Eindhoven University of Technology

Department of Mathematics and Computer Science

Master's Thesis

Unique Sequential Decomposition in Process Algebras with **0** and **1**

by

B.A.G. Senders

Supervisor: Dr. B. Luttik
Tutor: P.J.A. van Tilburg, MSc

Eindhoven, August 12, 2009

Preface

A little less than a year ago, I started the last batch of courses for my masters education in Computer Science and Engineering at the Eindhoven University of Technology. One of these courses was a formal methods seminar, given by the Formal Methods research group of the University's Department of Mathematics and Computer Science. This seminar focused on exploring the gap between process algebra and automata theory. I always had some inkling of interest in both subjects, but actual hands-on experience had been limited to small exercises. The seminar taught me that working on a purely theoretical research problem – especially with a group of peers – can be a very rewarding and fulfilling experience, and it motivated me to find a related subject for my master project.

This is why, in January 2009, I decided to continue along this path and perform my final master project at the Formal Methods group. There were several research topics available, of which the current one, suggested by Bas Luttik, piqued my interest the most: it provided the process algebraic setting which I came to like during the seminar, with a well-defined starting point (namely sequential decomposition in $\mathsf{BPA}_{\mathbf{1}}$) and several options to research after that (decomposition in $\mathsf{BPA}_{\mathbf{0,1}}$ and/or $\mathsf{BPP}_{\mathbf{0,1}}$, recursion, generalisation to monoids). This thesis is the culmination of about half a year's work of research, proving and writing on the subject.

I wish to heartily thank my supervisor Bas Luttik and my tutor and friend Paul van Tilburg for their continuous guidance during the project, helpful nudges in the right direction, and conversations that gave me motivation and support in times of doubt and uncertainty. Doubts during my studies were also alleviated by Pieter Cuijpers (who taught the seminar course and greatly stimulated my enthusiasm) and Jaap van der Woude, to both of whom I am very grateful as well.

Many thanks also go to my friends, especially those studying at the university, for providing a very pleasurable working environment with just the right amount of distraction to get the job done; and to my parents, sister and grandparents, for always showing interest and providing encouragement to finish my studies.

Bram Senders

Eindhoven, August 2, 2009

Summary

In this thesis we research the existence of unique sequential decomposition in process algebras, specifically in algebras containing the empty process **1** and optionally the deadlock process **0**. Having a unique decomposition result for an algebra facilitates proving properties like decidability of bisimulation and finite axiomatisation, and in general makes sure each process can be given in a standard form with known properties. Sequential decompositions are currently less well researched than parallel decompositions, and not many results for decomposition of processes including **0** and/or **1** exist.

Therefore we take the relatively simple process algebras $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$ to investigate whether these algebras have a unique sequential decomposition. We first prove a cancellation theorem that aids us in proving uniqueness of sequential decompositions. Then we prove that all processes in $\mathsf{BPA}_{\mathbf{1}}$ have a unique sequential prime decomposition. We briefly show that this does not hold in general for processes in $\mathsf{BPA}_{\mathbf{0,1}}$, then go on to investigate two alternative notions of unique decomposition for processes in $\mathsf{BPA}_{\mathbf{0,1}}$, both of which are as close as possible to a prime decomposition. One decomposition concerns processes which always end in deadlock; the other concerns processes which sometimes end in deadlock.

Furthermore we generalise the unique sequential decomposition result obtained for $\mathsf{BPA}_{\mathbf{1}}$ to a setting of monoids. We propose a number of properties that should be satisfied for a monoid to have a unique prime decomposition result, then prove that every element from a monoid satisfying these properties has a unique prime decomposition. This generalisation shows that decomposition does not depend on the exact semantics of the operators present in algebras like $\mathsf{BPA}_{\mathbf{1}}$; in fact, a decomposition can be given for any element in a setting with a sequential operator that adheres to the proposed properties. We also prove that cancellation is a necessary property that cannot be derived from the others. Finally we show that $\mathsf{BPA}_{\mathbf{1}}$ is in fact one of the algebras satisfying the properties.

Contents

Chapter 1

Introduction

In the field of process algebra, it is often useful to be able to write every process in some canonical form. A form that is often used for this is the *prime decomposition*. The idea of a prime decomposition is derived from the fundamental theorem of arithmetic as formulated by Euclid. This theorem states that every positive natural number can be written as a product of prime numbers, and that there is only one way of doing so for each number, i.e., that this "decomposition" of a number into a set of prime numbers is unique. For example, the only way to write the number 294 as a product of primes is as $2 \times 3 \times 7 \times 7$ (disregarding the ordering of the primes in the product).

