like in ATP, the choice operator of

TPL uses strong time factorization. Another important concept used in TPL, is the notion of *maximal progress*, which here means that processes that *can* communicate, *must* communicate. So, if there is still communication possible, no clock ticks can occur.

The authors end with a specification and verification of a simple protocol, the *Security Costs Protocol*.

8.6.3 Interval Process Algebra (Murphy)

In MURPHY [149, 150] and ACETO AND MURPHY [7, 8] we find a number of timed process algebras that use a dense time domain.

The process algebras described all derive from the *Interval Process Algebra*, or *IPA* for short, that was described in the dissertation of MURPHY [150]. IPA is a non-interleaving process algebra that has roots in both (untimed) CCS and Timed CSP. An interesting feature is that, unlike in most other timed process algebras we have encountered, actions in IPA have a non-zero duration associated with them. A function Δ , that is a parameter of the theory, associates with all actions a number $t \in \mathbb{R}$, $t > 0$, that signifies the duration of that action. Thus, the execution of a certain action at a certain moment in time marks the beginning of an interval in time during which that action is being executed.

In [150], we find an axiomatic semantics, a denotational semantics, and an operational semantics. In [149], a slightly different operational semantics is given. In [7, 8], a subset of IPA is given, called cIPA, together with an operational semantics using Plotkin-style rules, and a corresponding bisimulation model.

8.6.4 Calculus of Timed Refinement (Čerāns)

In ČERĀNS [60] we find a timed process algebra that uses a dense time domain. An operational semantics and a specification refinement semantics are given.

The author defines a process algebra called *Calculus of Timed Refinement*, or *CTR* for short, that is related to CCS. As the time domain $\mathbb{R}^{\geq 0}$, the set of non-negative real numbers, is used. The main construct to introduce time is the *time interval prefix*, denoted $[a, b].P$, with $0 \leq a \leq b \leq \infty$, which is the process that idles for an amount $d \in [a, b]$ of time, and then turns into the process P .

8.6.5 Algebra of Communicating Shared Resources (Brémond-Grégoire *et al.*)

In BRÉMOND-GRÉGOIRE, LEE, AND GERBER [54] we find a timed process algebra that uses a dense time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model.

The authors define a process algebra called *Algebra of Communicating Shared Resources*, or *ACSR* for short. As time domain, the positive real numbers are used. The authors distinguish two types of actions: those that consume time, and those which are instantaneous. The first kind represents the progress of time, the second kind provides a mechanism for synchronization and communication between actions. The main construct to introduce time is denoted $(t, A) : P$, which is the process that executes the action A for an amount t of time, and then proceeds as P .

8.6.6 Abstract-Time Process Algebra (Jeffrey)

In JEFFREY [108, 112] we find a timed process algebra that uses an abstract, partially ordered time domain. An operational semantics and a denotational semantics are given.

The author defines a process algebra called *Abstract-time Process Algebra*, or *APA* for short, that has roots in both CCS and CSP. The notion of a *timed observation* is defined, and subsequently used to introduce an abstract notion of timing in the process algebra.

8.6.7 Process Algebra with Multiple Clocks (Andersen and Mendler)

In ANDERSEN AND MENDLER [10] we find a process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a weak-bisimulation model.

The authors define a process algebra called *Process Algebra with Multiple Clocks*, or *PMC* for short, that is related to ATP (see Section 8.6.1). Instead of using a fixed, measurable, and global notion of time, as do most of the timed process algebras described in this chapter, PMC uses the notion of uninterpreted multiple clocks, that enforce broadcast communication between processes.

8.6.8 Calculus for Synchrony and Asynchrony (Cleaveland *et al.*)

In CLEAVELAND, LÜTTGEN, AND MENDLER [64] we find a process algebra that uses a discrete time domain. The semantics of the process algebra is given using Plotkin-style rules, leading to a strong-bisimulation model and a weak-bisimulation model.

The authors define a process algebra called *Calculus for Synchrony and Asynchrony*, or *CSA* for short, that is related to both TPL (see Section 8.6.1), and PMC (see Section 8.6.7). CSA uses multiple clocks, as does PMC, and combines it with the maximal progress assumption of TPL, but keeps the maximal progress assumption local to each clock. This locality restriction is called *clock scoping*.

8.6.9 Real Time Agents (Cardelli)

In CARDELLI [59] we find a process algebra that uses a dense time domain. An operational semantics of the process algebra is given, leading to a strong-bisimulation model.

Actions in this process algebra have a duration, like they do in IPA (see Section 8.6.3). The main construct used to introduce time is denoted $a[t]$, which is the process that performs the action a for an amount t of time. A related construct is denoted $a(t)$, which is the process that performs the action a for *at most* an amount t of time.

On a historical note: [59] appears to be one of the very first papers ever published in the field of timed process algebras.

8.6.10 Discrete-Time TOOLBUS (Bergstra and Klint)

In BERGSTRA AND KLINT [43, 44] we find a description of the *Discrete-Time TOOLBUS* system. This is not a process algebra, but rather a piece of software that is based on the timestamped discrete-time process algebras of BAETEN AND BERGSTRA [25] (see Section

8.2.3) which in turn are related to the process algebras described in Chapters 3 to 7 of this thesis. As such, the Discrete-Time TOOLBUS system has our interest.

The Discrete-Time TOOLBUS system was designed to facilitate the interconnection of a number of software components (called *tools*) in a modular, flexible, and dynamic way. Each of these tools may have a different user interface, be developed using a different programming languages, and run on a different computing platform. Using the TOOLBUS system, such a heterogeneous whole of tools can be integrated into one distributed system, in which all tools are required to communicate with each other through the TOOLBUS system. The specifics of these communications are implemented by a so-called *T script*, which can be viewed as an executable discrete-time process algebra specification. We will now look at this discrete-time process algebra; for more information about the TOOLBUS system in general, we refer to [43, 44].

Like the timestamped process algebras of Section 8.2.3, the process algebra used in the TOOLBUS system uses the natural numbers, \mathbb{N} , to enumerate an infinite number of time-slices. Timestamped actions are denoted $a^\vee(n)$, which should be interpreted as the absolute-time process that can execute an a in the $(n + 1)$ th time-slice, or, alternatively, idle forever. In the terminology of Section 8.2.3, we have:

$$a^\vee(n) = \underline{a}(n) + \delta$$

where δ denotes the delayable deadlock process. Using *time-spectrum abstraction*, the relative time counterpart of $a^\vee(n)$ is introduced: the process $a^\vee[n]$, which should be interpreted as the relative-time process that can execute an a in the $(n + 1)$ th time-slice after it is initialized, or, alternatively, idle forever. Absolute timing and relative timing may be freely mixed; the process algebra supports parametric timing.

Among the operators of the process algebra we find, next to the usual ACP operators, a *renaming operator*, a *process creation operator*, a *state operator*, an *iteration operator*, and a *condition operator*.

The authors define the ASCII syntax of T scripts, which are based on their process algebra, and extend T scripts with a number of communication primitives and facilities to handle data. Finally, two examples are given: a *calculator*, and a *distributed auction*.

8.7 Further Reading

A good starting point for more information about concurrency and process algebras, in both untimed and timed form, can be found in the proceedings of the *International Conference on Concurrency Theory*, or *CONCUR* for short, which is being held annually since 1990. As of this writing, the proceedings of 1990–1997 have appeared: [32, 33, 46, 65, 114, 130, 139, 146].

In NICOLLIN AND SIFAKIS [153] an overview and synthesis of timed process algebras is given, which includes an overview of useful properties and features in timed process algebras, and a section on “*How to cook your own timed process algebra*”.

Other interesting papers are JEFFREY [110], which contains a general discussion about timed process algebras; GODSKESEN AND LARSEN [85], which examines *expansion theorems* for timed process algebras; BRIM [55], which shows how to add modal logic to a timed process algebra; BOLOGNESI, LUCIDI, AND TRIGILA [52], which contains a general

discussion on timed extensions of LOTOS; and BUN [58], in which a comparison is made between the ACP and ATP approaches to timed process algebra.

Most of the papers mentioned in this chapter have a section in which related work is discussed. One of the most extensive discussions of these can be found in the dissertation of HANSSON [93].

9

Conclusions

9.1 Discrete-Time Process Algebra

We have argued that untimed process algebra is not suitable for the description and analysis of a certain class of protocols, namely those protocols whose correctness critically depends on timing aspects, and those protocols whose timing aspects we want to study quantitatively.

See Section 3.1

We have furthermore argued that it is a permissible simplification to look at time as if it were subdivided in a countably infinite number of time-slices, i.e., as if time proceeded discretely. Although this is less general than using a dense time domain, it leads to considerable simplification of the theory.

See Section 3.1

We have defined a number of discrete-time basic process algebras with relative timing. These process algebras are minor modifications of process algebras that were already known from the literature. We have modified them to bring them into a uniform format, to make sure that the definitions we use are prudently chosen with respect to proving soundness and completeness, and to make them sound and complete in the first place.

See Section 3.2

9.2 Soundness and Completeness

We have proven soundness and completeness properties for a number of discrete-time process algebras with relative timing, containing features such as immediate deadlock, unbounded start delay, delayable actions, free merge, merge, communication merge, and encapsulation.

See Chapters 4 and 5

In order to prove soundness and completeness, we have used several distinct techniques, ranging from simple induction proofs to term-rewriting analysis. Furthermore, we have shown how to extend given soundness and completeness results for a certain process algebra to another process algebra, using a ground equivalence technique if

the signatures of the old and the new process algebra are the same, or using Verhoef's method if the signature of the new process algebra is larger than that of the old one.

See Section 4.2

On the negative side: the methods we have used for proving soundness and completeness do not seem to extend to abstract process algebras.

See Section 5.5

While proving soundness and completeness, we found that it is very easy to make tiny mistakes when constructing an axiomatization, and that these mistakes only become apparent when soundness and completeness proofs are written out, *ad nauseam*, in all their entirety. This, combined with the fact that the proofs in themselves are not very complex but merely long and tedious, brings us to the conclusion that proofs of this kind are excellent candidates for automated proof checking or computer aided proof generation.

See Section 5.5

All the soundness and completeness proofs we have given are constructive. Therefore, these proofs can be viewed as algorithms. Using these algorithms, it should be not too difficult to construct a tool that can automatically decide whether two closed process terms are derivably equal in a given process algebra, and if so, provide a derivation.

See Chapters 4 and 5

Doing soundness and completeness proofs provides us with a clear insight into the exact inner workings of an axiomatization. Using this insight, we were able to replace conditional axioms by corresponding unconditional axioms. The recipe was always the same: identify the places in the completeness proof where the conditional axiom is used (which places turn out to be very localized), see how the conditional axiom is instantiated there, and add the corresponding equality as an unconditional axiom. Using this recipe, we were able to give unconditional variants of all conditional axiomatizations that occur in this thesis.

See Sections 4.3.5, 5.3.2, 5.3.4, and 5.3.5

9.3 Axioms for Concurrency

We have given two distinct ways to introduce concurrency in the setting of discrete-time process algebra with relative timing, and proved the resulting process algebras sound and complete.

See Chapter 5

The first way to introduce concurrency, using the so-called ν/σ -axiomatization style, proved very convenient from a practical viewpoint, as it makes calculations very practical. From a theoretical viewpoint, however, this axiomatization style is more troublesome, as it prevents the use of term rewriting techniques, and is also incompatible with the empty process. We conclude that the ν/σ -axiomatization style has its uses, but is not very elegant.

See Sections 5.2.1 and 5.2.3

The second way to introduce concurrency, using a more classic axiomatization style that is based on inductive definitions of closed terms, has less theoretical problems. Due to the fact that it is based on inductive definitions, it lends itself well to proving properties by means of induction. Also, it appears to lend itself better to being extended with additional features. However, the classic axiomatization style is less intuitive, and it is

not convenient to perform calculations in. We conclude this axiomatization style is more elegant, but not quite as elegant as we would want it to be.

See Sections 5.2.2 and 5.2.4

We have supplemented our discrete-time concurrent process algebras with delayable actions. To do this, we initially needed conditional axiomatizations, but ultimately we were able to derive unconditional variants of all these axiomatizations.

See Section 5.3

9.4 The Empty Process

We have successfully introduced the empty process in the context of discrete-time process algebra with relative timing. The axioms we have given lead to a sound and complete axiomatization of our bisimulation model. For closed terms, the axioms of standard concurrency are derivable.

See Chapter 6

In building our axiomatization, we found that there is not much room for choice: the constraints of the unit-element property with respect to sequential composition and merge, associativity of the merge, time determinism, and taking an existing concrete process algebra as a basis, almost completely determine which course to take.

See Section 6.3

As the behavior of the empty process is not always in accordance with one's first intuition one should be very careful when verifying protocols, to make sure that the protocol that is coded in process algebra, is indeed the same as the one that is supposed to be under study. Hence, the usefulness of the empty process with respect to real-life protocol verification remains to be determined.

See Section 6.3.2

We found that the empty process cannot straightforwardly be combined with the immediate deadlock process. This is rather worrisome, as some applications require both features simultaneously.

See Section 6.3.4

9.5 Fischer's Protocol

We have successfully verified a simple instance of Fischer's Protocol for Mutual Exclusion. The first part of the verification was algebraic, but the second part involved model-dependent reasoning.

See Section 7.5

We conclude that the use of bisimulation semantics in discrete-time process algebra places a burden on verifications, as bisimulation semantics preserve very many moments of choice, and therefore lead to long and error-prone calculations. It seems valid to doubt whether this is always necessary. Maybe a less discriminatory semantics should be developed for discrete-time process algebra.

See Section 7.6

We conclude that a pure algebraic approach is sometimes not very well suited to express the properties we want to prove of a certain protocol. In the case of Fischer's Pro-

toocol we run into serious problems when we want to formally express our proof requirements. As a result, the proof requirements remain too informal. Maybe discrete-time process algebra should be augmented with a temporal-logic oriented formalism to specify proof requirements.

See Section 7.2.2

We conclude that the process-algebraic verification of real-life protocols requires a large calculational effort. Therefore, we think it essential that computer support be developed, so that the large amount of calculations necessary can be more easily handled. This will be beneficial to both the size of the problems that can be handled, and to the accuracy with which the verification is performed.

See Section 7.6

9.6 Future Research

As can be observed from the conclusions above, and from the literature in general, discrete-time process algebra has been developed to a state of maturity that is satisfactory from a theoretical point of view, but that still leaves much to be desired in terms of practical applications. It seems that the design of discrete-time process algebra has often been focused on achieving pleasing theoretical results, while ultimately the attractiveness of timed process algebra lies in its application to protocol verification.

This does not mean that timed process algebra possesses no virtue of and in itself, but we cannot seriously claim that it is a discipline with applications in protocol verification, when most of the work that is being done is of a purely mathematical nature. If timed process algebra is to remain an active area of research in the 21st century, we must now focus our attention on gearing it towards practical applicability. Or, if we insist on developing more theory, we should let ourselves be guided by the demands that are made by the application of timed process algebra to real-life verifications.

Therefore, as directions for future research, we suggest the following:

- The development of efficient, professional tool support for the manipulation of timed process-algebraic specifications. We think of a system along the lines of the PSF system (MAUW [136], MAUW AND VELTINK [138], and VELTINK [189]), augmented with timed process algebra, and an order of a magnitude more complete, both in terms of capabilities and of easy of use. However, given the fact that the construction of PSF required several person-years (MAUW [137]), it seems reasonable to estimate that the construction of a mature timed variant would require several person-decades.
- The development of methods to integrate timed process-algebraic proofs in the framework of proof checking tools, such as *Coq* or *PVS*. In order to do this, we need a clean, formal, machine-readable specification language for timed process algebra. A promising candidate for this is *Timed μ CRL* (GROOTE [89]).
- The development of timed process algebras that do not use bisimulation semantics, but a less discriminatory process equivalence, for example ready semantics or failure semantics.

- The development of process algebras that combine pure algebraic methods with a non-algebraic specification formalism, for example a variant of temporal logic, like in BAETEN, BERGSTRA, AND BOL [28]. As argued before, this would greatly enhance our ability to formalize the properties of timed protocols.
- An ambitious effort to formalize, study, and ultimately verify, substantially larger protocols than has been done up till now.

We would like to conclude this chapter with an observation from GROOTE [89]. He argues that in the development of untimed process algebra, it has also been the case that theoretical advancements have always occurred much earlier than practical applications. Only when the theory behind untimed process algebra had stabilized and matured, he notices, did people successfully develop methods and tools to analyze and verify large systems. Furthermore, after the introduction of computer aided protocol verification (by means of the μ CRL formalism, see GROOTE AND PONSE [91] and KORVER [120]) it became possible to handle the verification of protocols many times larger than before.

If the above observation is correct, we may look forward to the successful verification of large time-dependent protocols in the near future.

A

Overview of Axioms

In this appendix we give a list of all axioms that occur in this thesis, and describe their nomenclature.

A.1 Nomenclature

Traditionally (BAETEN AND WEIJLAND [38]), the axioms of process algebras had simple names, consisting of one or two letters followed by a digit. For example, the axioms of BPA are simply called A1 to A5, for “Axiom 1” to “Axiom 5”. For more complex process algebras, more complex names were needed, and gradually the naming scheme has been expanded. We give some guidelines which we have tried to follow:

- Most (but not all) axioms that were already present in BAETEN AND WEIJLAND [38] have retained their names. Furthermore, some (but not all) new axioms that have clear counterparts in the old axioms, have received names that are clearly related to the names of their counterpart. For example, the untimed axiom CM4 has a discrete relative-time counterpart DRTCM4.
- Letters before the digits of an axiom often give an indication of the operator it defines. We have “B” for τ -laws (after Branching bisimulation), “CF” for Communication Function, “CM” for Communication Merge, “D” for encapsulation (after the ∂_H notation), “DCS” for Discrete Current Slice (the ν_{rel} operator), “DRT” for Discrete Relative Time, “E” for Empty process, “ID” for Immediate Deadlock, “M” for (free) Merge, “T” for abstraction (after the τ_I notation), and “USD” for Unbounded Start Delay. Note that these letters can be combined, so for example DRTECM6 is an axiom for the Empty process and the Communication Merge in Discrete Relative Time.
- Letters after the digits of an axiom indicate that the axiom is a variant of another axiom. Here “A” stands for Alternative version, and “ID” for Immediate Deadlock version. So, Axiom A6A is an axiom that can, in some contexts, serve as an alternative for Axiom A6, and Axiom M2ID is a version of Axiom M2 that has been adapted for the immediate deadlock. Note the difference between axioms that have “ID” before the digits, and those that have “ID” after the digits. For example, Axiom MID3

is an axiom for the merge and the immediate deadlock (that has no counterpart in a setting without immediate deadlock), and M3ID is an axiom for the merge that has been adapted for immediate deadlock (and that does have a counterpart in a setting without immediate deadlock, namely Axiom M3).

- Some axioms occur under several different names, for example CM1, DRTCM1, and DRTECM1. This was done to bring more structure in the definitions of which axioms are contained within a certain process algebra. If we look, for example, at the process algebras that are listed in Section B.3 on page 265, we see that these definitions would be much more fragmented if we would not give new, contiguous names to axioms that have already been used before.
- Some axioms have incomplete names, for example Axiom DRT1, for “Discrete Relative Time Axiom 1”. This was done for lack of an alternative, i.e., these axioms do not fit very well into the naming scheme.
- Some axioms have historically received ad-hoc names that fall outside the naming scheme, for example Axiom DA (for Delayable Action) and Axiom TF (for Time Factorization).

One final note: the axiom naming scheme has developed in an ad-hoc manner, and not always for the better. Some authors have given up on finding meaningful names, and simply do not give their axioms names at all. We feel that axiom names are indispensable though, because we need to refer to axioms in a concise and unambiguous way. This does not imply, however, that we are content with the current naming scheme.

A.2 List of Axioms

Below we give a list of all axioms that appear in this thesis.

$x + y = y + x$	A1
$(x + y) + z = x + (y + z)$	A2
$x + x = x$	A3
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5
$x + \delta = x$	A6
$a + \delta = a$	A6A
$x + \dot{\delta} = x$	A6ID
$\delta \cdot x = \delta$	A7
$\dot{\delta} \cdot x = \dot{\delta}$	A7ID
$x \cdot \varepsilon = x$	A8
$\varepsilon \cdot x = x$	A9
$a = \lfloor \underline{a} \rfloor^\omega$	ATS
$a b = c \quad \text{if } \gamma(a, b) = c$	CF

$x \parallel y = x \perp\!\!\!\perp y + y \perp\!\!\!\perp x + x \mid y$	CM1
$a \mid b \cdot x = (a \mid b) \cdot x$	CM2
$a \cdot x \mid b = (a \mid b) \cdot x$	CM3
$a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel y)$	CM4
$(x + y) \mid z = x \mid z + y \mid z$	CM5
$x \mid (y + z) = x \mid y + x \mid z$	CM6
$\partial_H(a) = a \quad \text{if } a \notin H$	D1
$\partial_H(a) = \delta \quad \text{if } a \in H$	D2
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D4
$\partial_H(\varepsilon) = \varepsilon$	D5
$\partial_H(\dot{\delta}) = \dot{\delta}$	D6
$a = \varepsilon \cdot \underline{a}$	DA
$\nu_{\text{rel}}(\underline{a}) = \underline{a}$	DCS1
$\nu_{\text{rel}}(x + y) = \nu_{\text{rel}}(x) + \nu_{\text{rel}}(y)$	DCS2
$\nu_{\text{rel}}(x \cdot y) = \nu_{\text{rel}}(x) \cdot y$	DCS3
$\nu_{\text{rel}}(\sigma_{\text{rel}}(x)) = \underline{\underline{\delta}}$	DCS4
$\nu_{\text{rel}}(\underline{\varepsilon}) = \underline{\varepsilon}$	DCSE1
$\nu_{\text{rel}}(x + y) = \nu_{\text{rel}}(x) + \nu_{\text{rel}}(y)$	DCSE2
$\nu_{\text{rel}}(\underline{a} \cdot x) = \underline{a} \cdot x$	DCSE3
$\nu_{\text{rel}}(\underline{\sigma} \cdot x) = \underline{\underline{\delta}}$	DCSE4
$\nu_{\text{rel}}(\dot{\delta}) = \dot{\delta}$	DCSID
$\varepsilon = \underline{\varepsilon} + \underline{\sigma} \cdot \varepsilon$	DEP
$\sigma_{\text{rel}}(x) + \sigma_{\text{rel}}(y) = \sigma_{\text{rel}}(x + y)$	DRT1
$\sigma_{\text{rel}}(x) \cdot y = \sigma_{\text{rel}}(x \cdot y)$	DRT2
$\underline{\underline{\delta}} \cdot x = \underline{\underline{\delta}}$	DRT3
$\underline{a} + \underline{\delta} = \underline{a}$	DRT4
$x + \underline{\underline{\delta}} = x$	DRT4A
$\sigma_{\text{rel}}(x) + \underline{\underline{\delta}} = \sigma_{\text{rel}}(x)$	DRT5
$x \cdot (\underline{\tau} \cdot (\nu_{\text{rel}}(y) + z + \underline{\underline{\delta}})) + \nu_{\text{rel}}(y) = x \cdot (\nu_{\text{rel}}(y) + z + \underline{\underline{\delta}})$	DRTB1
$x \cdot (\underline{\tau} \cdot (\nu_{\text{rel}}(y) + z + \underline{\underline{\delta}})) + z = x \cdot (\nu_{\text{rel}}(y) + z + \underline{\underline{\delta}})$	DRTB2
$x \cdot (\sigma_{\text{rel}}(\underline{\tau} \cdot (y + \underline{\underline{\delta}})) + \nu_{\text{rel}}(z)) = x \cdot (\sigma_{\text{rel}}(y + \underline{\underline{\delta}})) + \nu_{\text{rel}}(z)$	DRTB3
$x \cdot (\tau \cdot [y + z + \underline{\underline{\delta}}]^\omega + [y]^\omega) = x \cdot [y + z + \underline{\underline{\delta}}]^\omega$	DRTB4
$\underline{a} \mid \underline{b} = \underline{c} \quad \text{if } y(a, b) = c$	DRTCF
$x \parallel y = x \perp\!\!\!\perp y + y \perp\!\!\!\perp x + x \mid y$	DRTCM1
$\underline{a} \mid \underline{b} \cdot x = (\underline{a} \mid \underline{b}) \cdot x$	DRTCM2
$\underline{a} \cdot x \mid \underline{b} = (\underline{a} \mid \underline{b}) \cdot x$	DRTCM3
$\underline{a} \cdot x \mid \underline{b} \cdot y = (\underline{a} \mid \underline{b}) \cdot (x \parallel y)$	DRTCM4

$\sigma_{\text{rel}}(x) \mid \sigma_{\text{rel}}(y) = \sigma_{\text{rel}}(x \mid y)$	DRTCM5
$\sigma_{\text{rel}}(x) \mid \nu_{\text{rel}}(y) = \underline{\underline{\delta}}$	DRTCM6
$\sigma_{\text{rel}}(x) \mid (\nu_{\text{rel}}(y) + \underline{\underline{\delta}}) = \underline{\underline{\delta}}$	DRTCM6ID
$\nu_{\text{rel}}(x) \mid \sigma_{\text{rel}}(y) = \underline{\underline{\delta}}$	DRTCM7
$(\nu_{\text{rel}}(x) + \underline{\underline{\delta}}) \mid \sigma_{\text{rel}}(y) = \underline{\underline{\delta}}$	DRTCM7ID
$\underline{\underline{a}} \mid \sigma_{\text{rel}}(x) = \underline{\underline{\delta}}$	DRTCM8
$\sigma_{\text{rel}}(x) \mid \underline{\underline{a}} = \underline{\underline{\delta}}$	DRTCM9
$\underline{\underline{a}} \cdot x \mid \sigma_{\text{rel}}(y) = \underline{\underline{\delta}}$	DRTCM10
$\sigma_{\text{rel}}(x) \mid \underline{\underline{a}} \cdot y = \underline{\underline{\delta}}$	DRTCM11
$(x + y) \mid z = x \mid z + y \mid z$	DRTCM12
$x \mid (y + z) = x \mid y + x \mid z$	DRTCM13
$\partial_H(\underline{\underline{a}}) = \underline{\underline{a}} \quad \text{if } a \notin H$	DRTD1
$\partial_H(\underline{\underline{a}}) = \underline{\underline{\delta}} \quad \text{if } a \in H$	DRTD2
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	DRTD3
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	DRTD4
$\partial_H(\sigma_{\text{rel}}(x)) = \sigma_{\text{rel}}(\partial_H(x))$	DRTD5
$\partial_H(\underline{\underline{\delta}}) = \underline{\underline{\delta}}$	DRTD6
$x + \underline{\underline{\delta}} = x$	DRTE1
$\underline{\underline{\delta}} \cdot x = \underline{\underline{\delta}}$	DRTE2
$x \cdot \underline{\underline{\varepsilon}} = x$	DRTE3
$(x + \underline{\underline{\delta}}) \cdot \underline{\underline{\varepsilon}} = x + \underline{\underline{\delta}}$	DRTE3ID
$\underline{\underline{\varepsilon}} \cdot x = x$	DRTE4
$\underline{\underline{\varepsilon}} \cdot (x + \underline{\underline{\delta}}) = x + \underline{\underline{\delta}}$	DRTE4ID
$\underline{\underline{a}} + \underline{\underline{\delta}} = \underline{\underline{a}}$	DRTE5
$\underline{\underline{\sigma}} + \underline{\underline{\delta}} = \underline{\underline{\sigma}}$	DRTE6
$\underline{\underline{\varepsilon}} + \underline{\underline{\delta}} = \underline{\underline{\varepsilon}}$	DRTE7
$\underline{\underline{a}} \mid \underline{\underline{b}} = \underline{\underline{c}} \quad \text{if } y(a, b) = c$	DRTECF
$x \parallel y = x \parallel y + y \parallel x + x \mid y$	DRTECM1
$\underline{\underline{a}} \cdot x \mid \underline{\underline{b}} \cdot y = (\underline{\underline{a}} \mid \underline{\underline{b}}) \cdot (x \parallel y)$	DRTECM2
$\underline{\underline{\sigma}} \cdot x \mid \underline{\underline{\sigma}} \cdot y = \underline{\underline{\sigma}} \cdot (x \parallel y)$	DRTECM3
$\underline{\underline{\sigma}} \cdot x \mid \underline{\underline{a}} \cdot y = \underline{\underline{\delta}}$	DRTECM4
$\underline{\underline{a}} \cdot x \mid \underline{\underline{\sigma}} \cdot y = \underline{\underline{\delta}}$	DRTECM5
$x \mid \underline{\underline{\varepsilon}} = \underline{\underline{\delta}}$	DRTECM6
$\underline{\underline{\varepsilon}} \mid x = \underline{\underline{\delta}}$	DRTECM7
$(x + y) \mid z = x \mid z + y \mid z$	DRTECM8
$x \mid (y + z) = x \mid y + x \mid z$	DRTECM9
$\partial_H(\underline{\underline{a}}) = \underline{\underline{a}} \quad \text{if } a \notin H$	DRTED1
$\partial_H(\underline{\underline{a}}) = \underline{\underline{\delta}} \quad \text{if } a \in H$	DRTED2

$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	DRTED3
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	DRTED4
$\partial_H(\underline{\sigma}) = \underline{\sigma}$	DRTED5
$\partial_H(\underline{\varepsilon}) = \underline{\varepsilon}$	DRTED6
$x \parallel y = x \ll y + y \ll x$	DRTEM1
$\underline{a} \cdot x \ll y = \underline{a} \cdot (x \parallel y)$	DRTEM2
$(x + y) \ll z = x \ll z + y \ll z$	DRTEM3
$\underline{\varepsilon} \ll \underline{\varepsilon} = \underline{\varepsilon}$	DRTEM4
$\underline{\varepsilon} \ll \underline{a} \cdot x = \underline{\delta}$	DRTEM5
$\underline{\varepsilon} \ll \underline{\sigma} \cdot x = \underline{\delta}$	DRTEM6
$\underline{\varepsilon} \ll (x + y) = \underline{\varepsilon} \ll x + \underline{\varepsilon} \ll y$	DRTEM7
$\underline{\sigma} \cdot x \ll (\underline{a} \cdot y + z) = \underline{\sigma} \cdot x \ll z$	DRTEM8
$\underline{\sigma} \cdot x \ll \underline{\delta} = \underline{\delta}$	DRTEM9
$\underline{\sigma} \cdot x \ll \underline{\varepsilon} = \underline{\sigma} \cdot x$	DRTEM10
$\underline{\sigma} \cdot x \ll \underline{\sigma} \cdot y = \underline{\sigma} \cdot (x \ll y)$	DRTEM11
$\underline{\sigma} \cdot x \ll (\underline{\sigma} \cdot y + \underline{\varepsilon}) = \underline{\sigma} \cdot (x \ll y)$	DRTEM12
$\underline{\sigma} \cdot \delta = \underline{\delta}$	DRTESID
$x \parallel y = x \ll y + y \ll x$	DRTM1
$\underline{a} \ll x = \underline{a} \cdot x$	DRTM2
$\underline{a} \ll (x + \underline{\delta}) = \underline{a} \cdot (x + \underline{\delta})$	DRTM2ID
$\underline{a} \cdot x \ll y = \underline{a} \cdot (x \parallel y)$	DRTM3
$\underline{a} \cdot x \ll (y + \underline{\delta}) = \underline{a} \cdot (x \parallel (y + \underline{\delta}))$	DRTM3ID
$(x + y) \ll z = x \ll z + y \ll z$	DRTM4
$\sigma_{\text{rel}}(x) \ll \nu_{\text{rel}}(y) = \underline{\delta}$	DRTM5
$\sigma_{\text{rel}}(x) \ll (\nu_{\text{rel}}(y) + \underline{\delta}) = \underline{\delta}$	DRTM5ID
$\sigma_{\text{rel}}(x) \ll (\nu_{\text{rel}}(y) + \sigma_{\text{rel}}(z)) = \sigma_{\text{rel}}(x \ll z)$	DRTM6
$\sigma_{\text{rel}}(x) \ll \underline{a} = \underline{\delta}$	DRTM7
$\sigma_{\text{rel}}(x) \ll \underline{a} \cdot y = \underline{\delta}$	DRTM8
$\sigma_{\text{rel}}(x) \ll (\underline{a} + y) = \sigma_{\text{rel}}(x) \ll y$	DRTM9
$\sigma_{\text{rel}}(x) \ll (\underline{a} \cdot y + z) = \sigma_{\text{rel}}(x) \ll z$	DRTM10
$\sigma_{\text{rel}}(x) \ll \sigma_{\text{rel}}(y) = \sigma_{\text{rel}}(x \ll y)$	DRTM11
$x \ll \dot{\delta} = \dot{\delta}$	DRTMID1
$\dot{\delta} \ll x = \dot{\delta}$	DRTMID2
$x \mid \dot{\delta} = \dot{\delta}$	DRTMID3
$\dot{\delta} \mid x = \dot{\delta}$	DRTMID4
$\sigma_{\text{rel}}(\dot{\delta}) = \underline{\delta}$	DRTSID
$\tau_I(\underline{a}) = \underline{a}$ if $a \notin I$	DRTT1
$\tau_I(\underline{a}) = \underline{\tau}$ if $a \in I$	DRTT2

$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$	DRTT3
$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$	DRTT4
$\tau_I(\sigma_{\text{rel}}(x)) = \sigma_{\text{rel}}(\tau_I(x))$	DRTT5
$\tau_I(\dot{\delta}) = \dot{\delta}$	DRTT6
$x \cdot (y + z) = x \cdot y + x \cdot z$	LD
$x \parallel y = x \llcorner y + y \llcorner x$	M1
$a \llcorner x = a \cdot x$	M2
$a \llcorner (x + \delta) = a \cdot (x + \delta)$	M2ID
$a \cdot x \llcorner y = a \cdot (x \parallel y)$	M3
$a \cdot x \llcorner (y + \delta) = a \cdot (x \parallel (y + \delta))$	M3ID
$(x + y) \llcorner z = x \llcorner z + y \llcorner z$	M4
$\varepsilon \llcorner \varepsilon = \varepsilon$	ME1
$\varepsilon \llcorner a \cdot x = \delta$	ME2
$\varepsilon \llcorner (x + y) = \varepsilon \llcorner x + \varepsilon \llcorner y$	ME3
$x \mid \varepsilon = \delta$	ME4
$\varepsilon \mid x = \delta$	ME5
$x \llcorner \dot{\delta} = \dot{\delta}$	MID1
$\dot{\delta} \llcorner x = \dot{\delta}$	MID2
$x \mid \dot{\delta} = \dot{\delta}$	MID3
$\dot{\delta} \mid x = \dot{\delta}$	MID4
$\underline{\underline{\sigma}} \cdot x + \underline{\underline{\sigma}} \cdot y = \underline{\underline{\sigma}} \cdot (x + y)$	TF
$\llbracket x \rrbracket^\omega = \nu_{\text{rel}}(x) + \sigma_{\text{rel}}(\llbracket x \rrbracket^\omega)$	USD
$\llbracket a \rrbracket^\omega = a$	USD1
$\llbracket x \cdot y \rrbracket^\omega = \llbracket x \rrbracket^\omega \cdot y$	USD2
$\llbracket x + y \rrbracket^\omega = \llbracket x \rrbracket^\omega + \llbracket y \rrbracket^\omega$	USD3
$\llbracket \sigma_{\text{rel}}(x) \rrbracket^\omega = \delta$	USD4
$\llbracket \dot{\delta} \rrbracket^\omega = \delta$	USD5
$a \llcorner \llbracket x \rrbracket^\omega = a \cdot \llbracket x \rrbracket^\omega$	USD6
$a \cdot x \llcorner \llbracket y \rrbracket^\omega = a \cdot (x \parallel \llbracket y \rrbracket^\omega)$	USD7
$\llbracket x \mid y \rrbracket^\omega = \llbracket x \rrbracket^\omega \mid \llbracket y \rrbracket^\omega$	USD8
$\llbracket \partial_H(x) \rrbracket^\omega = \partial_H(\llbracket x \rrbracket^\omega)$	USD9
$a \mid b = c \quad \text{if } y(a, b) = c$	USDCF
$a \mid b \cdot x = (a \mid b) \cdot x$	USDCM2
$a \cdot x \mid b = (a \mid b) \cdot x$	USDCM3
$a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel y)$	USDCM4
$\partial_H(a) = a \quad \text{if } a \notin H$	USDD1
$\partial_H(a) = \delta \quad \text{if } a \in H$	USDD2

B

Overview of Process Algebras

In this appendix we give a list of all process algebras that occur in this thesis, describe their nomenclature, and give an overview of their signatures.

B.1 Nomenclature

In the naming of the process algebras defined in this thesis, we have used the following guidelines:

- We use “BPA” as a base name for process algebras that do not contain merge operators, “PA” for process algebras that only contain a free merge, and “ACP” for process algebras that contain a merge that supports a form of synchronization or communication.
- The presence of certain constants in the signature, e.g. δ , ε , $\dot{\delta}$, or τ , is indicated by placing these constants in a subscript to the base name. So, for example, $\text{BPA}_{\delta\varepsilon}$ denotes a basic process algebra with δ and ε in its signature. Whenever the presence of such a constant is self-evident, it is not included in the subscript. For example: PA_{ε} contains a δ , but this is not indicated in the subscript, because it is already implied by the presence of the ε and the \parallel .
- The subscript “ drt ” signifies “discrete relative time”.
- The superscript “ $-$ ” indicates the absence of delayable actions.
- The superscript “ $+$ ” indicates the presence of a recursion principle to deal with delayable actions (note that “ $-$ ” and “ $+$ ” are mutually exclusive).
- The presence of a prime (e.g., in BPA'_{drt}) indicates that the primed process algebra is a minor variation on the unprimed process algebra.
- The postfix “ $-\delta$ ” indicates the absence of all deadlock constants, non-delayable (“ $\underline{\delta}$ ”), delayable (“ δ ”), or immediate (“ $\dot{\delta}$ ”).
- The postfix “ $-\text{ID}$ ” indicates the absence of an immediate deadlock constant (“ $\dot{\delta}$ ”).

B.2 Signatures

Below we give tables showing the signatures of the process algebras we have described in this thesis. Table B.1 shows the process algebras that were defined in Chapter 2, Table B.2 shows the process algebras that were defined in Chapters 3, 4, and 5, and Table B.3 shows the process algebras that were defined in Chapters 6 and 7. The signatures of the process algebras BPA'_{drt} , PA'_{drt} , $PA_{drt}^{-ID'}$, $ACP_{drt}^{-ID'}$, ACP'_{drt} , and ACP''_{drt} are not given, as they are identical to the signatures of their unprimed counterparts.

	+	·	a	δ	ε	$\dot{\delta}$	\parallel	$\perp\!\!\!\perp$		∂_H
BPA	•	•	•							
BPA_{δ}	•	•	•	•						
BPA_{ε}	•	•	•		•					
$BPA_{\delta\varepsilon}$	•	•	•	•	•					
$BPA_{\dot{\delta}}$	•	•	•	•		•				
PA	•	•	•				•	•		
PA_{δ}	•	•	•	•			•	•		
PA_{ε}	•	•	•	•	•		•	•		
$PA_{\dot{\delta}}$	•	•	•	•		•	•	•		
ACP	•	•	•	•			•	•	•	•
ACP_{ε}	•	•	•	•	•		•	•	•	•
$ACP_{\dot{\delta}}$	•	•	•	•		•	•	•	•	•

Table B.1: Signatures of untimed process algebras.

	+	·	\underline{a}	$\underline{\delta}$	\bar{a}	$\bar{\delta}$	$\dot{\delta}$	σ	ν	$[]^{\omega}$	\parallel	$\perp\!\!\!\perp$		∂_H
$BPA_{drt}^{-\delta}$	•	•	•					•						
BPA_{drt}^{-ID}	•	•	•	•				•	•					
BPA_{drt}^{-}	•	•	•	•			•	•	•					
BPA_{drt}	•	•	•	•	•	•	•	•	•	•				
PA_{drt}^{-ID}	•	•	•	•				•	•		•	•		
PA_{drt}	•	•	•	•	•	•	•	•	•	•	•	•		
ACP_{drt}^{-ID}	•	•	•	•				•	•		•	•	•	•
ACP_{drt}	•	•	•	•	•	•	•	•	•	•	•	•	•	•

Table B.2: Signatures of concrete discrete-time process algebras.

	+	$\frac{a}{\delta}$	$\underline{\underline{\varepsilon}}$	$\underline{\underline{\tau}}$	$\frac{a}{\delta}$	ε	$\frac{\tau}{\delta}$	$\underline{\underline{\sigma}}$	σ	ν	$\lfloor \rfloor^\omega$	\parallel \perp	\mid ∂_H	τ_I
$BPA_{drt,\varepsilon}^-$ -ID	•	•	•					•		•				
$BPA_{drt,\varepsilon}$ -ID	•	•	•		•	•		•		•				
$PA_{drt,\varepsilon}^-$ -ID	•	•	•					•		•		•		
$PA_{drt,\varepsilon}$ -ID	•	•	•		•	•		•		•		•		
$ACP_{drt,\varepsilon}^-$ -ID	•	•	•					•		•		•	•	
$ACP_{drt,\varepsilon}$ -ID	•	•	•		•	•		•		•		•	•	
$ACP_{drt,\tau}$	•	•		•	•		•		•	•	•	•	•	•

Table B.3: Signatures of abstract discrete-time process algebras.

B.3 Axioms

In this section we list all process algebras that occur in this thesis, together with a list of axioms they contain.