In the same way, it is possible in some process algebras to write every process as a product of a number of prime processes. In this case, the "product" operator is usually a composition operator, such as the parallel or sequential operators present in many process algebras. Primeness then becomes the inability to further decompose a process by the chosen operator. However, it is not known for process algebras in general whether a prime decomposition exists, and if it does, whether this decomposition is necessarily unique.

Milner and Moller were the first to address the question of whether prime decomposition is unique with regard to parallel processes [18]; they proved that uniqueness of the decomposition depends on the congruence being used (bisimilarity, failures equivalence, trace equivalence), and on whether or not "infinite processes" – processes not limited to performing a bounded or finite number of actions – are allowed. They proved these results twice, first without, then with the aid of a cancellation lemma. Both results were first noted in Moller's PhD thesis [19]. The cancellation lemma was first used by Castellani and Hennessy (as part of a larger simplification lemma) in the context of distributed bisimulations [10].

Significant work has been performed by Luttik and Van Oostrom in generalising the notion of unique parallel prime decompositions to a setting of commutative monoids [17]. In this way, it suffices to provide a reduction of the process algebra to a commutative monoid for which the required properties hold, instead of needing to prove a unique decomposition separately for every process algebra.

Use for a unique prime decomposition

Having a unique decomposition result for process algebras is useful in several ways. In its most general setting, having unique prime decomposition fulfils a technical requirement for making sure each process can be given in a standard form with known properties. This standard form then facilitates being able to prove other properties. For one, it can be used to prove the decidability of bisimulation for an algebra, as is done for example by Christensen, Hirshfeld and Moller for BPP [12]. Secondly, unique decomposition can also be used to prove or disprove the existence of a finite axiomatisation of a process algebra. This is demonstrated by Moller, who proves the existence of such an axiomatisation for PA [20], and disproves it for CCS [21].

Further results regarding the use of a decomposition in general, and unique decomposition in particular, are given in Christensen's PhD thesis [11, pp. 4–7].

Sequential decomposition with 0 and 1

In this thesis, we research a number of aspects of unique prime decomposition that have not received as much attention as results obtained in other process algebras. Specifically we address *sequential decomposition*, in contrast with the more often-researched *parallel decomposition*; and we do so in a setting with the *empty process* **1** and optionally a *deadlock* process **0**.

To this end we pick two simple process algebras called $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$, both derived from BPA, the Basic Process Algebra. BPA was first defined by Bergstra and Klop [8]. $\mathsf{BPA}_{\mathbf{1}}$ adds the empty process **1** to BPA, and $\mathsf{BPA}_{\mathbf{0,1}}$ furthermore adds the deadlock process **0** to $\mathsf{BPA}_{\mathbf{1}}$. The empty process was initially introduced (represented by the symbol ε) by Koymans and Vrancken [15], and improved by Vrancken [22]. They defined a different semantics for these processes than the one we use here; we use Structural Operational Semantics as given for an algebra containing **0** and **1** by Baeten in e.g. [3]. The deadlock process was already present (represented by the symbol δ) in [8], but not in combination with BPA (which is also mentioned in that paper), only in the more general process theory ACP.

Our aim is now to prove that each process in $\mathsf{BPA}_{\mathbf{1}}$ has a unique prime decomposition with respect to the sequential composition operator. It is already known that not all processes in $\mathsf{BPA}_{\mathbf{0,1}}$ have a prime decomposition, so a unique prime decomposition result does not exist for $\mathsf{BPA}_{\mathbf{0,1}}$. In light of this situation, we investigate whether an alternative decomposition for processes in $\mathsf{BPA}_{\mathbf{0,1}}$ – one as close as possible to being a prime decomposition – does yield a unique result. We actually come up with two decomposition results for processes in $\mathsf{BPA}_{\mathbf{0,1}}$ which have deadlock; one for processes which always deadlock and one for processes which sometimes deadlock. Together with the decomposition results obtained for $\mathsf{BPA}_{\mathbf{1}}$, this provides a unique decomposition for every process in $\mathsf{BPA}_{\mathbf{0,1}}$.

Lastly we generalise this notion of sequential prime decomposition to a monoidic setting, in a similar way as has been done for decomposition using a commutative operator (like the parallel composition operator) by Luttik and Van Oostrom [17]. This shows that unique prime decomposition does not hold only for $\mathsf{BPA}_{\mathbf{1}}$, but for any algebra (or other structure) satisfying the proposed properties.

The role of 0 and 1

Both **0** and **1** enhance the expressiveness of a process algebra which includes them.

The addition of the deadlock process **0** enables processes to enter a state from which neither performing an action, nor terminating is possible. The occurrence of deadlock is most often the result of failing to communicate in a process algebra which includes the concepts of parallelism and communication. Even though these concepts are not included in the algebras we study here, studying **0** in a smaller context still enables us to study the effects of deadlock on sequential decomposition.

The empty process **1** is significant because it allows for *impure termination*: the possibility for a process to terminate when there are still actions that can be performed. To be able to do so is very important when considering the relation between process algebra and automata theory; two fields which have much in common, but have historically not been strongly linked. This link is explored in more detail by Baeten, Cuijpers, Luttik and Van Tilburg [5, 6, 7]. Now, in an automaton it is possible to designate any subset of states as final states, yet in both CCS- and ACP-style process algebras (including BPA without **1**) it is only possible to terminate when no actions can be performed anymore. This incompatibility is alleviated by introducing impure termination. A different case for inclusion of an explicit empty process in process algebras, for embedding untimed into timed process algebra, is also argued by Baeten [2].

By proving the existence (or absence) of a unique prime decomposition result in process algebras that include **0** and/or **1**, we hope to facilitate a better understanding of the consequences arising from adding these processes to an algebra. Counterexamples showing where unique decomposition fails are especially useful in this area.

Methodology

In both $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$ we prove the existence of a unique decomposition by way of a cancellation lemma; this way of proving decomposition was first performed by Moller in his PhD thesis [19], and enables a much shorter proof than is possible without using cancellation. We prove cancellation by showing that it is a bisimulation relation, as is also done by e.g. Milner and Moller for parallel processes [18]. However, while the technique is the same, the actual contents of the proof are quite different, on the one hand because we are proving a sequential rather than a parallel result, and on the other hand because impure termination using **1** complicates matters somewhat.

The actual decomposition proofs are performed by induction on the size of a process term; these are mostly inspired by a very brief proof of decomposition for BPA by Burkart, Caucal, Moller and Steffen [9]. The proof technique we use is the same as theirs, however our proof is more detailed, and also somewhat more complicated, again because of the addition of **1**.

Results in proving unique prime decomposition for monoids are based on the works of Luttik [16], especially as a basis for the properties which are required to prove decomposition in a monoidic setting, and Luttik and Van Oostrom [17]

for the way of proving that a process algebra divided using bisimulation is a monoid.

Limitations

The biggest limitation present in the current work is that we only consider finite processes, i.e., processes for which the specification is finite and non-recursive. This is because in general unique prime decomposition does not hold for infinite processes, and investigating alternative notions of decomposition where infinite processes are possible (analogous to notions of decomposition which allow for deadlock) was outside of the scope of this thesis. We also only consider closed terms. A consequence of this is that applications of decomposition like the decidability of bisimulation and finite axiomatisation are not applicable here yet, since these applications are only useful for algebras that contain infinite processes or open terms; in closed non-recursive terms the sequential composition operator can actually be factored out. Nevertheless, the results obtained here are a first step towards obtaining sequential decomposition in a broader, more open setting.

Some very preliminary work has been performed in supporting infinite processes by allowing recursive specifications; this work is included in the appendix.

Overview

This thesis is structured as follows: in Chapter 2 we introduce the process algebras $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$ and provide their syntax and semantics. Next, in Chapter 3 we give a cancellation theorem that is used for the unique decomposition proofs that follow. After that, in Chapter 4 we prove three separate unique decomposition results: one for processes without deadlock, one for processes which always end in deadlock and one for processes which sometimes end in deadlock. Together, these provide a unique decomposition result for each process in $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$. In Chapter 5 we generalise the decomposition result for deadlock-free processes to a monoidic setting, so that unique sequential decomposition holds for every monoid conforming to a number of properties. We conclude with Chapter 6, which also lists some future work. Finally, Appendix A provides some early results obtained in adding recursive specifications to the algebras, thereby making it possible to specify infinite processes.

Chapter 2

Syntax, semantics and other preliminaries

Before we begin to discuss decomposition, we first formally introduce the process algebras $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ and their syntax and semantics. For processes in these algebras we also define the well-known bisimulation relation, some axioms, and a measure of the *size* of a process, which is used extensively in the proofs in upcoming chapters.