Untimed Process Algebras

These process algebras were defined in Chapter 2:

- $BPA = A1-A5$
- $BPA_\delta = A1-A7$
- $BPA_\varepsilon = A1-A5 + A8-A9$
- $BPA_{\delta\varepsilon} = A1-A9$
- $BPA_{\dot{\delta}} = A1-A5 + A6A + A7 + A6ID-A7ID$
- $PA = A1-A5 + M1-M4$
- $PA_\delta = A1-A7 + M1-M4$
- $PA_\varepsilon = A1-A9 + M1 + M3-M4 + ME1-ME3$
- $PA_{\dot{\delta}} = A1-A5 + A6A + A7 + A6ID-A7ID + M1 + M2ID-M3ID + M4 + MID1-MID2$
- $ACP = A1-A7 + M2-M4 + CM1-CM6 + CF$
- $ACP_\varepsilon = A1-A9 + M3-M4 + ME1-ME5 + CM1 + CM4-CM6 + CF + D1-D5$
- $ACP_{\dot{\delta}} = A1-A5 + A6A + A7 + A6ID-A7ID + M2ID-M3ID + M4 + CM1-CM6 + CF + MID1-MID4 + D1-D4 + D6$

Discrete-Time Concrete Basic Process Algebras

These process algebras were defined in Chapters 3 and 4:

- $BPA_{drt}^- - \delta = A1-A5 + DRT1-DRT2$
- $BPA_{drt}^- - ID = A1-A5 + DRT1-DRT5 + DCS1-DCS4$
- $BPA_{drt}^- = A1-A5 + A6ID + A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID$
- $BPA_{drt} = A1-A5 + A6ID-A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD$
- $BPA'_{drt} = A1-A5 + A6ID-A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD + USD1-USD5$

Discrete-Time Concrete Concurrent Process Algebras

These process algebras were defined in Chapter 5:

- $PA_{drt}^- - ID = A1-A5 + DRT1-DRT5 + DCS1-DCS4 + DRTM1-DRTM6$
- $PA_{drt}^- - ID' = A1-A5 + DRT1-DRT5 + DCS1-DCS4 + DRTM1-DRTM4 + DRTM7-DRTM11$
- $ACP_{drt}^- - ID = A1-A5 + DRT1-DRT5 + DCS1-DCS4 + DRTM2-DRTM6 + DRTCM1-DRTCM7 + DRTCM12-DRTCM13 + DRTCF + DRTD1-DRTD5$
- $ACP_{drt}^- - ID' = A1-A5 + DRT1-DRT5 + DCS1-DCS4 + DRTM2-DRTM4 + DRTM7-DRTM11 + DRTCM1-DRTCM5 + DRTCM8-DRTCM13 + DRTCF + DRTD1-DRTD5$
- $PA_{drt} = A1-A5 + A6ID + A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD + DRTM1 + DRTM2ID-DRTM3ID + DRTM4 + DRTM5ID + DRTM6 + DRTMID1-DRTMID2$
- $PA'_{drt} = A1-A5 + A6ID + A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD + DRTM1 + DRTM2ID-DRTM3ID + DRTM4 + DRTM5ID + DRTM6 + DRTMID1-DRTMID2 + USD1-USD7$
- $ACP_{drt} = A1-A5 + A6ID + A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD + DRTM2ID-DRTM3ID + DRTM4 + DRTM5ID + DRTM6 + DRTCM1-DRTCM5 + DRTCM6ID-DRTCM7ID + DRTCM12-DRTCM13 + DRTCF + DRTD1-DRTD6 + DRTMID1-DRTMID4$
- $ACP'_{drt} = A1-A5 + A6ID + A7ID + DRT1-DRT5 + DRTSID + DCS1-DCS4 + DCSID + ATS + USD + DRTM2ID-DRTM3ID + DRTM4 + DRTM5ID + DRTM6 + DRTCM1-DRTCM5 + DRTCM6ID-DRTCM7ID + DRTCM12-DRTCM13 + DRTCF + DRTD1-DRTD6 + DRTMID1-DRTMID4 + USD1-USD7 + USDCF + USDCM2-USDCM4 + USDD1-USDD2$

- $ACP''_{\text{drt}} = A1-A5 + A6\text{ID} + A7\text{ID} + \text{DRT1-DRT5} + \text{DRTSID} + \text{DCS1-DCS4} + \text{DCSID} + \text{ATS} + \text{USD} + \text{DRTM2ID-DRTM3ID} + \text{DRTM4} + \text{DRTM5ID} + \text{DRTM6} + \text{DRTCM1-DRTCM5} + \text{DRTCM6ID-DRTCM7ID} + \text{DRTCM12-DRTCM13} + \text{DRTECF} + \text{DRTD1-DRTD6} + \text{DRTMID1-DRTMID4} + \text{USD1-USD9}$

Discrete-Time Process Algebras with Empty Process

These process algebras were defined in Chapter 6:

- $BPA_{\text{drt},\varepsilon}^- \text{-ID} = A1-A5 + \text{DRTE1-DRTE4} + \text{TF} + \text{DCSE1-DCSE4}$
- $PA_{\text{drt},\varepsilon}^- \text{-ID} = A1-A5 + \text{DRTE1-DRTE4} + \text{TF} + \text{DCSE1-DCSE4} + \text{DRTEM1-DRTEM12}$
- $ACP_{\text{drt},\varepsilon}^- \text{-ID} = A1-A5 + \text{DRTE1-DRTE4} + \text{TF} + \text{DCSE1-DCSE4} + \text{DRTEM2-DRTEM12} + \text{DRTECM1-DRTECM9} + \text{DRTECF} + \text{DRTED1-DRTED6}$
- $BPA_{\text{drt},\varepsilon} \text{-ID} = A1-A5 + \text{DRTE1-DRTE4} + \text{TF} + \text{DCSE1-DCSE4} + \text{DEP} + \text{DA}$
- $PA_{\text{drt},\varepsilon} \text{-ID} = A1-A5 + \text{DRTE1-DRTE4} + \text{TF} + \text{DCSE1-DCSE4} + \text{DEP} + \text{DA} + \text{DRTEM1-DRTEM12}$
- $ACP_{\text{drt},\varepsilon} \text{-ID} = A1-A5 + \text{DRTE1-DRTE4} + \text{TF} + \text{DCSE1-DCSE4} + \text{DEP} + \text{DA} + \text{DRTEM2-DRTEM12} + \text{DRTECM1-DRTECM9} + \text{DRTECF} + \text{DRTED1-DRTED6}$

Discrete-Time Process Algebras with Abstraction

This process algebra was defined in Chapter 7:

- $ACP_{\text{drt},\tau} = A1-A5 + A6\text{ID} + A7\text{ID} + \text{DRT1-DRT5} + \text{DRTSID} + \text{DCS1-DCS4} + \text{DCSID} + \text{ATS} + \text{USD} + \text{DRTM2ID-DRTM3ID} + \text{DRTM4} + \text{DRTM5ID} + \text{DRTM6} + \text{DRTCM1-DRTCM5} + \text{DRTCM6ID-DRTCM7ID} + \text{DRTCM12-DRTCM13} + \text{DRTECF} + \text{DRTD1-DRTD6} + \text{DRTMID1-DRTMID4} + \text{DRTB1-DRTB4} + \text{DRTT1-DRTT6}$

C

Overview of Theorems

In this appendix we give an overview of the most important definitions and theorems that appear in this thesis.

C.1 Elimination, Soundness, and Completeness

In Table C.1 on the next page we list the labels of the definitions of the axioms and semantics of the process algebras given in this thesis, and the labels of the corresponding theorems regarding elimination, soundness, and completeness. If a label is enclosed in parentheses, e.g., “(2.6.1.5)”, this means that the corresponding property has either not been completely proven or is not even true. In that case, the label either refers to an incomplete proof, or to a remark that gives further explanation.

	Axioms	Semantics	Elimination	Soundness	Completeness
<i>Untimed Process Algebras</i>					
BPA	2.3.1.6	2.3.1.11	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
BPA _δ	2.3.2.2	2.3.2.6	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
BPA _ε	2.3.3.2	2.3.3.4	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
BPA _{δε}	2.3.4.2	2.3.4.4	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
BPA _{δ̇}	2.3.5.2	2.3.5.3	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
PA	2.4.1.2	2.4.1.5	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
PA _δ	2.4.2.2	2.4.2.3	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
PA _ε	2.4.3.2	2.4.3.3	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
PA _{δ̇}	2.4.4.2	2.4.4.4	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
ACP	2.5.1.3	2.5.1.6	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
ACP _ε	2.5.2.2	2.5.2.3	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
ACP _{δ̇}	2.5.3.2	2.5.3.3	(2.6.1.5)	(2.6.1.7)	(2.6.1.9)
<i>Discrete-Time Concrete Process Algebras</i>					
BPA _{drt} ⁻ -δ	3.2.1.5	3.2.1.7	4.3.1.2	4.3.1.4	4.3.1.8
BPA _{drt} ⁻ -ID	3.2.2.2	3.2.2.4	4.3.2.1	4.3.2.3	4.3.2.6
BPA _{drt} ⁻	3.2.3.2	3.2.3.4	4.3.3.1	4.3.3.3	4.3.3.6
BPA _{drt} ⁺	3.2.4.5	3.2.4.9	4.3.4.1	4.3.4.5	4.3.4.9
BPA _{drt} [']	4.3.5.1	4.3.5.2	4.3.5.3	4.3.5.4	4.3.5.5
PA _{drt} ⁻ -ID	5.2.1.2	5.2.1.3	5.2.1.8	5.2.1.10	5.2.1.14
PA _{drt} ⁻ -ID'	5.2.2.2	5.2.2.3	5.2.2.8	5.2.2.17	5.2.2.19
PA _{drt} ⁺	5.3.1.2	5.3.1.3	5.3.1.9	5.3.1.11	5.3.1.14
PA _{drt} [']	5.3.2.1	5.3.2.2	5.3.2.3	5.3.2.4	5.3.2.5
ACP _{drt} ⁻ -ID	5.2.3.2	5.2.3.3	5.2.3.8	5.2.3.10	5.2.3.13
ACP _{drt} ⁻ -ID'	5.2.4.2	5.2.4.3	5.2.4.8	5.2.4.12	5.2.4.14
ACP _{drt} ⁺	5.3.3.2	5.3.3.3	5.3.3.12	5.3.3.14	5.3.3.17
ACP _{drt} [']	5.3.4.1	5.3.4.2	5.3.4.3	5.3.4.4	5.3.4.5
ACP _{drt} ^{''}	5.3.5.1	5.3.5.3	5.3.5.5	5.3.5.7	5.3.5.8
<i>Discrete-Time Abstract Process Algebras</i>					
BPA _{drt,ε} ⁻ -ID	6.3.1.3	6.3.1.4	(6.3.1.16)	(6.3.1.17)	6.3.1.19
BPA _{drt,ε} ⁺ -ID	6.4.1.3	6.4.1.7	(6.4.1.12)	(6.4.1.13)	(6.4.1.14)
PA _{drt,ε} ⁻ -ID	6.3.2.2	6.3.2.7	(6.3.2.11)	(6.3.2.12)	(6.3.2.13)
PA _{drt,ε} ⁺ -ID	6.4.2.2	6.4.2.3	(6.4.2.7)	(6.4.2.8)	(6.4.2.9)
ACP _{drt,ε} ⁻ -ID	6.3.3.2	6.3.3.3	(6.3.3.7)	(6.3.3.8)	(6.3.3.9)
ACP _{drt,ε} ⁺ -ID	6.4.3.2	6.4.3.3	(6.4.3.7)	(6.4.3.8)	(6.4.3.9)
ACP _{drt,τ}	7.3.2.2	7.3.2.4	(7.3.2.10)	(7.3.2.10)	(7.3.2.10)

Table C.1: Overview of definitions and theorems.

D

Notational Issues

In this appendix we define the general purpose symbols we have used, and give a historical overview of process algebraic notations that have been used in other publications. For an overview of all symbols, see the Symbol Index on page 303.

D.1 Mathematical Symbols

Definition D.1.1.1 (Numbers)

We use the symbol \mathbb{N} to denote the *natural numbers*: $\mathbb{N} = \{0, 1, 2, \dots\}$, and \mathbb{N}^+ to denote the *positive natural numbers*: $\mathbb{N}^+ = \{1, 2, \dots\}$. We use the symbol \mathbb{R} to denote the *real numbers*, and the symbol $\mathbb{R}^{\geq 0}$ to denote the *non-negative real numbers*: $\mathbb{R}^{\geq 0} = \{x \in \mathbb{R} \mid x \geq 0\}$.

Definition D.1.1.2 (Set Theory)

We use the symbol \in to denote set membership, the symbol \subseteq to denote the subset relation, and the symbol \times to denote a Cartesian product. A (repeated) Cartesian product of a set with itself is denoted by a superscript $n \in \mathbb{N}$, for example: $\mathbb{R}^3 = \mathbb{R} \times \mathbb{R} \times \mathbb{R}$.

Definition D.1.1.3 (Iff)

We use the word “iff” to denote “if and only if”.

Definition D.1.1.4 (Syntactical Equivalence)

Given two process terms x and y , we denote their syntactical equivalence by $x \equiv y$.

Definition D.1.1.5 (Equivalence by Definition)

Given a meta-expression x and a process term y , we write $x \stackrel{\text{def}}{=} y$ to denote that x is by definition equal to y .

Definition D.1.1.6 (Composition of Relations)

For two binary relations R and R' on a domain D , we denote their *composition* by $R \circ R'$:

$$R \circ R' = \{(x, z) \in D \times D \mid \exists y \in D : (x, y) \in R \text{ and } (y, z) \in R'\}$$

Definition D.1.1.7 (Reflexive Closure)

For a binary relation R on a domain D , we denote its *reflexive closure* by R^R :

$$R^R = R \cup \{(x, y) \in D \times D \mid x \equiv y\}$$

Definition D.1.1.8 (Symmetric Closure)

For a binary relation R on a domain D , we denote its *symmetric closure* by R^S :

$$R^S = R \cup \{(y, x) \in D \times D \mid (x, y) \in R\}$$

Definition D.1.1.9 (Transitive Closure)

For a binary relation R on a domain D , we denote its *transitive closure* by R^T , where R^T is the smallest relation R' such that $R' = R' \circ R'$ and $R \subseteq R'$.

Definition D.1.1.10 (Halmos)

We use the *Halmos*, denoted \blacksquare , to indicate the end of a proof.

Definition D.1.1.11 (Generic Operator)

We use the symbol \otimes to denote a *generic operator*, i.e., \otimes is a variable that ranges over the set of all operators applicable in a certain context.

Definition D.1.1.12 (Term Rewriting)

We use the symbol \rightarrow to denote the *rewrite relation* in the rewriting rules of a term-rewriting system, the symbol $<$ to denote a *well-founded ordering*, and the symbol $>_{\text{lpo}}$ to denote a *lexicographical path ordering*. Furthermore, we use the notation \otimes^* to indicate a “starred” operator, and \otimes_n to indicate a subscripted operator, both in the context of a lexicographical path ordering proof.

D.2 Alternative Process Algebraic Notations

When we compare the notations we have used in this thesis with the process-algebraic publications from the bibliography, the following differences are important:

- We find the notation “ BPA_{dt} ” for “ $\text{BPA}_{\text{drt}}^- - \delta$ ”, “ $\text{BPA}_{\delta\text{dt}}$ ” for “ $\text{BPA}_{\text{drt}}^- - \text{ID}$ ”, and “ $\text{PA}_{\delta\text{dt}}$ ” for “ $\text{PA}_{\text{drt}}^- - \text{ID}$ ” [37].
- We find the notation “ ACP_{dt} ” for “ $\text{ACP}_{\text{drt}}^- - \text{ID}$ ” [37, 191],
- We find the notations “ σ_a ” and “ $\sigma_{\text{rel}(1)}$ ” for “ σ_{rel} ” [37, 191].
- We find the notation “ $\text{cts}(a)$ ” (“*current time slice*”) for the undelayable action “ \underline{a} ”, and “ $\text{ats}(a)$ ” (“*any time slice*”) for the delayable action “ a ” [21, 24].
- We find the notation “ $\underline{a}[1]$ ” for the undelayable action “ \underline{a} ” [16], “ $a[1]$ ” (with the square brackets in a fuzzy outline font) and “ \underline{a} ” (idem dito for the underline) for the undelayable action “ \underline{a} ” [17], and “ \underline{a} ” for the delayable action “ a ” [16, 17].

- It has been suggested to refer to the delayable deadlock constant “ δ ” as *livelock*, and to the immediate deadlock constant “ $\dot{\delta}$ ” as the *immediate time stop*, *full time stop*, or *catastrophic deadlock* [12, 24].
- The unbounded start delay operator “[] ^{ω} ” derives from NICOLLIN AND SIFAKIS [154], and is therefore also known as one of the *Nicollin-Sifakis operators*.
- The naming scheme for axioms (A1, A2, etc.) has become quite problematic. Naming is generally not applied consistently *within* ACP articles published, let alone *across* them. We have no illusions this thesis is any different, although, and because, we sort of tried to adhere to the latest fashions (see Appendix A).

Summary

Process Algebra

Process algebra is a means of *specifying, verifying, and in general, talking about* computer algorithms and protocols in a clearly defined, formal manner. Compared with other formalisms developed for this purpose, process algebra's strong point is the fact that it is based on *algebra*. In this way, manipulations with processes become *algebraic manipulations*, allowing room for high-level abstractions to be made, potentially avoiding complexity blow-up.

The whole of methods and formalisms collectively known as *process algebra* can be subdivided into separate parts of which every single one, again, is called a *process algebra*.

At the heart of every process algebra lies a set of *axioms*. Every axiom consists of an *equality* between two *process terms* (that may contain free variables). Behind every axiom there is an intuitive motivation: an insight that explains why these two process terms should be considered equal. Together these axioms lead to an *algebra*: a mathematical structure that allows for the manipulation of terms.

Given the set of axioms of a certain process algebra, it is possible to construct a *model*: a mathematical "world" that obeys the equalities given by the axioms. Such a model is called a *semantics* for that process algebra. Typically, several clearly distinct models can be given for any given process algebra. However, there is a tendency always to use the same kind of model, called a *bisimulation model*.

In Chapter 2 we give an introduction into process algebra. We start by introducing a very simple process algebra, called BPA, and gradually add more features like *deadlock*, the *empty process*, the *immediate deadlock*, the *free merge operator*, and the *merge operator*. The chapter ends with a short discussion of the properties of process algebras.

Discrete Time

Originally, process algebra lacked the ability of quantitatively reasoning about *time*: although we can express the *sequence* in which actions take place, we cannot make explicit *the moment in time* at which they do.

One can reason that abstracting from timing aspects is justifiable, because often timing aspects are irrelevant in the verification of systems. However, sometimes the correctness of a protocol hinges on delicate timing aspects, or the timing aspects themselves of a protocol are the object of study. In such cases quantitative analysis of timing aspects cannot be avoided, and untimed process algebra is therefore of little help. We can try to

artificially encode timing aspects using untimed process algebra, but that is stretching untimed process algebra to its limits; clearly it was never built for these purposes.

One way of introducing time is to cut up time into a countably infinite number of *time-slices*. Between these time-slices, a *clock tick* (also called *time-step*), takes place that is explicitly indicated. This leads to so-called *discrete-time process algebra*: the passing of time is modeled as being *discrete*, consisting of the passage of a number of distinct time-slices.

In Chapter 3 we give an introduction into discrete-time process algebra with *relative timing* and introduce some process algebras that are discrete-time extensions of BPA.

Soundness and Completeness

Given a certain process algebra, we distinguish the *axioms*, that define an *algebraic equivalence relation*, and the *semantics*, that define a *model* of those axioms. If it is the case that all statements that logically follow from the axioms are indeed fulfilled in the model, we say that the process algebra has the *soundness property*. Conversely, if it is the case that all statements that are fulfilled in the model follow logically from the axioms, we say that the process algebra has the *completeness property*. Put in other words: *soundness* means that the axioms *always tell the truth* about the model, and *completeness* means that the axioms *can tell everything* about the model.

If a process algebra has both the soundness and the completeness property, the axioms and the model are in perfect harmony: statements that follow from the axioms, hold in the model, and statements that do not follow from the axioms, do also not hold in the model. So, loosely speaking, the axioms and the model are equivalent.

In Chapter 4 we prove these two properties, and some others, for the process algebras that are defined in Chapter 3.

Axioms for Concurrency

A key aspect of process algebra (and of any other formal verification method) is the use of *concurrent processes*: processes that are simultaneously active, executing in parallel, possibly communicating with each other, or influencing each other's behavior.

In process algebra, *concurrency* is expressed using so-called *merge operators*. As it turns out, the merge operators we introduced in Chapter 2 can be extended to discrete time in several distinct ways. In Chapter 5 we describe several process algebras with essentially different axiomatizations for these merge operators. We discuss the strong and the weak points of each axiomatization, and prove soundness and completeness properties for all concurrent process algebras we define.

Adding the Empty Process

In process algebra we have a special process called the *empty process*. This process is characterized by the fact that it cannot execute any action, but always has the option to *terminate successfully*. In other words: it is the process that does nothing, successfully. When we introduce the empty process, as defined in Chapter 2, in the context of

discrete-time process algebra, we find that its behavior is almost completely predetermined, due to the fact that we have some simple properties that we want to hold for all process algebras.

In Chapter 6 we look at an axiomatization for the empty process in discrete-time process algebra. We prove that a number of desirable properties hold for this axiomatization, and discuss the behavior of the empty process in discrete-time. Unfortunately, it turns out that the behavior of the empty process with respect to time steps is sometimes very counterintuitive.

Fischer's Protocol

In Chapter 7 we look at a protocol called *Fischer's Protocol for Mutual Exclusion*. This is a quite simple protocol whose correctness is nevertheless heavily dependent on subtle timing aspects. Therefore, it is very interesting to see how well suited discrete-time process algebra is to describe and verify this protocol. The degree to which this succeeds can be viewed as a measure of the maturity of discrete-time process algebra with respect to the verification of non-trivial protocols.

We specify the protocol using discrete-time process algebra, and then use process algebraic manipulations to bring this specification into a simplified format. After that, we need model-dependent reasoning to complete our verification. We conclude that discrete-time process algebra is indeed suitable to verify non-trivial protocols, but that there is still much room for improvement. The fact that the verification is partly model-dependent is disappointing, to say the least.

Related Work

In Chapter 8 we give an overview of other algebraic process formalisms that incorporate time. In this overview, we look at the approaches taken in a large number of publications from the literature. The formalisms that are closely related to our formalism, or that have interesting connections with it, are treated in some detail, but most are only briefly sketched. We explicitly do not try to provide a unifying framework to classify all formalisms encountered, and we only compare them to our own work, not to each other.

Samenvatting

Procesalgebra

Procesalgebra is een methode voor het *specificeren*, *verifiëren*, en in het algemeen, *praten over* computeralgoritmes en protocollen op een duidelijk gedefiniëerde, formele manier. Vergeleken met andere formalismes die voor dit doel ontwikkeld zijn, is het sterke punt van procesalgebra dat het gebaseerd is op *algebra*. Hierdoor worden manipulaties met processen *algebraïsche manipulaties*, waardoor de mogelijkheid bestaat om op hoog niveau abstracties te maken, waarmee potentiële complexiteitstoenames vermeden kunnen worden.

Het geheel van methodes en formalismes die gezamenlijk bekend staan als *procesalgebra* kan onderverdeeld worden in afzonderlijke delen die ieder, opnieuw, een *procesalgebra* genoemd worden.

De kern van iedere procesalgebra bestaat uit een verzameling *axioma's*. Ieder axioma bestaat uit een *gelijkheid* tussen twee *procestermen* (die vrije variabelen mogen bevatten). Achter ieder axioma schuilt een intuïtieve rechtvaardiging: een inzicht dat verklaart waarom deze twee procestermen als gelijk beschouwd dienen te worden. Gezamenlijk leiden deze axioma's tot een *algebra*: een wiskundige structuur waarmee termen gemanipuleerd kunnen worden.

Gegeven de verzameling axioma's van een zekere procesalgebra, is het mogelijk om een *model* te construeren: een wiskundige “wereld” die de gelijkheden eerbiedigt die gegeven worden door de axioma's. Zo'n model wordt een *semantiek* voor die procesalgebra genoemd. Over het algemeen kunnen meerdere, duidelijk verschillende, modellen gegeven worden voor een willekeurige gegeven procesalgebra. Echter, er bestaat de neiging om altijd hetzelfde soort model te gebruiken, dat een *bisimulatiemodel* genoemd wordt.