Definition 2.1. The syntax of terms over $\mathsf{BPA}_{\mathbf{1}}$ is defined by the following signature:

1. There is a *successful termination* process, or empty process $\mathbf{1}$.
2. For each action $a \in \mathcal{A}$, where $\mathcal{A}$ is a (finite) action set, there is a unary *action prefix* operator $a.\_$.
3. There is a binary *alternative composition* operator $\_ + \_$.
4. There is a binary *sequential composition* operator $\_ \cdot \_$.

The syntax of terms over $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ adds one constant to the above:

5. There is a *deadlock* process $\mathbf{0}$.

Note that we do not define recursive specifications for our process algebras, and we consider only finite-sized, closed terms, as this thesis will mostly concern finite processes. The signature given above results in the following grammar for $\mathsf{BPA}_{\mathbf{1}}$:

$$P ::= \mathbf{1} \mid a.P \mid P + P \mid P \cdot P \ ,$$

and for $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$:

$$P ::= \mathbf{0} \mid \mathbf{1} \mid a.P \mid P + P \mid P \cdot P \ .$$

We can now define semantics for the process algebras.

Definition 2.2. The operational semantics for $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$ are defined using *Structural Operational Semantics* [1] (or SOS for short), as given by the rules in Table 2.1. Here $\mathcal{A}$ is the set of possible actions.

$$1\ \frac{}{\mathbf{1}\downarrow} \qquad 2\ \frac{}{a.p \xrightarrow{a} p}$$

$$3\ \frac{p \xrightarrow{a} p'}{p + q \xrightarrow{a} p'} \qquad 4\ \frac{q \xrightarrow{a} q'}{p + q \xrightarrow{a} q'} \qquad 5\ \frac{p\downarrow}{p + q\downarrow} \qquad 6\ \frac{q\downarrow}{p + q\downarrow}$$

$$7\ \frac{p \xrightarrow{a} p'}{p \cdot q \xrightarrow{a} p' \cdot q} \qquad 8\ \frac{p\downarrow \quad q \xrightarrow{a} q'}{p \cdot q \xrightarrow{a} q'} \qquad 9\ \frac{p\downarrow \quad q\downarrow}{p \cdot q\downarrow}$$

Table 2.1: SOS rules for $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$ ($a \in \mathcal{A}$)

In these SOS rules, we write $p \xrightarrow{a} p'$ if a process p can perform an a-action and then continue as process p'. As a shorthand notation, we write $p \longrightarrow p'$ if there exists an $a \in \mathcal{A}$ such that $p \xrightarrow{a} p'$; we write $p \longrightarrow^n p'$ if there are processes $p_0, \ldots, p_n$ such that $p = p_0 \longrightarrow p_1 \longrightarrow \ldots \longrightarrow p_{n-1} \longrightarrow p_n = p'$, i.e., process p can get to process p' in n steps (where for $n = 0$, we take $p = p'$); and we write $p \longrightarrow^* p'$ if there is an $n \geq 0$ such that $p \longrightarrow^n p'$, i.e., process p can get to process p' in some number of steps (which may be zero).

We write $p\downarrow$ if a process p can terminate. It is allowed for a process to terminate even if it can still perform some action, in contrast with process algebras where termination is established by an absence of actions left to perform; for example, the process $a.\mathbf{1} + \mathbf{1}$ can either perform an a-action and then terminate, or terminate immediately. We call this option between choosing to terminate or performing an action impure termination, and define it as follows.

Definition 2.3. A process p has *impure termination* if $p \longrightarrow^* p'$ for some process p', and we have both $p'\downarrow$ and $p' \longrightarrow p''$ for some process p'', i.e., if it is possible from p to reach a state in 0 or more steps where both termination and performing an action are possible.

Regarding deadlock, since $\mathsf{BPA}_{\mathbf{0,1}}$ only extends $\mathsf{BPA}_{\mathbf{1}}$ by adding the deadlock constant $\mathbf{0}$, and this constant has no semantics for performing an action or terminating (since it can do neither), the semantics for $\mathsf{BPA}_{\mathbf{0,1}}$ are the same as those for $\mathsf{BPA}_{\mathbf{1}}$.