In Hoofdstuk 2 geven we een inleiding in de procesalgebra. We beginnen met het invoeren van een heel eenvoudige procesalgebra, BPA genaamd, en voegen dan geleidelijk meer mogelijkheden toe, zoals *vastzitten*¹, het *lege proces*², het *onmiddellijk vastzitten*³, de *vrije verwevingsoperator*⁴, en de *verwevingsoperator*⁵. Het hoofdstuk eindigt met een korte bespreking van de eigenschappen van procesalgebra's.

¹Ned. *vastzitten* = Eng. *deadlock*

²Ned. *lege proces* = Eng. *empty process*

³Ned. *onmiddellijk vastzitten* = Eng. *immediate deadlock*

⁴Ned. *vrije verwevingsoperator* = Eng. *free merge operator*

⁵Ned. *verwevingsoperator* = Eng. *merge operator*

Discrete Tijd

Oorspronkelijk ontbrak in de procesalgebra de mogelijkheid om kwantitatief over *tijd* te redeneren: alhoewel we de *volgorde* kunnen uitdrukken waarin acties plaatsvinden, kunnen we het *moment waarop* ze dat doen niet expliciet maken.

Men zou kunnen redeneren dat het gerechtvaardigd is om van tijdsaspecten te abstraheren, omdat tijdsaspecten vaak irrelevant zijn bij het verifiëren van systemen. Echter, soms hangt de correctheid van een protocol af van delicate tijdsaspecten, of zijn de tijdsaspecten zélf van een protocol het onderzoeksobject. In zulke gevallen kan de kwantitatieve analyse van tijdsaspecten niet vermeden worden, en hebben we weinig aan procesalgebra zonder tijd. We kunnen proberen om tijdsaspecten kunstmatig te coderen in procesalgebra zonder tijd, maar dat is dan wel een noodgreep; het is duidelijk dat procesalgebra zonder tijd hier niet voor gemaakt is.

Eén manier om tijd in te voeren is om de tijd onder te verdelen in een aftelbaar oneindig aantal *tijdsintervallen*⁶. Tussen deze tijdsintervallen vindt een *kloktik*⁷ (ook wel *tijdstap*⁸ genoemd) plaats, die expliciet wordt aangegeven. Dit leidt tot zogeheten *discrete tijd procesalgebra*⁹: het verstrijken van tijd wordt gemodelleerd alsof het *discreet* gebeurt, alsof het bestaat uit het verstrijken van een aantal verschillende tijdsintervallen.

In Hoofdstuk 3 geven we een inleiding in discrete tijd procesalgebra met *relatieve tijd*¹⁰, en voeren we enkele procesalgebra's in die discrete tijd uitbreidingen zijn van BPA.

Gezondheid en Volledigheid

Gegeven een zekere procesalgebra, onderscheiden we de *axioma's*, die een *algebraïsche equivalentierelatie* definiëren, en de *semantiek*, die een *model* van deze axioma's definieert. Wanneer het het geval is dat alle uitspraken die logisch volgen uit de axioma's inderdaad vervuld zijn in het model, dan zeggen we dat de procesalgebra de *gezondheidseigenschap*¹¹ bezit. Omgekeerd, wanneer alle uitspraken die in het model vervuld zijn logisch volgen uit de axioma's, dan zeggen we dat de procesalgebra de *volledigheidseigenschap*¹² bezit. Met andere woorden: *gezondheid* betekent dat de axioma's *altijd de waarheid vertellen* over het model, en *volledigheid* betekent dat de axioma's *alles kunnen vertellen* over het model.

Wanneer een procesalgebra zowel de gezondheid- als de volledigheidseigenschap bezit, zijn de axioma's en het model in perfecte harmonie: uitspraken die volgen uit de axioma's gelden in het model, en uitspraken die niet volgen uit de axioma's, gelden ook niet in het model. Dus, losjes gezegd, zijn de axioma's en het model equivalent.

In Hoofdstuk 4 bewijzen we deze twee eigenschappen, en een aantal andere, voor de procesalgebra's die gedefinieerd worden in Hoofdstuk 3.

⁶Ned. *tijdsintervallen* = Eng. *time-slices*

⁷Ned. *kloktik* = Eng. *clock tick*

⁸Ned. *tijdstap* = Eng. *time-step*

⁹Ned. *discrete tijd procesalgebra* = Eng. *discrete-time process algebra*

¹⁰Ned. *relatieve tijd* = Eng. *relative timing*

¹¹Ned. *gezondheidseigenschap* = Eng. *soundness property*

¹²Ned. *volledigheidseigenschap* = Eng. *completeness property*

Axioma's voor Coöperatie

Een sleutelaspect van de procesalgebra (en van iedere andere formele verificatiemethode) is het gebruik van *coöperatieve processen*¹³: processen die tegelijkertijd actief zijn, parallel worden uitgevoerd, en eventueel met elkaar communiceren, of elkaars gedrag beïnvloeden.

In de procesalgebra wordt *coöperatie*¹⁴ uitgedrukt door middel van zogeheten *verwevingsoperatoren*¹⁵. Naar het blijkt, kunnen de verwevingsoperatoren die we in Hoofdstuk 2 introduceerden op diverse verschillende manieren uitgebreid worden naar discrete tijd. In Hoofdstuk 5 beschrijven we diverse procesalgebra's met essentieel verschillende axiomatiseringen voor deze verwevingsoperatoren. We bespreken de sterke en de zwakke punten van iedere axiomatisering, en bewijzen gezondheids- en volledigheidseigenschappen voor alle coöperatieve procesalgebra's die we definiëren.

Toevoegen van het Lege Proces

In de procesalgebra hebben we een speciaal proces dat het *lege proces*¹⁶ wordt genoemd. Dit proces wordt gekarakteriseerd door het feit dat het geen enkele actie kan uitvoeren, maar wel altijd de mogelijkheid heeft om *succesvol te eindigen*¹⁷. Met andere woorden: het is het proces dat niets doet, op een succesvolle manier. Wanneer we het lege proces, zoals het gedefinieerd is in Hoofdstuk 2, invoeren in de context van discrete tijd procesalgebra, ontdekken we dat het gedrag hiervan al bijna volledig vastligt, op grond van het feit dat er enkele simpele eigenschappen bestaan waarvan we willen dat ze gelden voor alle procesalgebra's.

In Hoofdstuk 6 bekijken we een axiomatisering van het lege proces in discrete tijd procesalgebra. We bewijzen dat een aantal wenselijke eigenschappen gelden voor deze axiomatisering, en bespreken het gedrag van het lege proces in discrete tijd. Helaas blijkt dat het gedrag van het lege proces met betrekking tot tijdstappen soms erg tegenintuïtief is.

Fischer's Protocol

In Hoofdstuk 7 bekijken we een protocol dat *Fischer's Protocol voor Wederzijdse Uitsluiting*¹⁸ heet. Dit is een tamelijk eenvoudig protocol, waarvan de correctheid desalniettemin erg afhankelijk is van subtiele tijdsaspecten. Derhalve is het erg interessant om te zien hoe geschikt discrete tijd procesalgebra is om dit protocol te beschrijven en te verifiëren. De mate waarin dit lukt kan gezien worden als maatgevend voor de volwassenheid van discrete tijd procesalgebra met betrekking tot het verifiëren van niet-triviale protocollen.

¹³Ned. *coöperatieve processen* = Eng. *concurrent processes*

¹⁴Ned. *coöperatie* = Eng. *concurrency*

¹⁵Ned. *verwevingsoperatoren* = Eng. *merge operators*

¹⁶Ned. *lege proces* = Eng. *empty process*

¹⁷Ned. *succesvol te eindigen* = Eng. *to terminate successfully*

¹⁸Ned. *Fischer's Protocol voor Wederzijdse Uitsluiting* = Eng. *Fischer's Protocol for Mutual Exclusion*

We specificeren het protocol met behulp van discrete tijd procesalgebra, en gebruiken dan proces-algebraïsche manipulaties om deze specificatie in een vereenvoudigde vorm te brengen. Daarna hebben we een model-afhankelijke redenering nodig om onze verificatie te voltooien. We concluderen dat discrete tijd procesalgebra inderdaad geschikt is om niet-triviale protocollen te verifiëren, maar dat er nog steeds veel ruimte is voor verbeteringen. Het feit dat de verificatie gedeeltelijk model-afhankelijk is, is op zijn zachtst gezegd teleurstellend.

Gerelateerd Werk

In Hoofdstuk 8 geven we een overzicht van andere algebraïsche procesformalismes die tijd bevatten. In dit overzicht bekijken we de manier van aanpak in een groot aantal publicaties uit de literatuur. De formalismes die nauw verwant zijn met ons formalisme, of die interessante verbanden hiermee hebben, worden in enig detail behandeld, maar de meeste worden slechts kort geschetst. We proberen expliciet niet om een unificerend raamwerk te geven om alle formalismes die we tegen komen te classificeren, en we vergelijken ze alleen met ons eigen werk, niet met elkaar.

Acknowledgments

There are those who argue that theses should never contain acknowledgments. After all, the purpose of a thesis is to show that one can perform original scientific work, without external help, is it not? Furthermore, addressees will surely feel obliged to comment favorably on the acknowledger's work, thus compromising their independent judgement, will they not?

I think these objections are very much besides the point. First of all, I see no virtue in denying the involvement of others in my work. I would rather clearly indicate who was involved in what way, so that the originality of my work can be judged without guesswork on the part of the promotion committee. Secondly, as far as I know my colleagues, they are independent minds that have never had any hesitation whatsoever to point out the weak and the strong points in my work. I do not think they will change this attitude due to the mere fact that I have thanked them publicly. So here goes.

First of all, I would like to thank my *first promotor*, Jos Baeten, who was always willing to listen to my problems, and provide new insights whenever I got stuck. Furthermore, I thank him for giving me room for dissenting opinions, and showing patience with my progress when there was lack of it. Then, I thank Jozef Hooman, who was my daily supervisor during my first year, and Sjouke Mauw, who fulfilled that rôle during my last year. Also instrumental was Michel Reniers, my roommate for almost a year, and the co-author of the paper that led to Chapters 4 and 5 of this thesis. All four of these persons have been very important to me, and each one of them has at least once put me back on track after I had been seriously derailed in my research.

Theses are often too long and too technical to be pleasant reading, especially when they must be read cover to cover in barely a month. Nevertheless, three people have voluntarily done this. I thank Jan Bergstra, my *second promotor*, to whom I am also indebted for suggesting the title and core subject of my thesis; Scott Smolka, whom I also thank for making a long journey just to attend my thesis defense; and Alexander Ollongren, representative of my *Alma Mater*, whom I also thank for his willingness to be involved with both my Master's and Ph.D. education.

As any researcher knows, it is next to impossible to work efficiently when there is no one to serve as an audience for your ideas. Even the bare explaining of your work to a completely silent listener can do more for your own insight than a whole day of hard solitary work. I have had many listeners, and few of them were ever silent. I thank Twan Basten, Javier Blanco, Roland Bol, Arie van Deursen, Jan Friso Groote, Steven Klusener, Hans Mulder, and Chris Verhoef.

Then, there is the day-to-day routine of staying motivated. For this, you need people *outside* the group of your immediate colleagues, people with whom you can talk about subjects *outside* your work. I feel very lucky I had such people as roommates. I thank Dennis Dams and André Engels for the enjoyable company they have been, and for the far

too many hours we have spent talking just about everything *except* our ongoing research.

Not essential for productive work, but very desirable to keep enjoying it, is to be part of a research group in which people get along both socially and scientifically. I thank all members of the Formal Methods group I have not already mentioned: Roel Bloo, Tijn Borghuis, Alda Bouten, Thijs Cobben, Loe Feijs, Michael Franssen, Rob Gerth, Herman Geuvers, Kees Huizing, Jeroen Kleijn, Bart Knaack, Ruurd Kuiper, Twan Laan, Rob Nederpelt, Martijn Oostdijk, Peter Peters, Margriet Schetselaar, Pleun van der Steen, and Jan Zwanenburg.

There are a few people outside the University I like to thank for their interest in me and my work: Richard Kellermann Deibel, of SION, who was my contact with the organization that funded my research; Joost Engelfriet, whom I consider the most important teacher I have ever had; Hans van Dongen, Maarten van Dantzich, Gerard Borsboom, and Tycho Lamerigts, personal friends who showed a genuine interest in my progress; and finally, my parents, who have always encouraged me in my studies.

In a class completely of her own is Tieleke Stibbe, my wife. Against popular tradition, I have never neglected her by putting my work above her. And I am glad I never did, because the hours that I could have stolen from her, could never have compensated for the inspiration she has given me. Had I spent less time with her, my thesis would have been far from finished today.

Curriculum Vitae

The author was born in Vlaardingen, The Netherlands, on 25 august 1969, as the son of Frits Vereijken and Berth Baeten.

After he finished his secondary education at the Eckart College Eindhoven in 1987, he went to study Technical Computer Science at Eindhoven University of Technology, where in 1988 he completed his *propaedeuse* exam. Considering the Eindhoven curriculum too practical for his taste, he then went to Leiden University to study Theoretical Computer Science. There, in 1993, he completed his Master's degree under the supervision of Joost Engelfriet, with a thesis on *Graph Grammars* [190].

From 1993 to 1997, he did his Ph.D. work employed as an *onderzoeker in opleiding* in the Formal Methods research group of Jos Baeten at Eindhoven University of Technology, under a research grant from the Netherlands Organization for Scientific Research (NWO). During that period, he participated in the activities of the Institute for Programming Research and Algorithmics (IPA), the leading Dutch research school in formal methods. Currently, he is employed by *Lucent Technologies* at their research and development facility in Hilversum, The Netherlands.

Jan Joris Vereijken is married to Tieleke Stibbe; together they live in Amsterdam's colorful *Bijlmer* district.

Postal address:

Jan Joris Vereijken
Haardstee 105
NL-1102 NG Amsterdam Zuidoost
The Netherlands

Email address:

janjoris@acm.org

List of Tables

2.1	Axioms of BPA.	8
2.2	Left-distributivity.	9
2.3	Deduction rule for untimed actions.	10
2.4	Deduction rules for alternative and sequential composition.	11
2.5	Additional axioms for BPA_{δ}	14
2.6	Alternative for axiom for deadlock.	14
2.7	Additional axioms for BPA_{ϵ}	16
2.8	Deduction rules for untimed actions with empty process.	16
2.9	Deduction rules for alt. and seq. composition with empty process.	17
2.10	Additional axioms for BPA_{δ}	19
2.11	Deduction rules for immediate-deadlock predicate.	20
2.12	Additional axioms for PA.	21
2.13	Deduction rules for free merge.	23
2.14	Additional axioms for PA_{ϵ}	24
2.15	Deduction rules for free merge with empty process.	25
2.16	Additional axioms for PA_{δ}	26
2.17	Deduction rules for free merge with immediate deadlock.	27
2.18	Additional axioms for ACP.	29
2.19	Additional deduction rules for merge.	31
2.20	Deduction rules for encapsulation.	32
2.21	Additional axioms for ACP_{ϵ}	32
2.22	Additional deduction rules for merge with empty process.	33
2.23	Deduction rules for encapsulation with empty process.	33
2.24	Additional axioms for ACP_{δ}	34
2.25	Additional deduction rules for comm. merge with imm.-deadlock pred.	35
2.26	Additional deduction rules for encapsulation with imm.-deadlock pred.	35
3.1	Additional axioms for $BPA_{drt}^{-\delta}$	44
3.2	Deduction rules for untimed actions and time-unit delay.	44
3.3	Additional axioms for BPA_{drt}^{-ID}	47
3.4	Axioms for “now” operator.	47
3.5	Alternative axiom for undelayable deadlock.	48
3.6	Deduction rules for “now” operator.	48
3.7	Additional axioms for BPA_{drt}^{-}	52
3.8	Deduction rules for immediate-deadlock predicate.	53
3.9	Additional deduction rules for imm.-deadlock pred. and discrete time.	53
3.10	Additional axioms for BPA_{drt}	55
3.11	Recursive specification principle for unbounded start delay.	56
3.12	Deduction rules for delayable actions and unbounded start delay.	57

4.1	Term-Rewriting System for $BPA_{drt}^- - \delta$.	72
4.2	Term-Rewriting System for $BPA_{drt}^- - ID$.	79
4.3	Term-Rewriting System for BPA_{drt}^- .	86
4.4	Term-Rewriting System for BPA_{drt} .	92
4.5	Additional axioms for unbounded start delay.	102
5.1	Axioms for free merge.	106
5.2	Additional axioms for $PA_{drt}^- - ID$.	106
5.3	Deduction rules for free merge.	107
5.4	Additional axioms for $PA_{drt}^- - ID'$.	113
5.5	Additional term-rewriting rules for $PA_{drt}^- - ID'$.	115
5.6	Axioms for merge and encapsulation.	122
5.7	Additional axioms for $ACP_{drt}^- - ID$.	122
5.8	Additional deduction rules for merge.	123
5.9	Deduction rules for encapsulation.	123
5.10	Additional axioms for $ACP_{drt}^- - ID'$.	134
5.11	Additional term-rewriting rules for $ACP_{drt}^- - ID'$, Part I.	136
5.12	Additional term-rewriting rules for $ACP_{drt}^- - ID'$, Part II.	137
5.13	Additional term-rewriting rules for $ACP_{drt}^- - ID'$, Part III.	137
5.14	Additional axioms for PA_{drt} .	142
5.15	Deduction rules for free merge with immediate deadlock.	143
5.16	Additional axioms for unbounded start delay and left merge.	151
5.17	Additional axioms for ACP_{drt} .	153
5.18	Additional deduction rules for ACP_{drt} .	153
5.19	Axioms for communication merge and delayable actions.	168
5.20	Additional axioms for unbounded start delay.	169
6.1	Additional axioms for $BPA_{drt,\epsilon}^- - ID$.	179
6.2	Deduction rules for $BPA_{drt,\epsilon}^- - ID$.	180
6.3	Additional axioms for $PA_{drt,\epsilon}^- - ID$.	188
6.4	Additional deduction rules for $PA_{drt,\epsilon}^- - ID$.	191
6.5	Additional axioms for $ACP_{drt,\epsilon}^- - ID$.	199
6.6	Additional deduction rules for $ACP_{drt,\epsilon}^- - ID$.	200
6.7	Axioms for immediate deadlock with empty process.	202
6.8	Weakened axioms for undelayable deadlock with empty process.	202
6.9	Weakened axioms for empty process and sequential composition.	203
6.10	Additional axioms for $BPA_{drt,\epsilon} - ID$.	204
6.11	Recursive specification principle for delayable empty process.	205
6.12	Additional deduction rules for $BPA_{drt,\epsilon} - ID$.	206
7.1	Fischer's Protocol, first informal version.	215
7.2	Fischer's Protocol, second informal version.	216
7.3	Additional axioms for $ACP_{drt,\tau}$.	217
7.4	Additional deduction rules for $ACP_{drt,\tau}$.	218
7.5	Fischer's Protocol in $ACP_{drt,\tau}$.	220
7.6	The process graph of FP_{drt} .	225
7.7	The reduced process graph of $\tau_I(FP_{drt})$.	226

8.1	Axioms for the ATP merge and left merge operators.	242
8.2	Axioms for the ATP merge and left merge operators (interpreted).	243
B.1	Signatures of untimed process algebras.	262
B.2	Signatures of concrete discrete-time process algebras.	262
B.3	Signatures of abstract discrete-time process algebras.	263
C.1	Overview of definitions and theorems.	268

Bibliography

- [1] E. H. L. AARTS, *et al.*, editors. *Simplex Sigillum Veri*. Eindhoven University of Technology, 1995. *Liber Amicorum* dedicated to prof. dr. F. E. J. Kruseman Aretz.
- [2] M. ABADI, 1994. Personal communication.
- [3] M. ABADI AND L. LAMPORT. *An old-fashioned recipe for real time*. ACM Transactions on Programming Languages and Systems, **16**(5):1543–1571, 1994.
- [4] S. ABRAMSKY, D. M. GABBAY, AND T. S. E. MAIBAUM, editors. *Handbook of Logic in Computer Science*, volume 2: “Background: Computational Structures”. Oxford University Press, 1992.
- [5] S. ABRAMSKY, D. M. GABBAY, AND T. S. E. MAIBAUM, editors. *Handbook of Logic in Computer Science*, volume 4: “Semantic Modelling”. Oxford University Press, 1995.
- [6] L. ACETO AND A. JEFFREY. *A complete axiomatization of timed bisimulation for a class of timed regular behaviours*. Theoretical Computer Science, **152**(2):251–268, 1995.
- [7] L. ACETO AND D. MURPHY. *On the ill-timed but well-caused*. In [46], pages 97–111, 1994.
- [8] L. ACETO AND D. MURPHY. *Timing and causality in process algebra*. Acta Informatica, **33**(4):317–350, 1996.
- [9] ACM. *POPL ’87*. Association for Computing Machinery, 1987. Conference record of the Fourteenth Annual ACM Symposium on Principles of Programming Languages[®], Munich, West Germany, January 1987.
- [10] H. R. ANDERSEN AND M. MENDLER. *An asynchronous process algebra with multiple clocks*. In [182], pages 58–73, 1994.
- [11] J. C. M. BAETEN, editor. *Applications of Process Algebra*. Number 17 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.
- [12] J. C. M. BAETEN, 1996. Personal communication.
- [13] J. C. M. BAETEN AND J. A. BERGSTRA. *Global renaming operators in concrete process algebra*. Information and Computation, **78**(3):205–245, 1988.
- [14] J. C. M. BAETEN AND J. A. BERGSTRA. *Process algebra with a zero object*. In [33], pages 83–98, 1990.

- [15] J. C. M. BAETEN AND J. A. BERGSTRA. *Real space process algebra*. In [32], pages 96–110, 1991.
- [16] J. C. M. BAETEN AND J. A. BERGSTRA. *Real time process algebra*. *Formal Aspects of Computing*, 3(2):142–188, 1991.
- [17] J. C. M. BAETEN AND J. A. BERGSTRA. *Discrete time process algebra*. In [65], pages 401–420, 1992.
- [18] J. C. M. BAETEN AND J. A. BERGSTRA. *Real space process algebra*. *Formal Aspects of Computing*, 5(6):481–529, 1993.
- [19] J. C. M. BAETEN AND J. A. BERGSTRA. *On sequential composition, action prefixes and process prefix*. *Formal Aspects of Computing*, 6(3):250–268, 1994.
- [20] J. C. M. BAETEN AND J. A. BERGSTRA. *Process algebra with partial choice*. In [114], pages 465–480, 1994.
- [21] J. C. M. BAETEN AND J. A. BERGSTRA. *Discrete time process algebra with abstraction*. In [178], pages 1–15, 1995.
- [22] J. C. M. BAETEN AND J. A. BERGSTRA. *Real time process algebra with infinitesimals*. In [167], pages 148–187, 1995.
- [23] J. C. M. BAETEN AND J. A. BERGSTRA. *Some simple calculations in relative discrete time process algebra*. In [1], pages 67–74, 1995.
- [24] J. C. M. BAETEN AND J. A. BERGSTRA. *Discrete time process algebra*. *Formal Aspects of Computing*, 8(2):188–208, 1996.
- [25] J. C. M. BAETEN AND J. A. BERGSTRA. *Discrete time process algebra: Absolute time, relative time and parametric time*. *Fundamenta Informaticæ*, 29(1, 2):51–76, 1997.
- [26] J. C. M. BAETEN AND J. A. BERGSTRA. *Process algebra with propositional signals*. *Theoretical Computer Science*, 177(2):381–405, 1997.
- [27] J. C. M. BAETEN, J. A. BERGSTRA, AND R. N. BOL. *A real-time process logic*. Technical Report CSN 93/15, Eindhoven University of Technology, Computing Science Department, 1993.
- [28] J. C. M. BAETEN, J. A. BERGSTRA, AND R. N. BOL. *A real-time process logic*. In [81], pages 30–47, 1994. Extended abstract of [27].
- [29] J. C. M. BAETEN, J. A. BERGSTRA, AND M. A. RENIERS. *Discrete time process algebra with silent step*. In [165], 1998. To appear.
- [30] J. C. M. BAETEN, J. A. BERGSTRA, AND S. A. SMOLKA. *Axiomatizing probabilistic processes: ACP with generative probabilities*. *Information and Computation*, 121(2):234–255, 1995.
- [31] J. C. M. BAETEN AND R. J. VAN GLABBEEK. *Merge and termination in process algebra*. In [157], pages 153–172, 1987.

-
- [32] J. C. M. BAETEN AND J. F. GROOTE, editors. *CONCUR '91*. Number 527 in Lecture Notes in Computer Science. Springer-Verlag, 1991. Proceedings of CONCUR '91, 2nd International Conference on Concurrency Theory, Amsterdam, The Netherlands, August 1991.
- [33] J. C. M. BAETEN AND J. W. KLOP, editors. *CONCUR '90, Theories of Concurrency: Unification and Extension*. Number 458 in Lecture Notes in Computer Science. Springer-Verlag, 1990. Proceedings of CONCUR '90, Amsterdam, The Netherlands, August 1990.
- [34] J. C. M. BAETEN AND S. MAUW. *Delayed choice: an operator for joining Message Sequence Charts*. In [99], pages 340–354, 1995.
- [35] J. C. M. BAETEN AND M. A. RENIERS. *Discrete time process algebra with relative timing*. Unpublished lecture notes of Eindhoven University of Technology Course 2L920 “Process Algebra”.
- [36] J. C. M. BAETEN AND J. J. VEREIJKEN. *Discrete-time process algebra with empty process*. Technical Report CSR 97/05, Eindhoven University of Technology, Computing Science Department, 1997. Also appeared as Chapter 6 of [193].
- [37] J. C. M. BAETEN AND C. VERHOEF. *Concrete process algebra*. In [5], pages 149–268, 1995.
- [38] J. C. M. BAETEN AND W. P. WEIJLAND. *Process Algebra*. Number 18 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1990.
- [39] J. W. DE BAKKER, *et al.*, editors. *Real-Time: Theory in Practice*. Number 600 in Lecture Notes in Computer Science. Springer-Verlag, 1992. Proceedings of the REX Workshop, Mook, The Netherlands, June 1991.
- [40] T. BASTEN. *Branching bisimilarity is an equivalence indeed!* Information Processing Letters, **58**(3):141–147, 1996.
- [41] J. A. BERGSTRA, I. BETHKE, AND A. PONSE. *Process algebra with iteration and nesting*. The Computer Journal, **37**(4):243–258, 1994.
- [42] J. A. BERGSTRA, W. J. FOKKINK, AND C. A. MIDDELBURG. *Algebra of timed frames*. International Journal of Computer Mathematics, **61**(3–4):227–255, 1996.
- [43] J. A. BERGSTRA AND P. KLINT. *The discrete time TOOLBUS*. Technical Report P9502, University of Amsterdam, Programming Research Group, 1995.
- [44] J. A. BERGSTRA AND P. KLINT. *The discrete time TOOLBUS*. In [205], pages 286–305, 1996. Extended abstract of [43].
- [45] J. A. BERGSTRA AND J. W. KLOP. *Process algebra for synchronous communication*. Information and Control, **60**(1/3):109–137, 1984.
- [46] E. BEST, editor. *CONCUR '93*. Number 715 in Lecture Notes in Computer Science. Springer-Verlag, 1993. Proceedings of CONCUR '93, 4th International Conference on Concurrency Theory, Hildesheim, Germany, August 1993.

- [47] G. BIRTWISTLE, editor. *IV Higher Order Workshop*. Workshops in Computing. Springer-Verlag, 1991. Proceedings of the IV Higher Order Workshop, Banff, Alberta, Canada, September 1990.
- [48] J. O. BLANCO. *The State Operator in Process Algebra*. Ph.D. thesis, Eindhoven University of Technology, 1996.
- [49] G. VON BOCHMANN, R. DSSOULI, AND O. RAFIQ, editors. *Formal Description Techniques, VIII*. Chapman & Hall, 1996. Proceedings of FORTE '95, the IFIP TC6 Eight International Conference on Formal Description Techniques, Montreal, Canada, October 1995.
- [50] T. BOLOGNESI AND F. LUCIDI. *LOTOS-like process algebras with urgent or timed interactions*. In [161], pages 249–264, 1992.
- [51] T. BOLOGNESI AND F. LUCIDI. *Timed process algebras with urgent interactions and a unique powerful binary operator*. In [39], pages 124–146, 1992.
- [52] T. BOLOGNESI, F. LUCIDI, AND S. TRIGILA. *Converging towards a timed LOTOS standard*. Computer Standards & Interfaces, **16**(2):87–118, 1994.
- [53] S. H. J. BOS AND M. A. RENIERS. *The I²C-bus in discrete-time process algebra*. Science of Computer Programming, **29**(1-2):235–258, 1997.
- [54] P. BRÉMOND-GRÉGOIRE, I. LEE, AND R. GERBER. *ACSR: And algebra of communicating shared resources with dense time and priorities*. In [46], pages 417–431, 1993.
- [55] L. BRIM. *Modal logics in timed process algebras*. In [169], pages 13–26, 1993.
- [56] N. G. DE BRUIJN. *Additional comments on a problem in concurrent programming control*. Communications of the ACM, **10**(3):137–138, 1967.
- [57] J. BRYANS, J. DAVIES, AND S. SCHNEIDER. *Towards a denotational semantics for ET-LOTOS*. In [130], pages 269–283, 1995.
- [58] L. BUN. *ACP versus ATP and ATG*. Master's thesis, Eindhoven University of Technology, Department of Computing Science, 1993.
- [59] L. CARDELLI. *Real time agents*. In [155], pages 94–106, 1982.
- [60] K. ČERĀNS. *A calculus of timed refinement*. In [130], pages 516–530, 1995.
- [61] A. CERONE, *et al.* *Modelling a time-dependent protocol using the Circal process algebra*. In [135], pages 124–138, 1997.
- [62] L. CHEN. *Timed Processes: Models, Axioms and Decidability*. Ph.D. thesis, University of Edinburgh, 1992.
- [63] E. M. CLARKE AND J. M. WING. *Formal methods: state of the art and future directions*. ACM Computing Surveys, **28**(4):626–634, 1996.
- [64] R. CLEAVELAND, G. LÜTTGEN, AND M. MENDLER. *An algebraic theory of multiple clocks*. In [139], pages 166–180, 1997.

-
- [65] W. R. CLEAVELAND, editor. *CONCUR '92*. Number 630 in Lecture Notes in Computer Science. Springer-Verlag, 1992. Proceedings of CONCUR '92, Third International Conference on Concurrency Theory, Stony Brook, NY, USA, August 1992.
- [66] J. H. CONWAY. *On Numbers and Games*. Academic Press, 1976.
- [67] M. DANIELS. *Modelling real-time behavior with an interval time calculus*. In [199], pages 53–71, 1991.
- [68] P. R. D'ARGENIO AND S. MAUW. *Delayed choice for process algebra with abstraction*. In [130], pages 501–513, 1995.
- [69] J. DAVIES. *Specification and Proof in Real-Time CSP*. Distinguished Dissertations in Computer Science. Cambridge University Press, 1993.
- [70] J. DAVIES, *et al.* *Timed CSP: Theory and practice*. In [39], pages 640–675, 1992.
- [71] J. DAVIES, J. W. BRYANS, AND S. A. SCHNEIDER. *Real-time LOTOS and timed observations*. In [49], pages 383–397, 1996.
- [72] J. DAVIES AND S. SCHNEIDER. *Using CSP to verify a timed protocol over a fair medium*. In [65], pages 355–369, 1992.
- [73] J. DAVIES AND S. SCHNEIDER. *A brief history of Timed CSP*. Theoretical Computer Science, **138**(2):243–271, 1995.
- [74] J. W. DAVIES. *Specification and Proof in Real-Time Systems*. Ph.D. thesis, Oxford University, 1991. Also appeared as [69].
- [75] P. DEUSSEN, editor. *Theoretical Computer Science*. Number 104 in Lecture Notes in Computer Science. Springer-Verlag, 1981. Proceedings of the 5th GI-conference, Karlsruhe, West Germany, March 1981.
- [76] M. DIAZ AND R. GROZ, editors. *Formal Description Techniques, V*. North-Holland, 1993. Proceedings of FORTE '92, the IFIP TC6/WP6.1 Fifth International Conference on Formal Description Techniques for Distributed Systems and Communications Protocols, Perros-Guirec, France, October 1992.
- [77] E. W. DIJKSTRA. *Solution of a problem in concurrent programming control*. Communications of the ACM, **8**(9):569, 1965.
- [78] C. FIDGE. *A constraint oriented real-time process calculus*. In [76], pages 363–378, 1993.
- [79] M. FISCHER. *Re: where are you?* Electronic mail message from Michael Fischer to Leslie Lamport. Arpanet message sent on June 25, 1985 18:56:29 EDT, number 8506252257.AA07636@yale-bulldog.yale.arpa (47 lines).
- [80] W. J. FOKKINK. *Clocks, Trees and Stars in Process Theory*. Ph.D. thesis, University of Amsterdam, 1994.

- [81] D. M. GABBAY AND H. J. OHLBACH, editors. *Temporal Logic, First International Conference*. Number 827 in Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science). Springer-Verlag, 1994. Proceedings of ICTL' 94, Bonn, Germany, July 1994.
- [82] R. GERTH AND A. BOUCHER. *A timed failures model for extended communicating processes*. In [159], pages 95–114, 1987.
- [83] R. J. VAN GLABBEK. *Comparative Concurrency Semantics and Refinement of Actions*. Ph.D. thesis, Vrije Universiteit, Amsterdam, 1990. An extended abstract of Chapter 1 appeared as [84].
- [84] R. J. VAN GLABBEK. *The linear time - branching time spectrum*. In [33], pages 278–297, 1990.
- [85] J. CHR. GODSKESEN AND K. G. LARSEN. *Real-time calculi and expansion theorems*. In [169], pages 3–12, 1993.
- [86] J. F. GROOTE. *A new strategy for proving ω -completeness applied to process algebra*. In [33], pages 314–331, 1990.
- [87] J. F. GROOTE. *Specification and verification of real time systems in ACP*. In [132], pages 261–274, 1990. Extended abstract; full version appeared as Chapter 6 of [88].
- [88] J. F. GROOTE. *Process Algebra and Structured Operational Semantics*. Ph.D. thesis, University of Amsterdam, 1991.
- [89] J. F. GROOTE. *The syntax and semantics of timed μ CRL*. Technical Report SEN-R9709, Centrum voor Wiskunde en Informatica, Amsterdam, 1997.
- [90] J. F. GROOTE AND J. VAN DE POL. *A bounded retransmission protocol for large data packets*. In [205], pages 536–550, 1996.
- [91] J. F. GROOTE AND A. PONSE. *The syntax and semantics of μ CRL*. In [167], pages 26–62, 1995. Also appeared as Chapter 7 of [88].
- [92] H. HANSSON. *Modeling timeouts and unreliable media with a timed probabilistic calculus*. In [161], 1992. 67–82.
- [93] H. A. HANSSON. *Time and Probability in Formal Design of Distributed Systems*. Ph.D. thesis, Uppsala University, 1991.
- [94] M. HENNESSY. *Algebraic Theory of Processes*. MIT Press Series in the Foundations of Computing. MIT Press, 1988.
- [95] M. HENNESSY AND T. REGAN. *A temporal process algebra*. In [173], pages 33–48, 1991.
- [96] M. HENNESSY AND T. REGAN. *A process algebra for timed systems*. *Information and Computation*, **117**(2):221–239, 1995.

-
- [97] J. HILLEBRAND. *The ABP and the CABP—a comparison of performances in real time process algebra*. In [167], pages 124–147, 1995.
- [98] C. A. R. HOARE. *Communicating Sequential Processes*. International Series in Computer Science. Prentice Hall, 1985.
- [99] D. HOGREFE AND S. LEUE, editors. *Formal Description Techniques VII*. Chapman & Hall, 1995. Proceedings of FORTE '94, the 7th IFIP WG 6.1 International Conference on Formal Description Techniques, Berne, Switzerland, October 1994.
- [100] C. HUIZING, R. GERTH, AND W. P. DE ROEVER. *Full abstraction of a real-time denotational semantics for an OCCAM-like language*. In [9], pages 223–237, 1987.
- [101] ISO. *Information Processing Systems – Open Systems Interconnection – LOTOS – A Formal Description Technique Based on the Temporal Ordering of Observational Behavior. ISO 8807*. ISO, Geneva, 1989.
- [102] ITU-T. *ITU-T Recommendation Z.100: Specification and Description Language (SDL)*. ITU-T, Geneva, 1988.
- [103] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, Geneva, 1993.
- [104] ITU-TS. *ITU-TS Recommendation Z.120 Annex B: Algebraic semantics of Message Sequence Charts*. ITU-TS, Geneva, 1995.
- [105] ITU-TS. *ITU-TS Draft Recommendation Z.120: Message Sequence Chart 1996 (MSC96)*. ITU-TS, Geneva, 1996.
- [106] ITU-TS. *ITU-TS Recommendation Z.120 Annex C: Static Semantics of Message Sequence Charts*. ITU-TS, Geneva, 1996.
- [107] W. JANSSEN, *et al.* *Layering of real-time distributed processes*. In [125], pages 393–417, 1994.
- [108] A. JEFFREY. *Abstract timed observation and process algebra*. In [32], pages 332–345, 1991.
- [109] A. JEFFREY. *Discrete Timed CSP*. Technical Report PMG Memo 78, Chalmers University of Technology, Department of Computer Sciences, 1991.
- [110] A. JEFFREY. *Timed process algebra \neq time \times process algebra*. Technical Report PMG Memo 79, Chalmers University of Technology, Department of Computer Sciences, 1991.
- [111] A. JEFFREY. *A linear time process algebra*. In [126], pages 433–442, 1992.
- [112] A. S. JEFFREY. *Observation Spaces and Timed Processes*. Ph.D. thesis, Oxford University, 1992.
- [113] G. JONES. *A Timed Model for Communicating Processes*. Ph.D. thesis, Oxford University, 1982.

- [114] B. JONSSON AND J. PARROW, editors. *CONCUR '94: Concurrency Theory*. Number 836 in Lecture Notes in Computer Science. Springer-Verlag, 1994. Proceedings of CONCUR '94, 5th International Conference, Uppsala, Sweden, August 1994.
- [115] J. J. T. KLEIJN. *Modelling and Analysing Industrial Systems: χ and Process Algebra, a Combined Approach*. Master's thesis, Eindhoven University of Technology, Department of Mechanical Engineering, 1997.
- [116] J. W. KLOP. *Term rewriting systems*. In [4], pages 1-116, 1992.
- [117] A. S. KLUSENER. *Models and Axioms for a Fragment of Real Time Process Algebra*. Ph.D. thesis, Eindhoven University of Technology, 1993.
- [118] D. E. KNUTH. *Additional comments on a problem in concurrent programming control*. Communications of the ACM, **9**(5):321-322, 1966.
- [119] D. E. KNUTH. *Surreal Numbers*. Addison-Wesley, 1974.
- [120] H. P. KORVER. *Protocol Verification in μ CRL*. Ph.D. thesis, University of Amsterdam, 1994.
- [121] L. KOTT, editor. *Automata, Languages and Programming*. Number 226 in Lecture Notes in Computer Science. Springer-Verlag, 1986. Proceedings of ICALP '86, 13th International Colloquium, Rennes, France, July 1986.
- [122] C. P. J. KOYMANS AND J. L. M. VRANCKEN. *Extending process algebra with the empty process ε* . Technical Report LGPS 1, Utrecht University, Department of Philosophy, 1985.
- [123] L. LAMPORT. *A new solution of Dijkstra's concurrent programming problem*. Communications of the ACM, **17**(8):453-455, 1974.
- [124] L. LAMPORT. *A fast mutual exclusion algorithm*. ACM Transactions on Computer Systems, **5**(1):1-11, 1987.
- [125] H. LANGMAACK, W.-P. DE ROEVER, AND J. VYTOPIL, editors. *Formal Techniques in Real-Time and Fault-Tolerant Systems*. Number 863 in Lecture Notes in Computer Science. Springer-Verlag, 1994. Proceedings of the Third International Symposium Organized Jointly with the Working Group Provably Correct Systems - ProCos, Lübeck, Germany, September 1994.
- [126] K. G. LARSEN AND A. SKOU, editors. *Computer Aided Verification*. Number 575 in Lecture Notes in Computer Science. Springer-Verlag, 1992. Proceedings of CAV '91, 3rd International Workshop, Aalborg, Denmark, July 1991.
- [127] J. LEACH ALBERT, B. MONIEN, AND M. RODRÍGUEZ ARTALEJO, editors. *Automata, Languages and Programming*. Number 510 in Lecture Notes in Computer Science. Springer-Verlag, 1991. Proceedings of ICALP '91, 18th International Colloquium, Madrid, Spain, July 1991.
- [128] G. LEDUC. *An upward compatible timed extension to LOTOS*. In [161], pages 217-232, 1992.

-
- [129] G. LEDUC AND L. LÉONARD. *A timed LOTOS supporting a dense time domain and including new timed operators*. In [76], pages 87–102, 1993.
- [130] I. LEE AND S. SMOLKA, editors. *CONCUR '95: Concurrency Theory*. Number 962 in Lecture Notes in Computer Science. Springer-Verlag, 1995. Proceedings of CONCUR '95, 6th International Conference, Philadelphia, PA, USA, August 1995.
- [131] L. LÉONARD AND G. LEDUC. *An enhanced version of timed LOTOS and its application to a case study*. In [187], pages 483–498, 1994.
- [132] L. LOGRIppo, R. L. PROBERT, AND H. URAL, editors. *Protocol Specification, Testing, and Verification, X*. North-Holland, 1990. Proceedings of the IFIP WG 6.1 tenth international symposium on protocol specification, testing, and verification, Ottawa, Ontario, Canada, June 1990.
- [133] G. LOWE. *Probabilities and Priorities in Timed CSP*. Ph.D. thesis, Oxford University, 1993.
- [134] G. LOWE. *Probabilistic and prioritized models of Timed CSP*. Theoretical Computer Science, **138**(2):315–352, 1995.
- [135] O. MAHLER, editor. *Hybrid and Real-Time Systems*. Number 1201 in Lecture Notes in Computer Science. Springer-Verlag, 1997. Proceedings of the International Workshop HART '97, Grenoble, France, March 1997.
- [136] S. MAUW. *A Process Specification Formalism*. Ph.D. thesis, University of Amsterdam, 1991.
- [137] S. MAUW, 1997. Personal communication.
- [138] S. MAUW AND G. J. VELTINK, editors. *Algebraic Specification of Communication Protocols*. Number 36 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1993.
- [139] A. MAZURKIEWICZ AND J. WINKOWSKI, editors. *CONCUR '97: Concurrency Theory*. Number 1243 in Lecture Notes in Computer Science. Springer-Verlag, 1997. Proceedings of CONCUR '97, 8th International Conference, Warsaw, Poland, July 1997.
- [140] C. MIGUEL, A. FERNÁNDEZ, AND L. VIDALLER. *Extending LOTOS towards performance evaluation*. In [76], pages 103–117, 1993.
- [141] G. J. MILNE. *CIRCAL and the representation of communication, concurrency, and time*. ACM Transactions on Programming Languages and Systems, **7**(2):270–298, 1985.
- [142] R. MILNER. *Communication and Concurrency*. International Series in Computer Science. Prentice Hall, 1989.
- [143] F. MOLLER. *The importance of the left merge operator in process algebras*. In [162], pages 752–764, 1990.
- [144] F. MOLLER. *Process algebra as a tool for real time analysis*. In [47], pages 32–53, 1991.