We call a state in which no SOS-rule is applicable, i.e. where it is not possible to perform an action or to terminate, a *deadlock state*. Obviously such states can only exist in $\mathsf{BPA}_{\mathbf{0,1}}$, not in $\mathsf{BPA}_{\mathbf{1}}$. We say a process *has deadlock* if it can reach a deadlock state, and we call it *deadlock-free* if it cannot. It is easy to see that for every transition $p \xrightarrow{a} p'$, if p is deadlock-free, then p' is deadlock-free as well.

Note. In the remainder of this thesis, if we reason about arbitrary processes without mentioning in which algebra they originate, then we assume these processes are taken to be from $\mathsf{BPA}_{\mathbf{0,1}}$. Since every process in $\mathsf{BPA}_{\mathbf{1}}$ is also a process in $\mathsf{BPA}_{\mathbf{0,1}}$, this means the reasoning is taken to be as general as possible.

2.1 Bisimilarity of processes

We use bisimilarity for reasoning about the equality of processes. In the presence of explicit termination using $\mathbf{1}$, bisimulation is defined as is done in e.g. [17]. We have slightly adapted this definition to explicitly enumerate the symmetrical cases.

Definition 2.4. A binary relation $\mathcal{R}$ on processes in $\mathsf{BPA}_{\mathbf{1}}$ or $\mathsf{BPA}_{\mathbf{0,1}}$ is a *bisimulation relation* iff the following four properties hold, for all pairs of processes $(p, q) \in \mathcal{R}$ and for all $a \in \mathcal{A}$:

1. If there is a process p' such that $p \xrightarrow{a} p'$, then there must be a process q' such that $q \xrightarrow{a} q'$ and $(p', q') \in \mathcal{R}$.
2. If there is a process q' such that $q \xrightarrow{a} q'$, then there must be a process p' such that $p \xrightarrow{a} p'$ and $(p', q') \in \mathcal{R}$.
3. If process p can terminate (so $p\downarrow$), then q must also be able to terminate.
4. If process q can terminate (so $q\downarrow$), then p must also be able to terminate.

Two arbitrary processes p and q are called *bisimilar* if there exists a bisimulation relation $\mathcal{R}$ such that $(p, q) \in \mathcal{R}$. For this we use the notation $p \leftrightarroweq q$.

In the sequel we are going to need some axioms regarding the bisimilarity of processes; these are presented in this section. Note that this is not a complete axiomatisation of $\mathsf{BPA}_{\mathbf{1}}$ or $\mathsf{BPA}_{\mathbf{0,1}}$; it just lists the axioms that are actually used in the remainder of this thesis. For example, we do not use the idempotency or associativity of the alternative composition operator. These axioms will be applied throughout, without explicit reference to the lemma.

Lemma 2.5. *The following axioms regarding bisimilarity of processes are sound for arbitrary processes p, q and r:*

$$
\begin{aligned}
p + q &\leftrightarroweq q + p \,, \\
p + \mathbf{0} &\leftrightarroweq p \,, \\
p \cdot (q \cdot r) &\leftrightarroweq (p \cdot q) \cdot r \,, \\
p \cdot \mathbf{1} &\leftrightarroweq p \,, \\
\mathbf{1} \cdot q &\leftrightarroweq q \,, \\
\mathbf{0} \cdot q &\leftrightarroweq \mathbf{0} \,, \\
(a.p) \cdot q &\leftrightarroweq a.(p \cdot q) \,, \\
(p + q) \cdot r &\leftrightarroweq (p \cdot r) + (q \cdot r) \,.
\end{aligned}
$$

Proof. Both sides of each equivalence are in a relation for which the bisimulation properties from Definition 2.4 hold, as is apparent from the SOS rules in Table 2.1. □

Note. The associativity axiom $p \cdot (q \cdot r) \leftrightarroweq (p \cdot q) \cdot r$ is mostly applied implicitly in the sequel, so we write e.g. $p \cdot q \cdot r$ instead of either $p \cdot (q \cdot r)$ or $(p \cdot q) \cdot r$.

For further treatment of these axioms, see Chapter 6, "Sequential processes", of the forthcoming book on process algebra by Baeten, Basten and Reniers [4].

2.2 Size of processes

In the remainder of this thesis we will perform some proofs by induction on the size of a process. Since many variations on size are possible, we give here a formal definition.

Definition 2.6. We define the *size* of a process p, written $|p|$, to be the length of the longest transition sequence starting at p:

$$|p| = \max\{n \in \mathbb{N} \mid \exists p' : p \longrightarrow^n p'\} .$$

When calculating the size of a process, we do not only take into account transition sequences ending in termination, but also those ending in deadlock, so e.g. $|a.\mathbf{1} + a.a.\mathbf{0}| = 2$.