- [145] F. MOLLER AND C. TOFTS. *A temporal calculus of communicating systems*. In [33], pages 401–415, 1990.
- [146] U. MONTANARI AND V. SASSONE, editors. *CONCUR '96: Concurrency Theory*. Number 1119 in Lecture Notes in Computer Science. Springer-Verlag, 1996. Proceedings of CONCUR '96, 7th International Conference, Pisa, Italy, August 1996.
- [147] J. M. MORRIS AND R. C. SHAW, editors. *4th Refinement Workshop*. Workshops in Computing. Springer-Verlag, 1991. Proceedings of the 4th Refinement Workshop, Cambridge, UK, January 1991.
- [148] J. C. MULDER. *Case Studies in Process Specification and Verification*. Ph.D. thesis, University of Amsterdam, 1990.
- [149] D. MURPHY. *Timed process algebra, Petri nets, and event refinement*. In [147], pages 456–478, 1991.
- [150] D. J. V. MURPHY. *Time, Causality, and Concurrency*. Ph.D. thesis, University of Surrey, 1989.
- [151] A. NAKATA, T. HIGASHINO, AND K. TANIGUCHI. *LOTOS enhancement to specify time constraint among non-adjacent actions using first order logic*. In [187], pages 451–466, 1994.
- [152] P. G. NEUMANN. *Computer-related risks*. Addison-Wesley, 1995.
- [153] X. NICOLLIN AND J. SIFAKIS. *An overview and synthesis on timed process algebras*. In [126], pages 376–398, 1992.
- [154] X. NICOLLIN AND J. SIFAKIS. *The algebra of timed processes ATP: Theory and application*. Information and Computation, **114**(1):131–178, 1994.
- [155] M. NIELSEN AND E. M. SCHMIDT, editors. *Automata, Languages and Programming*. Number 140 in Lecture Notes in Computer Science. Springer-Verlag, 1982. Proceedings of ICALP 82, 9th International Colloquium on Automata, Languages and Programming, Aarhus, Denmark, July 1982.
- [156] E. R. NIEUWLAND. *Proving mutual exclusion with process algebra*. In [11], pages 45–51, 1990.
- [157] K. V. NORI, editor. *Foundations of Software Technology and Theoretical Computer Science*. Number 287 in Lecture Notes in Computer Science. Springer-Verlag, 1987. Proceedings of the 7th conference on the foundations of software technology and theoretical computer science, Pune, India, December 1987.
- [158] E.-R. OLDEROG, editor. *Programming Concepts, Methods and Calculi*, volume A-56 of *IFIP Transactions A: Computer Science and Technology*. North-Holland, 1994. Proceedings of the IFIP TC2/WG2.1/WG2.2/WG2.3 Working Conference (PROCOMET '94), San Miniato, Italy, June 1994.
- [159] T. OTTMANN, editor. *Automata, Languages and Programming*. Number 267 in Lecture Notes in Computer Science. Springer-Verlag, 1987. Proceedings of the 14th International Colloquium, Karlsruhe, West Germany, July 1987.


-
- [160] D. PARK. *Concurrency and automata on infinite sequences*. In [75], pages 167–183, 1981.
- [161] K. R. PARKER AND G. A. ROSE, editors. *Formal Description Techniques, IV*. North-Holland, 1992. Proceedings of FORTE '91, the IFIP TC6/WG6.1 Fourth International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, Sydney, Australia, November 1991.
- [162] M. S. PATERSON, editor. *Automata, Languages and Programming*. Number 443 in Lecture Notes in Computer Science. Springer-Verlag, 1990. Proceedings of ICALP '90, 17th International Colloquium, Warwick University, England, July 1990.
- [163] I. PHILIPS. *Refusal testing*. Theoretical Computer Science, **50**(3):241–284, 1987.
- [164] J.-L. PICARD AND W. T. RIKER. *On Q-continuous operators in subspace process algebra*. Acta Transwarpa Federatia, **1701**(D):11–42, 2371.
- [165] G. PLOTKIN, C. STIRLING, AND M. TOFTE, editors. *Proof, Language and Interaction: Essays in Honour of Robin Milner*. MIT Press, 1998. To appear.
- [166] G. D. PLOTKIN. *A structural approach to operational semantics*. Technical Report DAIMI FN-19, Århus University, Computer Science Department, 1981.
- [167] A. PONSE, C. VERHOEF, AND S. F. M. VAN VLIJMEN, editors. *Algebra of Communicating Processes*. Workshops in Computing. Springer-Verlag, 1995. Proceedings of ACP94, the First Workshop on the Algebra of Communicating Processes, Utrecht, The Netherlands, May 1994.
- [168] A. PONSE, C. VERHOEF, AND S. F. M. VAN VLIJMEN, editors. *De Proceedings: ACP' 95*. Number CSR 95/14 in Computing Science Reports. Eindhoven University of Technology, Computing Science Department, 1995.
- [169] S. PURUSHOTHAMAN AND A. ZWARICO, editors. *NAPAW 92*. Workshops in Computing. Springer-Verlag, 1993. Proceedings of the First North American Process Algebra Workshop, Stony Brook, New York, USA, August 1992.
- [170] J. QUEMADA, A. AZCORRA, AND D. DE FRUTOS. *TIC: a timed calculus for LOTOS*. In [198], pages 195–209, 1990.
- [171] J. QUEMADA AND A. FERNANDEZ. *Introduction of quantitative relative time into LOTOS*. In [181], pages 105–121, 1987.
- [172] J. QUEMADA, D. DE FRUTOS, AND A. AZCORRA. *TIC: A Timed Calculus*. Formal Aspects of Computing, **5**(3):224–252, 1993.
- [173] J. QUEMADA, J. MAÑAS, AND E. VÁZQUEZ, editors. *Formal Description Techniques, III*. North-Holland, 1991. Proceedings of FORTE '90, the IFIP TC/WG 6.1 Third International Conference on Formal Description Techniques for Distributed Systems and Communications Protocols, Madrid, Spain, November 1990.
- [174] G. M. REED. *A Uniform Mathematical Theory for Real-Time Distributed Computing*. Ph.D. thesis, Oxford University, 1988.

- [175] G. M. REED AND A. W. ROSCOE. *A timed model for communicating sequential processes*. In [121], pages 314–323, 1986.
- [176] G. M. REED AND A. W. ROSCOE. *A timed model for communicating sequential processes*. *Theoretical Computer Science*, **58**(1–3):249–261, 1988.
- [177] G. M. REED, A. W. ROSCOE, AND S. A. SCHNEIDER. *CSP and timewise refinement*. In [147], pages 258–280, 1991.
- [178] H. REICHEL, editor. *FCT '95, International Conference on Fundamentals of Computation Theory*. Number 965 in *Lecture Notes in Computer Science*. Springer-Verlag, 1995. Proceedings of FCT '95, 10th International Conference, Dresden, Germany, August 1995.
- [179] M. A. RENIERS. *Message Sequence Chart: Syntax and Semantics*. Ph.D. thesis, Eindhoven University of Technology, 1998. In preparation; title tentative.
- [180] M. A. RENIERS AND J. J. VEREIJKEN. *Completeness in discrete-time process algebra*. Technical Report CSR 96/15, Eindhoven University of Technology, Computing Science Department, 1996. Parts also appeared as Chapters 4 and 5 of [193].
- [181] H. RUDIN AND C. H. WEST, editors. *Protocol Specification, Testing, and Verification, VII*. North-Holland, 1987. Proceedings of the IFIP WG 6.1 Seventh International Conference on Protocol Specification, Testing, and Verification, Zürich, Switzerland, May 1987.
- [182] D. SANNELLA, editor. *Programming Languages and Systems – ESOP '94*. Number 788 in *Lecture Notes in Computer Science*. Springer-Verlag, 1994. Proceedings of ESOP '94, 5th European Symposium on Programming, Edinburgh, U.K., April 1994.
- [183] F. SCHNEIDER, 1994. Personal communication.
- [184] F. SCHNEIDER, B. BLOOM, AND K. MARZULLO. *Putting time into proof outlines*. In [39], pages 618–639, 1992.
- [185] S. A. SCHNEIDER. *Correctness and Communication in Real-Time Systems*. Ph.D. thesis, Oxford University, 1990.
- [186] S. A. SCHNEIDER. *An operational semantics for Timed CSP*. *Information and Computation*, **116**(2):193–213, 1995.
- [187] R. L. TENNEY, P. D. AMER, AND M. Ü. UYAR, editors. *Formal Description Techniques, VI*. North-Holland, 1994. Proceedings of FORTE '93, the IFIP TC6/WG6.1 Sixth International Conference on Formal Description Techniques, Boston, MA, USA, October 1993.
- [188] F. W. VAANDRAGER. *Algebraic Techniques for Concurrency and their Applications*. Ph.D. thesis, University of Amsterdam, 1990.
- [189] G. J. VELTINK. *Tools for PSF*. Ph.D. thesis, University of Amsterdam, 1995.
- [190] J. J. VEREIJKEN. *Graph Grammars and Operations on Graphs*. Master's thesis, Leiden University, 1993.

-
- [191] J. J. VEREIJKEN. *Fischer's protocol in timed process algebra*. In [168], pages 245–283, 1995. Parts also appeared as Chapter 7 of [193].
- [192] J. J. VEREIJKEN. *A process algebra for hybrid systems*. Extended abstract of talk presented at The Second European Workshop on Real-Time and Hybrid Systems, Grenoble, France, May 1995.
- [193] J. J. VEREIJKEN. *Discrete-Time Process Algebra*. Ph.D. thesis, Eindhoven University of Technology, 1997.
- [194] C. VERHOEF. *A congruence theorem for structured operational semantics with predicates and negative premises*. In [114], pages 433–448, 1994.
- [195] C. VERHOEF. *A general conservative extension theorem in process algebra*. In [158], pages 149–168, 1994.
- [196] J. L. M. VRANCKEN. *Studies in Process Algebra, Algebraic Specifications and Parallelism*. Ph.D. thesis, University of Amsterdam, 1991.
- [197] J. L. M. VRANCKEN. *The algebra of communicating processes with empty process*. *Theoretical Computer Science*, 177(2):287–328, 1997. Also appeared as Chapter 1 of [196].
- [198] S. T. VUONG, editor. *Formal Description Techniques, II*. North-Holland, 1990. Proceedings of FORTE '89, the IFIP TC/WG 6.1 Second International Conference on Formal Description Techniques for Distributed Systems and Communications Protocols, Vancouver, Canada, December 1989.
- [199] J. VYTOPIĽ, editor. *Formal Techniques in Real-Time and Fault-Tolerant Systems*. Number 571 in Lecture Notes in Computer Science. Springer-Verlag, 1991. Proceedings of the Second International Symposium, Nijmegen, The Netherlands, January 1992.
- [200] J. J. VAN WAMEL. *Verification Techniques for Elementary Data Types and Retransmission Protocols*. Ph.D. thesis, University of Amsterdam, 1995.
- [201] Y. WANG. *Real-time behaviour of asynchronous agents*. In [33], pages 502–520, 1990.
- [202] Y. WANG. *A Calculus of Real-Time Systems*. Ph.D. thesis, Chalmers University of Technology, 1991.
- [203] Y. WANG. *CCS + time = an interleaving model for real time systems*. In [127], pages 217–228, 1991.
- [204] Y. WANG. *Algebraic reasoning for real-time probabilistic processes with uncertain information*. In [125], pages 680–693, 1994.
- [205] M. WIRSING AND M. NIVAT, editors. *Algebraic Methodology and Software Technology*. Number 1101 in Lecture Notes in Computer Science. Springer-Verlag, 1996. Proceedings of AMAST '96, 5th International Conference, Munich, Germany, July 1996.

Symbol Index

$\tau_I(x)$ abstraction216-218	$x \sim_p y$ bisimulation equivalence 11
a action symbol 7 action, delayable ... 54, 55, 57, 168, 204, 206 action, untimed 10, 15, 16	$x \sim y$ shorthand for $x \sim_p$ 11
$ats(a)$ action, delayable 55	\times Cartesian product 269
\underline{a} action, undelayable 44, 180	$C(P)$ set of closed terms 8
$cts(a)$ action, undelayable 42	$\gamma(a, b)$ communication function28
A alphabet7	$x + y$ composition, alternative 7-9, 11, 17, 20, 44, 53, 180
A_δ alphabet plus δ7	\circ composition, of relations 269
$A_{\delta\varepsilon}$ alphabet plus δ and ε 7	$x \cdot y$ composition, sequential 7-9, 11, 17, 20, 44, 53, 180
$A_{\delta\tau}$ alphabet plus δ and τ7	δ deadlock, delayable 55 deadlock, untimed 13, 14, 19
A_σ alphabet plus σ 7	$\dot{\delta}$ deadlock, immediate 19, 20, 26, 34, 53, 142, 153, 202
A_τ alphabet plus τ7	$\underline{\underline{\delta}}$ deadlock, undelayable ..47, 48, 179, 202
$x := y$ assignment 215	\vdash derivability relation 9
$\langle S \rangle$ atomic execution 215	

- \subseteq
 embedding relation 38
- ε
 empty process, delayable ..204–206
 empty process, untimed . 16, 24, 32
- $\dot{\varepsilon}$
 empty process, immediate203
- $\underline{\varepsilon}$
 empty process, undelayable ... 179,
 180, 188, 199, 203
- $\partial_H(x)$
 encapsulation ...28, 29, 32–35, 122,
 123, 153, 168, 169, 199, 200
- end of proof 270
- =
 equality, algebraic 9
 equality, in test 215
- $x \stackrel{\text{def}}{=} y$
 equivalence by definition 269
- $[x]_{\sim}$
 equivalence class12
- \otimes
 generic operator 270
- 
 helping hand 5
- $ID(x)$
 immediate deadlock (predicate) .20,
 27, 35, 53, 143, 153, 218
- \perp
 inconsistent state178
- \otimes_n
 indexed operator 270
- ∞
 infinity 215
- $\sigma_{\text{rel}}^*(x)$
 iterated delay 55
- \succ_{lpo}
 lexicographical path ordering .. 270
- $x \parallel y$
 merge 28, 29, 31, 33, 122, 123, 199,
 200
 merge, free 21, 23, 25, 27, 106, 107,
 143, 188, 191
- $x \mid y$
 merge, communication 29, 31–
 35, 122, 123, 134, 153, 168, 169,
 199, 200
- $x \ll y$
 merge, left ... 21–28, 106, 107, 113,
 142, 143, 151, 188, 191
- \models
 modeling relation 11
- \mathbb{N}
 natural numbers 269
- \mathbb{N}^+
 natural numbers, positive 269
- $\nu_{\text{rel}}(x)$
 “now” operator (rel.) .47, 48, 52, 53,
 72, 179, 180
- $\nu(x)$
 shorthand for $\nu_{\text{rel}}(x)$ 72
- $n(x)$
 number of symbols .. 46, 50, 54, 60,
 107, 114, 123, 135, 182
- $<$
 ordering, well-founded 270
- \mathbb{R}
 real numbers 269
- $\mathbb{R}^{\geq 0}$
 real numbers, non-negative 269

P^+	recursion principle superscript .. 56	$\xrightarrow{\sigma}$	refl., trans. closure of $\xrightarrow{\sigma}$ 94
R^R	reflexive closure 270	\subseteq	subset relation 269
\in	set membership 269	\oplus	sum of term-deduction systems 112
$\underline{\tau}$	silent action, undelayable .213, 216, 217	$\sum_{i \in I} t_i$	summation 49
τ	silent action, delayable 217 silent action, untimed 67	R^S	symmetric closure 270
\otimes^*	starred operator 270	\equiv	syntactical equivalence 269
$x \xrightarrow{a} y$	step, a 10, 11, 16, 17, 23, 25, 27, 31-33, 44, 48, 57, 107, 123, 143, 180, 191, 200, 206, 218	$T(P)$	term-deduction system 10
$x \xrightarrow{a_1, \dots, a_n} x'$	step, a (multiple) 10	\rightarrow	term-rewriting step 270
$x \xrightarrow{a}$	step, no a 10	\surd	termination operator 26
$x \leftrightarrow$	step, none 10	$x \downarrow$	termination option .. 16, 17, 25, 33, 180, 191, 200, 206
$x \xrightarrow{\tau} y$	step, silent 218	$x \dagger$	termination option, no 16
$\xrightarrow{\tau}$	refl., trans. closure of $\xrightarrow{\tau}$ 218	$\underline{\underline{\sigma}}$	time-unit delay process ... 179, 202
$x \xrightarrow{a} \surd$	step, terminating 10, 11, 31, 32, 44, 48, 57, 107, 123, 218	$\sigma_{\text{abs}}(x)$	time-unit delay operator (abs.) .. 72
$x \xrightarrow{\sigma} y$	step, time 44, 53, 57, 107, 123, 143, 180, 191, 200, 206, 218	$\sigma_{\text{rel}}(x)$	time-unit delay operator (rel.) ... 42, 44, 52, 53, 72
$x \xrightarrow{\sigma}$	step, no time 44	$\sigma(x)$	shorthand for $\sigma_{\text{rel}}(x)$ 72
		R^T	transitive closure 270
		0	true zero 178

$[x]^\omega$

unbounded start delay . 54-57, 102,
151, 169

Axiom Index

A

A18, 68, 74
A28, 74
A38, 74-76
A48, 74, 75, 189
A58, 75, 179
A614, 20, 48, 179, 202
A6A14, 20
A6ID19, 52, 87, 202
A714, 47, 179
A7ID19, 87, 202
A816
A916
ATPCM1243
ATPLM1243
ATPLM2243
ATPLM3243
ATPLM4243
ATPLM5243
ATS55, 95

C

CF29, 56, 121
CM129, 121
CM229, 32, 121
CM329, 32, 121
CM429, 121
CM529, 121
CM629, 121

D

D129, 30, 56, 121
D229, 30, 56, 121
D329, 30, 121
D429, 30, 121
D532
D634, 153
DA204

DCS147, 81, 180
DCS247, 81, 180
DCS347, 48, 81, 180
DCS447, 82, 180
DCSE1179, 180
DCSE2179, 180
DCSE3179, 180
DCSE4179, 180
DCSID52, 88
DEP204
DRT144, 52, 76, 179
DRT244, 52, 76, 179
DRT347, 52, 81
DRT447, 48, 81, 173
DRT4A48, 173
DRT547, 48, 52, 81, 173
DRTB1217
DRTB2217
DRTB3217
DRTB4217
DRTCF121, 122, 130, 157
DRTCM1121, 122, 127, 243
DRTCM2121, 122, 128, 157, 174
DRTCM3121, 122, 128, 157, 174
DRTCM4121, 122, 128, 157
DRTCM5122, 129
DRTCM6 .. 122, 129, 134, 140, 153, 173
DRTCM6ID153, 165, 173
DRTCM7122, 129, 134, 140, 153
DRTCM7ID153, 166
DRTCM8134, 139
DRTCM9134, 139
DRTCM10134, 139
DRTCM11134, 139
DRTCM12121, 122, 129
DRTCM13121, 122, 130
DRTD1121, 122, 131, 157
DRTD2121, 122, 131, 157

DRTD3 121, 122, 131
 DRTD4 121, 122, 132
 DRTD5 122, 133
 DRTD6 153, 166
 DRTE1 179
 DRTE2 179
 DRTE3 179, 203
 DRTE3ID 203
 DRTE4 179, 203
 DRTE4ID 203
 DRTE5 202
 DRTE6 202
 DRTE7 202
 DRTECF 198, 199
 DRTECM1 198, 199
 DRTECM2 198, 199
 DRTECM3 198, 199
 DRTECM4 198, 199
 DRTECM5 198, 199
 DRTECM6 198, 199
 DRTECM7 198, 199
 DRTECM8 198, 199
 DRTECM9 198, 199
 DRTED1 198, 199
 DRTED2 198, 199
 DRTED3 198, 199
 DRTED4 198, 199
 DRTED5 198, 199
 DRTED6 198, 199
 DRTEM1 187, 188
 DRTEM2 187, 188
 DRTEM3 187, 188
 DRTEM4 187-189
 DRTEM5 187-189
 DRTEM6 187-189
 DRTEM7 187-189
 DRTEM8 187, 188
 DRTEM9 187, 188
 DRTEM10 187, 188, 190
 DRTEM11 187, 188
 DRTEM12 187-189
 DRTESID 202
 DRTM1 106, 109, 187
 DRTM2 106, 110, 142, 174, 243
 DRTM2ID 142, 149, 157
 DRTM3 106, 110, 142, 174, 187, 243
 DRTM3ID 142, 149

DRTM4 106, 110, 187, 243
 DRTM5 ... 106, 109, 111, 118, 142, 174,
 243
 DRTM5ID 142, 149
 DRTM6 ... 106, 109, 111, 119, 174, 243
 DRTM7 113, 114, 117, 174
 DRTM8 113, 114, 117, 174
 DRTM9 113, 118, 174
 DRTM10 113, 118, 174
 DRTM11 113, 118, 174
 DRTMID1 142, 150
 DRTMID2 142, 150
 DRTMID3 153, 166
 DRTMID4 153, 166
 DRTSID 51-53, 87
 DRTT1 217
 DRTT2 217
 DRTT3 217
 DRTT4 217
 DRTT5 217
 DRTT6 217

L

LD 8, 9, 25

M

M1 21, 22, 29, 106
 M2 21-23, 25-27, 32, 52, 106
 M2ID 26, 27, 142
 M3 21-23, 25-27, 106
 M3ID 26, 27, 142
 M4 21, 22, 25, 106
 ME1 24, 25
 ME2 24, 25
 ME3 24, 25
 ME4 32
 ME5 32
 MID1 26, 27, 34, 142
 MID2 26, 27, 34, 142
 MID3 34, 153
 MID4 34, 153

T

TF 179

U

USD 55, 95
 USD1 102, 174

USD2	102, 174
USD3	102, 174
USD4	102, 174
USD5	102, 174
USD6	151, 174
USD7	151, 174
USD8	169, 170
USD9	169-171
USDCF	167-169, 174
USDCM2	167-169, 174
USDCM3	167-169, 174
USDCM4	167-169, 174
USDD1	167-169
USDD2	167-169