Note. This definition does not allow for processes for which $|p| = \infty$, as we do not consider terms with infinite size. Also note that the size of any process is strictly within $\mathbb{N}$; we do not allow for a negative size (since any transition sequence consists of at least zero transitions).

Now the size of most processes can easily be defined recursively, as per the following lemma.

Lemma 2.7. *For processes p and q and for $a \in \mathcal{A}$, the following equalities can be derived:*

$$\begin{aligned} |\mathbf{0}| &= 0 , \\ |\mathbf{1}| &= 0 , \\ |a.p| &= 1 + |p| , \\ |p + q| &= \max(|p|, |q|) , \\ |p \cdot q| &= |p| + |q| \quad \textit{if } p \textit{ is deadlock-free} . \end{aligned}$$

Proof. By definition of size (Definition 2.6) and the SOS rules in Table 2.1. □

Note that from these equations, the only way for a process p to have $|p| = 0$ is if either $p \leftrightarrows \mathbf{0}$ or $p \leftrightarrows \mathbf{1}$; for $\mathsf{BPA}_{\mathbf{1}}$, this means $|p| = 0$ iff $p \leftrightarrows \mathbf{1}$.

Some lemmas regarding the size of processes are going to be useful in the remainder of this thesis; we state and prove them here.

Lemma 2.8. *For processes p and q, if $p \leftrightarrows q$, then $|p| = |q|$.*

Proof. We prove the lemma by contradiction. Assume without loss of generality that (for $p \leftrightarrows q$) we have $|p| > |q|$. Then there is a transition sequence of length $|p|$ starting at p which is longer than all sequences in q, so q cannot simulate this sequence. But then $p \not\leftrightarrows q$, which is a contradiction. Hence it must be that $|p| = |q|$. □

Lemma 2.9. *If $p \xrightarrow{a} p'$ for processes p and p' and some action $a \in \mathcal{A}$, then $|p'| < |p|$.*

Proof. Suppose $|p'| = n$, so the longest transition sequence starting at p' takes n steps. By $p \xrightarrow{a} p'$ it is possible to get from p to p' in one step, so there is a transition sequence of $n + 1$ steps starting at p. Thus the longest transition sequence starting at p must also be at least $n + 1$ steps long. Hence, $|p| \geq n + 1 > n = |p'|$, so $|p'| < |p|$. □

Chapter 3

Cancellation

Cancellation is a property of processes stating that if two processes with a common right-hand argument of a sequential composition are bisimilar, then the two processes with the right-hand argument removed are also bisimilar. That is, if $p \cdot r \leftrightarrows q \cdot r$ holds, then $p \leftrightarrows q$ holds as well. More properly this is called *right cancellation*, since it removes a component on the right-hand side of a process (as opposed to removing a component on the left-hand side); yet we refer to the property simply by "cancellation", since right cancellation is the only cancellation we use.

Having cancellation greatly simplifies the proof of unique decomposition. While there are ways to prove decomposition without using cancellation – only establishing cancellation as a result of proving decomposition, instead of the other way around – this results in a longer and more involved proof. Cancellation was already used for proving unique prime decomposition in the fundamental theorem of arithmetic. Moller was the first one to use cancellation (which he called simplification) for proving decomposition in a process algebraic context; specifically for decomposition into parallel components [19]. We will use cancellation to prove decomposition as well.

This property holds for all processes in $\mathsf{BPA}_{\mathbf{1}}$, but not, in general, for those in $\mathsf{BPA}_{\mathbf{0,1}}$. First of all, p and q should be deadlock-free: take process $r = a.\mathbf{0} + \mathbf{1}$; then it can easily be shown that $\mathbf{1} \cdot r \leftrightarrows (a.\mathbf{0} + \mathbf{1}) \cdot r$, but obviously $\mathbf{1} \not\leftrightarrows a.\mathbf{0} + \mathbf{1}$. Secondly, it is required that $r \not\leftrightarrows \mathbf{0}$: take $p = a.\mathbf{1} + \mathbf{1}$, $q = a.\mathbf{1}$, and $r = \mathbf{0}$, then $p \cdot r \leftrightarrows q \cdot r$, but $p \not\leftrightarrows q$. Hence we require that p and q are deadlock-free and $r \not\leftrightarrows \mathbf{0}$.