Person Index

A

Abadi, M. 213, 227
Aceto, L. 236, 245
Andersen, H. R. 246
Azcorra, A. 239

B

Baeten, H. J. M. 282, 283
Baeten, J. C. M. .. iv, vii, viii, 5-233, 246,
253, 255, 281, 283
Basten, T. 46, 281
Bergstra, J. A. iv, viii, 3, 5-234, 246, 253,
281
Bethke, I. 39
Blanco, J. O. 13, 39, 281
Bloo, C. J. 282
Bloom, B. 213, 227
Bol, R. N. 228, 253, 281
Bolognesi, T. 239, 247
Borghuis, V. A. J. 282
Borsboom, G. J. J. M. 282
Bos, S. H. J. 42, 48, 177
Boucher, A. 238
Bouten, A. 282
Brémond-Grégoire, P. 245
Brim, L. 247
Bruijn, N. G. de 213
Bryans, J. 240
Bun, L. 248

C

Cardelli, L. 246
Čerāns, K. 245
Cerone, A. 41
Chen, L. 236
Clarke, E. M. 2
Cleaveland, R. 246
Cobben, J. M. H. 282

Conway, J. H. 232

D

D'Argenio, P. R. 227
Dams, D. R. 281
Daniels, M. 237
Dantzich, M. R. van 282
Davies, J. W. 238, 240
Deursen, A. van 281
Dijkstra, E. W. 213, 214
Dongen, H. P. A. van 282

E

Einstein, A. 233
Engelfriet, J. vii, 282, 283
Engels, A. 281

F

Feijs, L. M. G. 282
Fernández, A. 239, 240
Fidge, C. 237
Fischer, M. 41, 213
Fokkink, W. J. 39, 41, 45, 234
Franssen, M. 282
Frutos, D. de 239

G

Gerber, R. 245
Gerth, R. T. 238, 282
Geuvers, J. H. 282
Glabbeek, R. J. van 9, 177, 189
Godskesen, J. Chr. 247
Groote, J. F. 10, 41, 43, 67, 229, 234,
252, 253, 281

H

Hansson, H. A. 236, 248
Hennessy, M. 39, 42, 244
Higashino, T. 241

Hillebrand, J. 41
 Hoare, C. A. R. 39, 238
 Hooman, J. J. M. 281
 Huizing, C. 238, 282

J

Janssen, W. 213
 Jeffrey, A. S. 236-238, 246, 247
 Jones, G. 238

K

Kellerman Deibel, R. J. 282
 Kleijn, J. J. T. 55, 282
 Klint, P. 246
 Klop, J. W. 3, 5, 68, 229
 Klusener, A. S. 41, 233, 281
 Knaack, B. T. 282
 Knuth, D. E. 213, 232
 Korver, H. P. 253
 Koymans, C. P. J. 177
 Kuiper, R. 282

L

Laan, T. D. L. 282
 Lamerigts, T. T. H. 282
 Lamport, L. 213
 Larsen, K. G. 247
 Leduc, G. 239, 240
 Lee, I. 245
 Léonard, L. 240
 Liang, C. *see* Chen, L.
 Lowe, G. 238
 Lucidi, F. 239, 247
 Lüttgen, G. 246

M

Marzullo, K. 213, 227
 Mauw, S. iv, 227, 252, 281
 Mendler, M. 246
 Middelburg, C. A. 45, 234
 Miguel, C. 240
 Milne, G. J. 41
 Milner, R. 39, 235
 Moller, F. 21, 22, 43, 235
 Mulder, J. C. 281
 Murphy, D. V. J. 245

N

Nakata, A. 241
 Nederpelt, R. P. 282
 Neumann, P. G. 1
 Newton, I. 233
 Nicollin, X. 43, 60, 106, 241, 247, 271
 Nieuwland, E. R. 220

O

Ollongren, A. 281
 Oostdijk, M. D. 282

P

Park, D. 11
 Peters, P. J. F. 282
 Philips, I. 42
 Picard, J.-L. 299
 Plotkin, G. D. 10, 229
 Pol, J. C. van de 41
 Ponse, A. 39, 234, 253

Q

Quemada, J. 239

R

Reed, G. M. 238
 Regan, T. 42, 244
 Rem, M. iii
 Reniers, M. A. viii, 41-228, 281
 Riker, W. T. 299
 Roever, W.-P. de 238
 Roscoe, A. W. 238

S

Schetselaar, M. M. 282
 Schneider, F. 213, 227
 Schneider, S. A. 238, 240
 Sifakis, J. 43, 60, 106, 241, 247, 271
 Smolka, S. A. 18, 281
 Steen, P. van der 282
 Stibbe, M. M. J. E. v, 282, 283

T

Taniguchi 241
 Tofts, C. 43, 235
 Trigila, S. 247

V

- Veltink, G. J.252
Vereijken, G. J. J.282, 283
Vereijken, J. J. iii-viii, 174, 177, 226, 283
Verhoef, C. 5, 12, 39, 41-212, 281
Vidaller, L. 240
Vrancken, J. L. M. 26, 177, 189

W

- Wang, Y. 235, 236
Weijland, W. P. .. 5-39, 50, 56, 178, 217,
227, 229, 255
Wing, J. M. 2

X

- X *see* Verhoef, C.

Y

- Yi, W. *see* Wang, Y.

Z

- Zwanenburg, J.282

Index

A	
absolute time	42, 72, 230, 231
abstract process algebra	39
abstraction	39, 216
axiom nomenclature for	255
axioms for	217
deduction rules for	218
ACP	
axioms of	29
basic terms of	32
cola machines in	31
semantics of	31
signature of	29
ACP _δ	
axioms of	34
basic terms of	35
semantics of	34
signature of	34
ACP _{drt}	
axioms of	152
basic terms of	153
semantics of	153
signature of	152
ACP _{drt,ε-ID}	
axioms of	209
basic terms of	210
semantics of	209
signature of	209
time determinism for	209
ACP _{drt,ε} ⁺ -ID	
axioms of standard concurrency in 210	
completeness of	210
elimination for	210
soundness of	210
ACP _{drt,ε} ⁻ -ID	
axioms of	198
axioms of standard concurrency in 201	
basic terms of	200
completeness of	200
elimination of	200
semantics of	198
signature of	198
soundness of	200
time determinism for	198
ACP _{drt} ⁻ -ID	
axioms of	121
basic terms of	123
completeness of	133, 134
conservativity of	133
elimination for	124, 126, 127
ground equivalence involving ..	139, 140
semantics of	122
signature of	121
soundness of	127, 133
ACP _{drt} ⁻ -ID'	
axioms of	134
basic terms of	135
completeness of	141
elimination for	135, 138, 139
ground equivalence involving ..	139, 140
semantics of	134
signature of	134
soundness of	141
ACP' _{drt}	
axioms of	167
basic terms of	168
completeness of	168
elimination for	168
semantics of	168
signature of	168
soundness of	168

- ACP''_{drt}
 axioms of 169
 basic terms of 169
 completeness of 172
 elimination for 170
 semantics of 169
 signature of 169
 soundness of 172
- ACP^+_{drt}
 completeness of 167
 conservativity of 166
 elimination for 159, 165
 soundness of 165, 166
- $ACP_{drt,\tau}$
 axioms of 217
 bisimulation for 218, 219
 completeness of 219
 elimination for 219
 semantics of 218
 signature of 217
 soundness of 219
- ACP_ε
 axioms of 32
 basic terms of 34
 semantics of 33
 signature of 32
- ACP's (class) 28
- ACSR 245
- action step
 deduction rules for 10, 11,
 16, 17, 23, 25, 27, 31-33, 44, 48,
 57, 107, 123, 143, 180, 191, 200,
 206, 218
- action transition *see* action step
- actions 6, 7
 delayable *see* delayable actions
 expired 231
 immediate 233
 internal 216
 multi- 233
 undelayable *see* undelayable actions
 untimed *see* untimed actions
 urgent 233, 234
- algebraic advantage
 absence of 227
- alphabet 7
- Alternating Bit Protocol 227, 238
- alternative composition 7
 associativity of 8
 axioms for 8, 9, 44
 behaving deterministically 14
 behaving non-deterministically .. 14
 commutativity of 8
 deduction rules for .. 11, 17, 20, 44,
 53, 180
 distributivity of 8, 22, 24, 29, 48, 58,
 102
 idempotency of 8
 zero element of 14, 19, 20
- alternative normal form
 for $BPA^-_{drt,\varepsilon}$ -ID 184
 equivalent to basic terms 184
 induction over 186
- ANF *see* alternative normal form
- APA 246
- associativity
 of alternative composition 8
 of free merge 22, 198, 209
 of merge 30, 202, 211
 of sequential composition 8
- atomicity 6
- ATP 241-244
- auxiliary operators 21
- axiom nomenclature
 for abstraction 255
 for communication function ... 255
 for communication merge 255
 for empty process 255
 for encapsulation 255
 for free merge 255
 for immediate deadlock 255
 for merge 255
 for "now" operator 255
 for unbounded start delay 255
- axiom schemes 56
- axioms 6
 for abstraction 217
 of ACP 29
 of ACP_δ 34
 of ACP_{drt} 152
 of $ACP_{drt,\varepsilon}$ -ID 209
 of $ACP^-_{drt,\varepsilon}$ -ID 198
 of ACP^-_{drt} -ID 121
 of ACP^-_{drt} -ID' 134

of ACP'_{drt}	167
of ACP''_{drt}	169
of $ACP_{drt,\tau}$	217
of ACP_{ϵ}	32
for alternative composition .8, 9, 44	
of BPA.....	8
of BPA_{δ}	14
of BPA_{δ}	19
of $BPA_{\delta\epsilon}$	18
of BPA_{drt}	55
of $BPA_{drt,\epsilon}$ -ID.....	204
of $BPA_{drt,\epsilon}$ -ID.....	179
of BPA_{drt}	51
of $BPA_{drt}-\delta$	44
of BPA_{drt} -ID.....	47
of BPA'_{drt}	102
of BPA_{ϵ}	16
for communication merge ...29, 32, 34, 122, 134, 153, 168, 169, 199	
conditional	56
for delayable actions .. 55, 168, 204	
for delayable empty process ...204, 205	
for delayable silent action	217
for encapsulation ...29, 32, 34, 122, 153, 168, 169, 199	
for free merge	21, 106, 188
for immediate deadlock . 19, 26, 34, 142, 153, 202	
for left merge . 21, 24, 26, 106, 113, 142, 151, 188	
for left merge (ATP)	242
for merge	29, 122, 199
for merge (ATP)	242
nomenclature of	255
for “now” operator	47, 52, 179
overview of	256
of PA.....	21
of PA_{δ}	23
of PA_{δ}	26
of PA_{drt}	142
of $PA_{drt,\epsilon}$ -ID.....	207
of $PA_{drt,\epsilon}$ -ID.....	187
of PA_{drt} -ID.....	106
of PA_{drt} -ID'.....	113
of PA'_{drt}	151
of PA_{ϵ}	24

for sequential composition . 8, 9, 44	
of standard concurrency <i>see</i> axioms of standard concurrency	
for time-unit delay constant ... 179, 202	
for time-unit delay operator . 44, 52	
for unbounded start delay .. 55, 56, 102, 151, 169	
for undelayable deadlock47, 48, 179, 202	
for undelayable empty process 179, 188, 199, 203	
for undelayable silent action ... 217	
for untimed deadlock	14, 19
for untimed empty process . 16, 24, 32	
axioms of standard concurrency .37, 38	
in $ACP'_{drt,\epsilon}$ -ID.....	210
in $ACP_{drt,\epsilon}$ -ID.....	201
in discrete-time process algebra 172	
in $PA'_{drt,\epsilon}$ -ID.....	209
in $PA_{drt,\epsilon}$ -ID.....	195
in untimed process algebra	37

B

basic terms

of ACP.....	32
of ACP_{δ}	35
of ACP_{drt}	153
of $ACP_{drt,\epsilon}$ -ID.....	210
of $ACP_{drt,\epsilon}$ -ID.....	200
of ACP_{drt} -ID.....	123
of ACP_{drt} -ID'.....	135
of ACP'_{drt}	168
of ACP''_{drt}	169
of ACP_{ϵ}	34
of BPA.....	13
of BPA_{δ}	15
of BPA_{δ}	20
of $BPA_{\delta\epsilon}$	19
of BPA_{drt}	57
of $BPA_{drt,\epsilon}$ -ID.....	206
of $BPA_{drt,\epsilon}$ -ID.....	182
of BPA_{drt}	53
of $BPA_{drt}-\delta$	46
of BPA_{drt} -ID.....	49
of BPA'_{drt}	102

- of BPA_ε 17
- deadlock-free 13
- $(\delta, \underline{\delta})$ - 20
- (δ, ε) - 19
- different from literature 15
- $(\underline{\sigma}, \underline{\delta}, \underline{\varepsilon})$ - 182
- elimination to 36
- ε - 17
- equivalent to ANF terms 184
- named misleadingly 13
- of PA 23
- of PA_δ 24
- of PA_δ 28
- of PA_{drt} 143
- of $PA_{drt,\varepsilon}$ -ID 208
- of $PA_{drt,\varepsilon}^-$ -ID 192
- of PA_{drt}^- -ID 107
- of PA_{drt}^- -ID' 114
- of PA'_{drt} 151
- of PA_ε 25
- σ - 46
- $(\sigma, \underline{\delta})$ - 49
- $(\sigma, \underline{\delta}, \delta, \underline{\delta})$ - 57
- $(\sigma, \underline{\delta}, \underline{\delta})$ - 53
- standard 15
- Bijlmer 283
- bisimulation
 - for $ACP_{drt,\tau}$ 218, 219
 - for BPA 11
 - for BPA_δ 20
 - for $BPA_{drt,\varepsilon}^-$ -ID 182
 - for BPA_{drt}^- 53
 - for BPA_{drt}^- - δ 45
 - for BPA_ε 17
 - branching tail 218
 - is a congruence 12, 35
 - σ - 234
 - time factorization w.r.t. 45, 234
- σ -bisimulation *see under S*
- bisimulation model 6, 12
- bisimulation semantics 9
- BPA
 - axioms of 8
 - basic terms of 13
 - bisimulation for 11
 - cola machines in 9
 - semantics of 10
 - signature of 7
- BPA_δ
 - axioms of 14
 - basic terms of 15
 - cola machines in 14
 - semantics of 15
 - signature of 13
- BPA_δ
 - axioms of 19
 - basic terms of 20
 - bisimulation for 20
 - semantics of 20
 - signature of 19
- $BPA_{\delta\varepsilon}$
 - axioms of 18
 - basic terms of 19
 - cola machines in 18
 - semantics of 18
 - signature of 18
- BPA_{drt}
 - axioms of 55
 - basic terms of 57
 - general form of basic terms of ... 58
 - recursion principles for 55
 - semantics of 57
 - signature of 54
- $BPA_{drt,\varepsilon}$ -ID
 - axioms of 204
 - basic terms of 206
 - general form of basic terms of . 206
 - semantics of 205
 - signature of 204
 - time determinism for 206
- $BPA_{drt,\varepsilon}^+$ -ID
 - completeness of 207
 - elimination for 206
 - soundness of 207
- $BPA_{drt,\varepsilon}^-$
 - absence of 202
- $BPA_{drt,\varepsilon}^-$ -ID
 - alternative normal form for 184
 - axioms of 179
 - basic terms of 182
 - bisimulation for 182
 - completeness of 186
 - elimination for 186
 - representation of terms of 183

- semantics of 180
 signature of 179
 soundness of 186
 time determinism for 181
- BPA_{drt}^-
 axioms of 51
 basic terms of 53
 bisimulation for 53
 completeness of 90, 91
 elimination for 85, 87
 semantics of 52
 signature of 51
 soundness of 87, 88
- $BPA_{drt}^- - \delta$
 axioms of 44
 basic terms of 46
 bisimulation for 45
 completeness of 78
 elimination for 72, 73
 semantics of 44
 signature of 42
 soundness of 73, 76
- $BPA_{drt}^- - ID$
 axioms of 47
 basic terms of 49
 completeness of 84, 85
 elimination for 79, 80
 general form of basic terms of ... 49
 representation of terms of 50
 semantics of 48
 signature of 47
 soundness of 80, 82
- BPA_{drt}'
 axioms of 102
 basic terms of 102
 completeness of 103
 elimination for 103
 semantics of 102
 signature of 102
 soundness of 103
- BPA_{drt}^+
 completeness of 100, 102
 elimination for 91, 94
 representation of terms of ... 61, 64
 soundness of 94, 96
- BPA_ϵ
 axioms of 16
 basic terms of 17
 bisimulation for 17
 semantics of 16
 signature of 16
- BPA 's (class) 6
 branching tail bisimulation 218
 branching-time semantics 9
-
- ## C
-
- Cartesian product 269
 catastrophic deadlock 271
 CCS 39, 235–237
 with Interval Time 237
 Linear Timed 237
 Temporal 235, 239
 Timed (Chen) 236
 Timed (Wang) 235–236
 Timed Probabilistic 236
 CCSiT *see* CCS, with Interval Time
 choice 7
 cIPA 245
 CIRCAL 41
 classical space-time 233
 clock tick 41
 closed terms 8
 closure
 reflexive 270
 symmetric 270
 transitive 270
 cola machines
 in ACP 31
 in BPA 9
 in BPA_δ 14
 in $BPA_{\delta\epsilon}$ 18
 cola-machine₁ 9
 cola-machine₂ 10
 cola-machine₃ 10
 cola-machine₄ 15
 cola-machine₅ 18
 cola-machine₆ 22
 cola-machine₇ 27
 cola-machine₈ 31
 cola-machine₉ 31
 in PA 22
 in PA_δ 27
 communication 28
 communication function 28

axiom nomenclature for 255
 empty 38
 communication merge
 axiom nomenclature for 255
 axioms for 29, 32, 34, 122, 134, 153,
 168, 169, 199
 deduction rules for . 31, 33, 35, 123,
 153, 200
 distributivity of .. 29, 122, 169, 198
 commutativity
 of alternative composition 8
 of encapsulation 122, 169
 of free merge 22, 198, 209
 of merge 30, 202, 211
 of time-unit delay operator 122
 of unbounded start delay 169
 completeness 37
 of $ACP_{drt,\varepsilon}^+$ -ID 210
 of $ACP_{drt,\varepsilon}^-$ -ID 200
 of ACP_{drt}^- -ID 133, 134
 of ACP_{drt}^- -ID' 141
 of ACP'_{drt} 168
 of ACP''_{drt} 172
 of ACP_{drt}^+ 167
 of $ACP_{drt,\tau}$ 219
 of $BPA_{drt,\varepsilon}^+$ -ID 207
 of $BPA_{drt,\varepsilon}^-$ -ID 186
 of BPA_{drt}^- 90, 91
 of BPA_{drt}^- - δ 78
 of BPA_{drt}^- -ID 84, 85
 of BPA'_{drt} 103
 of BPA_{drt}^+ 100, 102
 by ground equivalence 120
 for open terms 67
 of $PA_{drt,\varepsilon}^+$ -ID 208
 of $PA_{drt,\varepsilon}^-$ -ID 192
 of PA_{drt}^- -ID 112, 113
 of PA_{drt}^- -ID' 121
 of PA'_{drt} 152
 of PA_{drt}^+ 151
 proof outlines for 70
 of real-time process algebra 234
 sufficient condition for 76
 in untimed process algebra 37
 ω -completeness *see under O*
 composition

alternative *see alternative*
 composition
 parallel *see merge*
 of relations 269
 sequential *see sequential*
 composition
 concrete process algebra 39
 CONCUR 247
 concurrent process algebra 28
 condition operator 247
 conditional axioms 56
 congruence 35
 bisimulation is a 12
 conservativity 39
 of ACP_{drt}^- -ID 133
 of ACP_{drt}^+ 166
 operational 112
 of PA_{drt}^- -ID 112
 of PA_{drt}^+ 150
 Coq 252
 μ CRL *see under M*
 CSA 246
 CSP 39, 238
 Discrete Timed 238
 Timed 238
 CTR 245

D

DCSP *see CSP, Discrete Timed*
 deadlock
 catastrophic *see catastrophic*
 deadlock
 immediate . *see immediate deadlock*
 untimed *see untimed deadlock*
 deadlock freedom *see basic terms*
 deadlock process 13
 deadlock-like processes 178
 deduction rules 10
 for abstraction 218
 for action step 10, 11,
 16, 17, 23, 25, 27, 31-33, 44, 48,
 57, 107, 123, 143, 180, 191, 200,
 206, 218
 for alternative composition . 11, 17,
 20, 44, 53, 180
 for communication merge ... 31, 33,
 35, 123, 153, 200

for delayable actions	57, 206	$T(\text{BPA})$	10
for delayable empty process ...	206	$T(\text{BPA}_\delta)$	15
for encapsulation ...	32, 33, 35, 123, 153, 200	$T(\text{BPA}_\delta)$	20
for free merge .	23, 25, 27, 107, 143, 191	$T(\text{BPA}_{\delta\epsilon})$	18
for immediate deadlock	20, 53	$T(\text{BPA}_{\text{drt}})$	57
for immediate deadlock (predicate)	20, 27, 35, 53, 143, 153, 218	$T(\text{BPA}_{\text{drt},\epsilon\text{-ID}})$	205
for left merge .	23, 25, 27, 107, 143, 191	$T(\text{BPA}_{\text{drt},\epsilon\text{-ID}}^-)$	180
for merge	31, 33, 123, 200	$T(\text{BPA}_{\text{drt}}^-)$	52
for “now” operator	48, 53, 180	$T(\text{BPA}_{\text{drt}}^- - \delta)$	44
in <i>panth</i> format	12	$T(\text{BPA}_{\text{drt}}^- \text{-ID})$	48
in <i>path</i> format	112	$T(\text{BPA}'_{\text{drt}})$	102
purity of	112	$T(\text{BPA}_\epsilon)$	16
for sequential composition ..	11, 17, 20, 44, 53, 180	$T(\text{PA})$	23
for silent action	218	$T(\text{PA}_\delta)$	24
stratification of	12	$T(\text{PA}_\delta)$	27
for terminating step .	10, 11, 31, 32, 44, 48, 57, 107, 123, 218	$T(\text{PA}_{\text{drt}})$	142
for termination option ..	16, 17, 25, 33, 180, 191, 200, 206	$T(\text{PA}_{\text{drt},\epsilon\text{-ID}})$	207
for time step ..	44, 53, 57, 107, 123, 143, 180, 191, 200, 206, 218	$T(\text{PA}_{\text{drt},\epsilon\text{-ID}}^-)$	190
for time-unit delay constant	180	$T(\text{PA}_{\text{drt}}^- \text{-ID})$	107
for time-unit delay operator .	44, 53	$T(\text{PA}_{\text{drt}}^- \text{-ID}')$	114
for unbounded start delay	57	$T(\text{PA}'_{\text{drt}})$	151
for undelayable actions	44, 180	$T(\text{PA}_\epsilon)$	25
for untimed actions	10, 16	delayable actions	54
for untimed deadlock	15	axioms for	55, 168, 204
for untimed empty process	16	deduction rules for	57, 206
well-foundedness of	12, 112	delayable deadlock	55
deduction systems		delayable empty process	204
sum of	112	axioms for	204, 205
$T(\text{ACP})$	31	deduction rules for	206
$T(\text{ACP}_\delta)$	34	delayable silent action	
$T(\text{ACP}_{\text{drt}})$	153	axioms for	217
$T(\text{ACP}_{\text{drt},\epsilon\text{-ID}})$	209	derivability relation	9
$T(\text{ACP}_{\text{drt},\epsilon\text{-ID}}^-)$	198	discrete-time empty process	174,
$T(\text{ACP}_{\text{drt}}^- \text{-ID})$	122	177–212, 230	
$T(\text{ACP}_{\text{drt}}^- \text{-ID}')$	134	design goals for	178
$T(\text{ACP}'_{\text{drt}})$	168	discrete-time process algebra ...	<i>passim</i>
$T(\text{ACP}'_{\text{drt}})$	169	axioms of standard concurrency in	172
$T(\text{ACP}_{\text{drt},\tau})$	218	embeddings for	65, 172, 211
$T(\text{ACP}_\epsilon)$	33	distributivity	9
		of alternative composition	8, 22, 24,
		29, 48, 58, 102	
		of communication merge ..	29, 122,
		169, 198	
		of encapsulation	30
		of free merge	22
		of left merge	22, 24

- of “now” operator 48
of sequential composition 8, 58, 102
of time-unit delay constant 198
of time-unit delay operator 122
of unbounded start delay .. 58, 102,
169
-
- ## E
-
- elimination
for $ACP_{drt,\epsilon}^+$ -ID 210
of $ACP_{drt,\epsilon}^-$ -ID 200
for ACP_{drt}^- -ID 124, 126, 127
for ACP_{drt}^- -ID' 135, 138, 139
for ACP_{drt}' 168
for ACP_{drt}'' 170
for ACP_{drt}^+ 159, 165
for $ACP_{drt,\tau}$ 219
to basic terms 36
for $BPA_{drt,\epsilon}^+$ -ID 206
for $BPA_{drt,\epsilon}^-$ -ID 186
for BPA_{drt}^- 85, 87
for $BPA_{drt}-\delta$ 72, 73
for BPA_{drt}^- -ID 79, 80
for BPA_{drt}' 103
for BPA_{drt}^+ 91, 94
for $PA_{drt,\epsilon}^+$ -ID 208
for $PA_{drt,\epsilon}^-$ -ID 192
for PA_{drt}^- -ID 107, 109
for PA_{drt}^- -ID' 114, 116, 117
for PA_{drt}' 152
for PA_{drt}^+ 145, 149
to process algebra 36
proof outlines for 67
in untimed process algebra 36
- embedding relation 38
embeddings 38
for discrete-time process algebra 65,
172, 211
for untimed process algebra 38
- empty process
axiom nomenclature for 255
combined with immediate deadlock
21, 202
delayable *see* delayable empty
process
discrete-time *see* discrete-time
empty process
- immediate *see* immediate empty
process
undelayable . *see* undelayable empty
process
untimed *see* untimed empty process
- encapsulation
axiom nomenclature for 255
axioms for 29, 32, 34, 122, 153, 168,
169, 199
commutativity of 122, 169
deduction rules for . 32, 33, 35, 123,
153, 200
distributivity of 30
- equational logic 9
ET-LOTOS 240
expansion theorems 247
-
- ## F
-
- failure semantics 227
Fischer’s Protocol ... vii, 4, 41, 213-228,
234
Floyd-Hoare logic 213
formal methods 2
frames 234
free merge
associativity of 22, 198, 209
axiom nomenclature for 255
axioms for 21, 106, 188
commutativity of 22, 198, 209
deduction rules for . 23, 25, 27, 107,
143, 191
distributivity of 22
unit element of 25, 192, 208
- full time stop 271
-
- ## G
-
- general form
of basic terms of BPA_{drt} 58
of basic terms of $BPA_{drt,\epsilon}$ -ID 206
of basic terms of BPA_{drt}^- -ID 49
- generic operator 270
graph model 230
ground equivalence 117
involving ACP_{drt}^- -ID 139, 140
involving ACP_{drt}^- -ID' 139, 140
axiom 117
completeness by 120

-
- deduction-system117
 involving PA_{drt}^-ID 117, 120
 involving PA_{drt}^-ID' 117, 120
 soundness by 120
-
- H**
-
- Halmos270
-
- I**
-
- idempotency
 of alternative composition 8
 iff 269
 immediate actions 233
 immediate deadlock19
 axiom nomenclature for 255
 axioms for 19, 26, 34, 142, 153, 202
 combined with empty process .. 21,
 202
 deduction rules for 20, 53
 justification for 231
 not allowed here 27
 immediate deadlock (predicate)
 deduction rules for .. 20, 27, 35, 53,
 143, 153, 218
 immediate empty process 203
 immediate time stop271
 inconsistent state 178
 infinitesimals 232
 integration 232, 233
 prefix 234
 unrestricted232
 internal actions 216
 IPA (process algebra)245
 IPA (research school) iv, 283
 iterated delay 55, 244
 iteration39, 247
-
- L**
-
- left merge
 axioms for 21, 24, 26, 106, 113, 142,
 151, 188
 deduction rules for .23, 25, 27, 107,
 143, 191
 distributivity of 22, 24
 raison d'être of22
 unit element of25, 33, 192, 200,
 208, 210
 zero element of . 192, 200, 208, 210
 left merge (ATP)
 axioms for 242
 lexicographical path ordering 68
 linear-time semantics8
 livelock271
 LOTOS 238-241, 248
 LOTOS-T 240
 LOTOS/T 241
 LTCCS *see* CCS, Linear Timed
 ℓ TCCS235
 Lucent Technologies 283
-
- M**
-
- maximal progress 43, 245
 merge
 associativity of 30, 202, 211
 axiom nomenclature for 255
 axioms for 29, 122, 199
 communication .*see* communication
 merge
 commutativity of 30, 202, 211
 deduction rules for 31, 33, 123, 200
 free *see* free merge
 left *see* left merge
 unit element of 33, 200, 210
 merge (ATP)
 axioms for 242
 ν TTCS323
 modeling relation 11
 models6
 bisimulation 6, 12
 graph 230
 junk in 37
 one-point13
 other 13
 projective-limit 13, 230
 term 13
 MSC 177, 212
 μ CRL 234, 253
 timed 234-235, 252
 multi-actions233
-
- N**
-
- natural numbers 269
 nomenclature
 of axioms 255

of process algebras 261
 non-standard real numbers 232
 notation regarding semantics 10, 16, 44
 “now” operator
 axiom nomenclature for 255
 axioms for 47, 52, 179
 deduction rules for 48, 53, 180
 distributivity of 48
 shorthand for 72
 NWO iv, 283

O

ω -completeness 67
 open terms 8
 completeness for 67
 operational conservativity 112
 operator precedence 9
 overview
 of axioms 256
 of process algebras 263

P

PA
 axioms of 21
 basic terms of 23
 cola machines in 22
 semantics of 23
 signature of 21
 PA_{δ}
 axioms of 23
 basic terms of 24
 semantics of 24
 signature of 23
 PA_{δ}
 axioms of 26
 basic terms of 28
 cola machines in 27
 semantics of 27
 signature of 26
 PA_{drt}
 axioms of 142
 basic terms of 143
 semantics of 142
 signature of 142
 $PA_{drt,\epsilon}$ -ID
 axioms of 207
 basic terms of 208

semantics of 207
 signature of 207
 time determinism for 207

$PA_{drt,\epsilon}$ ⁺-ID
 axioms of standard concurrency in
 209
 completeness of 208
 elimination for 208
 soundness of 208

$PA_{drt,\epsilon}$ ⁻-ID
 axioms of 187
 axioms of standard concurrency in
 195
 basic terms of 192
 completeness of 192
 elimination for 192
 semantics of 190
 signature of 187
 soundness of 192
 time determinism for 190

PA_{drt} ⁻-ID
 axioms of 106
 basic terms of 107
 completeness of 112, 113
 conservativity of 112
 elimination for 107, 109
 ground equivalence involving .. 117,
 120
 semantics of 107
 signature of 106
 soundness of 109, 112

PA_{drt} ⁻-ID'
 axioms of 113
 basic terms of 114
 completeness of 121
 elimination for 114, 116, 117
 ground equivalence involving .. 117,
 120
 semantics of 114
 signature of 113
 soundness of 120, 121

PA'_{drt}
 axioms of 151
 basic terms of 151
 completeness of 152
 elimination for 152
 semantics of 151

signature of	151	projective-limit model	13, 230
soundness of	152	proof outlines	
PA_{drt}^+		for completeness	70
completeness of	151	for elimination	67
conservativity of	150	for soundness	69
elimination for	145, 149	PSF	252
soundness of	149, 150	purity	
PA_ε		of deduction rules	112
axioms of	24	PVS	252
basic terms of	25		
semantics of	25	R	
signature of	24	ready semantics	227
PA's (class)	21	real numbers	269
<i>panth</i> format	12	non-standard	232
parallel composition	<i>see merge</i>	real time	
parametric time	231, 232, 247	misleading use of	229
<i>path</i> format	112	real-space process algebra	233
Plotkin-style semantics	10	real-time process algebra ..	41, 232-234
PMC	246	completeness of	234
precedence of operators	9	recursion	37, 39, 56, 219
prefix integration	234	recursion principles	
process algebra	6	for BPA_{drt}	55
concrete	<i>see concrete process</i>	RSP	56
algebra		RSP(USD)	55
concurrent ..	<i>see concurrent process</i>	reflexive closure	270
algebra		relative time	42, 44, 72, 230, 231
discrete-time	<i>see discrete-time</i>	relativistic space-time	233
process algebra		renaming	39, 247
elementary principles of	6	representation	
elimination to	36	of terms of $BPA_{drt,\varepsilon}^-$ -ID	183
how to cook your own	247	of terms of BPA_{drt}^- -ID	50
real-space ...	<i>see real-space process</i>	of terms of BPA_{drt}^+	61, 64
algebra		root condition	219
real-time	<i>see real-time process</i>	RSP	56
algebra		RSP(DEP)	205
untimed	<i>see untimed process</i>	RSP(USD)	54-56, 60
algebra		keeping track of use of	145
very essence of	8	soundness of	95
process algebras		S	
nomenclature of	261	SDL	177, 212
overview of	263	semantics	6
signatures of	262	of ACP	31
process creation	247	of ACP_δ	34
process terms	8	of ACP_{drt}	153
processes	6	of $ACP_{drt,\varepsilon}$ -ID	209
projection	39	of $ACP_{drt,\varepsilon}^-$ -ID	198
projection function	38		

- of ACP_{drt}^- -ID 122
- of ACP_{drt}^- -ID' 134
- of ACP'_{drt} 168
- of ACP''_{drt} 169
- of $ACP_{drt,\tau}$ 218
- of ACP_ϵ 33
- bisimulation 9
- of BPA 10
- of BPA_δ 15
- of BPA_δ 20
- of $BPA_{\delta\epsilon}$ 18
- of BPA_{drt} 57
- of $BPA_{drt,\epsilon}$ -ID 205
- of $BPA_{drt,\epsilon}^-$ -ID 180
- of BPA_{drt}^- 52
- of BPA_{drt}^- - δ 44
- of BPA_{drt}^- -ID 48
- of BPA'_{drt} 102
- of BPA_ϵ 16
- branching-time 9
- failure 227
- linear-time 8
- notation regarding 10, 16, 44
- of PA 23
- of PA_δ 24
- of PA_δ 27
- of PA_{drt} 142
- of $PA_{drt,\epsilon}$ -ID 207
- of $PA_{drt,\epsilon}^-$ -ID 190
- of PA_{drt}^- -ID 107
- of PA_{drt}^- -ID' 114
- of PA'_{drt} 151
- of PA_ϵ 25
- ready 227
- trace 8, 22
- sequential composition
 - associativity of 8
 - axioms for 8, 9, 44
 - deduction rules for .. 11, 17, 20, 44, 53, 180
 - distributivity of 8, 58, 102
 - unit element of 16, 179
 - zero element of 14, 19
- set theory 269
- σ -step *see* time step
- σ -transition *see* time step
- σ -bisimulation 234
- signature 7
 - of ACP 29
 - of ACP_δ 34
 - of ACP_{drt} 152
 - of $ACP_{drt,\epsilon}$ -ID 209
 - of $ACP_{drt,\epsilon}^-$ -ID 198
 - of ACP_{drt}^- -ID 121
 - of ACP_{drt}^- -ID' 134
 - of ACP'_{drt} 168
 - of ACP''_{drt} 169
 - of $ACP_{drt,\tau}$ 217
 - of ACP_ϵ 32
 - of BPA 7
 - of BPA_δ 13
 - of BPA_δ 19
 - of $BPA_{\delta\epsilon}$ 18
 - of BPA_{drt} 54
 - of $BPA_{drt,\epsilon}$ -ID 204
 - of $BPA_{drt,\epsilon}^-$ -ID 179
 - of BPA_{drt}^- 51
 - of BPA_{drt}^- - δ 42
 - of BPA_{drt}^- -ID 47
 - of BPA'_{drt} 102
 - of BPA_ϵ 16
 - of PA 21
 - of PA_δ 23
 - of PA_δ 26
 - of PA_{drt} 142
 - of $PA_{drt,\epsilon}$ -ID 207
 - of $PA_{drt,\epsilon}^-$ -ID 187
 - of PA_{drt}^- -ID 106
 - of PA_{drt}^- -ID' 113
 - of PA'_{drt} 151
 - of PA_ϵ 24
- signatures
 - of process algebras 262
- silent action
 - deduction rules for 218
 - delayable *see* delayable silent action
 - undelayable .. *see* undelayable silent action
 - untimed .. *see* untimed silent action
- silly index entry 326
- SION iv, 282
- SOS *see* structured operational semantics
- soundness 36, 37

of $ACP_{drt,\epsilon}^+$ -ID	210
of $ACP_{drt,\epsilon}^-$ -ID	200
of ACP_{drt}^- -ID	127, 133
of ACP_{drt}^- -ID'	141
of ACP'_{drt}	168
of ACP''_{drt}	172
of ACP^+_{drt}	165, 166
of $ACP_{drt,\tau}$	219
of $BPA^+_{drt,\epsilon}$ -ID	207
of $BPA^-_{drt,\epsilon}$ -ID	186
of BPA^-_{drt}	87, 88
of $BPA^-_{drt}-\delta$	73, 76
of BPA^-_{drt} -ID	80, 82
of BPA'_{drt}	103
of BPA^+_{drt}	94, 96
by ground equivalence	120
of $PA^+_{drt,\epsilon}$ -ID	208
of $PA^-_{drt,\epsilon}$ -ID	192
of PA^-_{drt} -ID	109, 112
of PA^-_{drt} -ID'	120, 121
of PA'_{drt}	152
of PA^+_{drt}	149, 150
proof outlines for	69
of RSP(USD)	95
in untimed process algebra	36
space-time	
classical	233
relativistic	233
SRM	178
stack	177
standard basic terms	<i>see</i> basic terms
standard concurrency	
axioms of	<i>see</i> axioms of standard concurrency
start delay	<i>see</i> unbounded start delay
state operators	39, 247
σ -step	<i>see under S</i>
strong time factorization	43
structured operational semantics	10
sum of deduction systems	112
summation	49, 54
surreal numbers	232
symmetric closure	270
synchronization	28
syntactical equivalence	269

T

T-LOTOS	239
TCCS (Chen)	<i>see</i> CCS, Timed (Chen)
TCCS (Moller and Tofts)	<i>see</i> CCS, Temporal
TCCS (Wang)	<i>see</i> CCS, Timed (Wang)
TE-LOTOS	240
temporal logic	213
term model	13
term rewriting	270
term-deduction systems	<i>see</i> deduction systems
term-rewriting system	68
terminating step	
deduction rules for	10, 11, 31, 32, 44, 48, 57, 107, 123, 218
termination	
successful	16
unsuccessful	14
termination operator	26
termination option	
deduction rules for	16, 17, 25, 33, 180, 191, 200, 206
TIC	239
time determinism	65, 172
for $ACP_{drt,\epsilon}$ -ID	209
for $ACP^-_{drt,\epsilon}$ -ID	198
for $BPA_{drt,\epsilon}$ -ID	206
for $BPA^-_{drt,\epsilon}$ -ID	181
for $PA_{drt,\epsilon}$ -ID	207
for $PA^-_{drt,\epsilon}$ -ID	190
time factorization	42, 65
w.r.t. bisimulation	45, 234
rejection of	230
strong	43
weak	43, 44, 178, 179, 181, 198
time step	
deduction rules for	44, 53, 57, 107, 123, 143, 180, 191, 200, 206, 218
time-slices	41
time-unit delay constant	179
axioms for	179, 202
deduction rules for	180
distributivity of	198
time-unit delay operator	42

axioms for 44, 52
 commutativity of 122
 deduction rules for 44, 53
 distributivity of 122
 shorthand for 72
 Timed LOTOS 240
 Enhanced *see* ET-LOTOS
 timestamped notation 231
 TLOTOS 239
 TOOLBUS 246–247
 TPCCS *see* CCS, Timed Probabilistic
 TPL 244–245
 trace semantics 8
 σ -transition *see under* **S**
 transitive closure 270
 twin axioms 25, 52
 two-phase notation 231

U

U-LOTOS 239
 unbounded start delay 54, 55, 244
 axiom nomenclature for 255
 axioms for 55, 56, 102, 151, 169
 commutativity of 169
 deduction rules for 57
 distributivity of 58, 102, 169
 undelayable actions 42
 deduction rules for 44, 180
 undelayable deadlock
 axioms for 47, 48, 179, 202
 undelayable empty process 179
 axioms for 179, 188, 199, 203
 undelayable silent action 213
 axioms for 217
 unit element
 of free merge 25, 192, 208
 of left merge . 25, 33, 192, 200, 208,
 210
 of merge 33, 200, 210
 of sequential composition .. 16, 179
 untimed actions
 deduction rules for 10, 16
 untimed deadlock
 axioms for 14, 19
 deduction rules for 15
 untimed empty process 16
 axioms for 16, 24, 32

 deduction rules for 16
 difference with literature 26
 untimed process algebra 5–39
 axioms of standard concurrency in
 37
 completeness in 37
 elimination in 36
 embeddings for 38
 soundness in 36
 untimed silent action 67
 urgent actions 233, 234

V

Verhoef's method 71

W

weak time factorization 43, 44, 178,
179, 181, 198

well-foundedness
 of deduction rules 112

Z

zero element
 of alternative composition .. 14, 19,
 20
 of left merge 192, 200, 208, 210
 of sequential composition ... 14, 19
 true 178

Titles in the IPA Dissertation Series

The State Operator in Process Algebra

J. O. Blanco

Faculty of Mathematics and Computing Science, TUE, 1996-1

Transformational Development of Data-Parallel Algorithms

A. M. Geerling

Faculty of Mathematics and Computer Science, KUN, 1996-2

Interactive Functional Programs: Models, Methods, and Implementation

P. M. Achten

Faculty of Mathematics and Computer Science, KUN, 1996-3

Parallel Local Search

M. G. A. Verhoeven

Faculty of Mathematics and Computing Science, TUE, 1996-4

The Implementation of Functional Languages on Parallel Machines with Distrib. Memory

M. H. G. K. Kessler

Faculty of Mathematics and Computer Science, KUN, 1996-5

Distributed Algorithms for Hard Real-Time Systems

D. Alstein

Faculty of Mathematics and Computing Science, TUE, 1996-6

Communication, Synchronization, and Fault-Tolerance

J. H. Hoepman

Faculty of Mathematics and Computer Science, UvA, 1996-7

Reductivity Arguments and Program Construction

H. Doornbos

Faculty of Mathematics and Computing Science, TUE, 1996-8

Functorial Operational Semantics and its Denotational Dual

D. Turi

Faculty of Mathematics and Computer Science, VUA, 1996-9

Single-Rail Handshake Circuits

A. M. G. Peeters

Faculty of Mathematics and Computing Science, TUE, 1996-10

A Systems Engineering Specification Formalism

N. W. A. Arends

Faculty of Mechanical Engineering, TUE, 1996-11

Normalisation in Lambda Calculus and its Relation to Type Inference

P. Severi de Santiago

Faculty of Mathematics and Computing Science, TUE, 1996-12

Abstract Interpretation and Partition Refinement for Model Checking

D. R. Dams

Faculty of Mathematics and Computing Science, TUE, 1996-13

Topological Dualities in Semantics

M. M. Bonsangue

Faculty of Mathematics and Computer Science, VUA, 1996-14

Algorithms for Graphs of Small Treewidth

B. L. E. de Fluiter

Faculty of Mathematics and Computer Science, UU, 1997-01

Process-algebraic Transformations in Context

W. T. M. Kars

Faculty of Computer Science, UT, 1997-02

A Generic Theory of Data Types

P. F. Hoogendijk

Faculty of Mathematics and Computing Science, TUE, 1997-03

The Evolution of Type Theory in Logic and Mathematics

T. D. L. Laan

Faculty of Mathematics and Computing Science, TUE, 1997-04

Preservation of Termination for Explicit Substitution

C. J. Bloo

Faculty of Mathematics and Computing Science, TUE, 1997-05

Discrete-Time Process Algebra

J. J. Vereijken

Faculty of Mathematics and Computing Science, TUE, 1997-06