Now, to prove cancellation we first need an auxiliary lemma.

Lemma 3.1. *If $r \xrightarrow{a} r'$ for processes r and r' and some action $a \in \mathcal{A}$, then there is no deadlock-free process p such that $p \cdot r \leftrightarrows r'$.*

Proof. We prove the lemma by contradiction. Suppose that there is a deadlock-free process p for which $p \cdot r \leftrightarrows r'$. Then, by Lemma 2.8, $|p \cdot r| = |r'|$, and by Lemma 2.7 (since p is deadlock-free), $|p \cdot r| = |p| + |r|$, so $|p| + |r| = |r'|$ and $|p| = |r'| - |r|$. But because we have $r \xrightarrow{a} r'$, it holds by Lemma 2.9 that

$|r'| < |r|$, so $|r'| - |r| < 0$ and thus $|p| < 0$. But there cannot exist a process with a longest transition sequence smaller than zero, so this is a contradiction. Hence no such p exists. □

Note that such a p can exist when it is allowed to have deadlock, e.g. take $p = \mathbf{0}$ and $r = a.\mathbf{0}$, then $r' = \mathbf{0}$, so $p \cdot r \leftrightarrows r'$.

Using the previous lemma we can prove the following cancellation theorem for processes in $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0,1}}$.

Theorem 3.2. *If for some deadlock-free processes p and q there exists a process r (which may have deadlock) with $r \not\leftrightarrows \mathbf{0}$, such that $p \cdot r \leftrightarrows q \cdot r$, then $p \leftrightarrows q$.*

Proof. Consider the following relation $\mathcal{R}$:

$$\mathcal{R} = \{(p, q) \mid p, q \text{ deadlock-free} \wedge \exists r : r \not\leftrightarrows \mathbf{0} \wedge p \cdot r \leftrightarrows q \cdot r\}$$

Obviously, if $\mathcal{R}$ is a bisimulation relation, then the theorem holds. Recall the definition of a bisimulation (Definition 2.4); we need to prove that each of the four properties of a bisimulation relation hold for $\mathcal{R}$, to establish that $\mathcal{R}$ is a bisimulation relation. For each property, assume $(p, q) \in \mathcal{R}$ with p, q deadlock-free, and take an r such that $r \not\leftrightarrows \mathbf{0}$ and $p \cdot r \leftrightarrows q \cdot r$.

1. Suppose $p \xrightarrow{a} p'$ for some $a \in \mathcal{A}$; we need to show that there exists a q' such that $q \xrightarrow{a} q'$ and $(p', q') \in \mathcal{R}$.

   By SOS rule 7 (see Table 2.1) on $p \xrightarrow{a} p'$ we have $p \cdot r \xrightarrow{a} p' \cdot r$, so by bisimilarity of $p \cdot r$ and $q \cdot r$ there exists an $s \leftrightarrows p' \cdot r$ such that $q \cdot r \xrightarrow{a} s$. Fix a derivation of $q \cdot r \xrightarrow{a} s$; we distinguish cases according to which SOS rule has been applied last in this derivation. This can either be rule 7 or rule 8:

   (a) By SOS rule 7 there is a q' such that $q \xrightarrow{a} q'$; then $s = q' \cdot r$, so $p' \cdot r \leftrightarrows q' \cdot r$. By definition of $\mathcal{R}$, since $p' \cdot r \leftrightarrows q' \cdot r$, and p', q' deadlock-free (since p, q deadlock-free) and $r \not\leftrightarrows \mathbf{0}$, this means that $(p', q') \in \mathcal{R}$.

   (b) By SOS rule 8 it holds that $q\downarrow$ and there is an r' such that $r \xrightarrow{a} r'$; then $s = r'$, so $p' \cdot r \leftrightarrows r'$. But by applying Lemma 3.1 (which is possible because p is deadlock-free, so p' is deadlock-free also) this is a contradiction, so this case cannot occur.

   Only the first case can occur, so there is a q' such that $q \xrightarrow{a} q'$ with $(p', q') \in \mathcal{R}$; this satisfies the bisimulation relation requirement.

2. Suppose $q \xrightarrow{a} q'$ for some $a \in \mathcal{A}$; we need to show that there exists a p' such that $p \xrightarrow{a} p'$ and $(p', q') \in \mathcal{R}$.

   This is analogous to the case for $p \xrightarrow{a} p'$ above.

3. Suppose $p\downarrow$; we need to show that $q\downarrow$.

   To show this, we apply case distinction on r. Since $r \not\leftrightarrows \mathbf{0}$, it is always possible for r to either perform an action and/or terminate. We distinguish between these two options:

(a) Process r can perform an action: $r \xrightarrow{a} r'$ for some $a \in \mathcal{A}$. Then by SOS rule 8, $p \cdot r \xrightarrow{a} r'$, and by bisimilarity of $p \cdot r$ and $q \cdot r$ there exists an $s \leftrightarrows r'$ with $q \cdot r \xrightarrow{a} s$. Fix a derivation of $q \cdot r \xrightarrow{a} s$; we again distinguish cases according to which SOS rule has been applied last in this derivation, which is either SOS rule 7 or rule 8:

i. By SOS rule 7 there is a q' such that $q \xrightarrow{a} q'$; then $s = q' \cdot r$, so $r' \leftrightarrows q' \cdot r$. But by Lemma 3.1 (which we can apply, since q is deadlock-free, so q' is deadlock-free also) this is a contradiction, so this case cannot occur.

ii. By SOS rule 8 it holds that $q\downarrow$ and $s = r'$; so we have $q\downarrow$.

Only the second case can occur, and in this case we have $q\downarrow$.

(b) Process r can terminate: $r\downarrow$. Since $p\downarrow$ holds, by SOS rule 9 it holds that $p \cdot r\downarrow$, and by bisimilarity of $p \cdot r$ and $q \cdot r$ it also holds that $q \cdot r\downarrow$. This can only be satisfied by SOS rule 9, so $q\downarrow$.

In either of the above cases we have $q\downarrow$; this satisfies the bisimulation relation requirement.

4. Suppose $q\downarrow$; we need to show that $p\downarrow$.

   This is analogous to the case for $p\downarrow$ above.

Summing up, the four requirements for a bisimulation relation are met, so $\mathcal{R}$ is indeed a bisimulation relation. This concludes our proof of the theorem. □

Contrast this result with the cancellation lemma for BPA as defined by Burkart et al. [9], where the proof proceeds by induction on the *norm* of the process; this is possible in that setting because of the assumption that a terminating state has no outgoing transitions, i.e., that impure termination is not possible. Since the norm of a process can be finite even when the process itself is infinite, this also makes it possible to specify some infinite processes.

A form of cancellation where $r \leftrightarrows \mathbf{0}$ – but with other restrictions – is possible; see Lemma 4.9 in Section 4.2. See also Appendix A for a variant of cancellation where some form of infinite behaviour is allowed.

Chapter 4

Decomposition

We will now prove a notion of unique sequential decomposition of processes in three cases. In Section 4.1 we treat the case where no deadlock is involved (i.e., in $\mathsf{BPA}_{\mathbf{1}}$). The other two cases both concern processes containing deadlock, i.e. processes in $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$. In Section 4.2 we handle processes for which all transition sequences end in deadlock, and in Section 4.3 we handle processes for which only some – but explicitly not all – transition sequences end in deadlock.

Whereas deadlock-free processes can be uniquely decomposed in such a way that each of the resulting processes in the decomposition is prime, in the other two cases (of processes containing deadlock) this is not possible, as we shall see in the following sections. As a result, we propose for such processes a slightly adapted notion of decomposition such that a unique decomposition once again becomes possible. We give a short analysis and try to justify the particular form of decomposition chosen for these processes in Section 4.4.

4.1 Deadlock-free processes

Using the cancellation theorem (Theorem 3.2), we can prove that every process in $\mathsf{BPA}_{\mathbf{1}}$ can be decomposed uniquely into prime processes. We first define these notions, then we prove that this is the case.

Definition 4.1. A process p is called *prime* if $p \not\leftrightarrows \mathbf{1}$ and for all processes q and r, we have that $p \leftrightarrows q \cdot r$ implies either $q \leftrightarrows \mathbf{1}$ or $r \leftrightarrows \mathbf{1}$.

Definition 4.2. A *sequential decomposition* of a process p is a sequence of processes $\langle p_1, p_2, \ldots, p_n \rangle$ such that $p \leftrightarrows p_1 \cdot p_2 \cdot \ldots \cdot p_n$; call $p_1 \cdot p_2 \cdot \ldots \cdot p_n$ the process associated with decomposition $\langle p_1, p_2, \ldots, p_n \rangle$. We define $\mathbf{1}$ to be the process associated with the empty sequence $\langle\rangle$, because $\mathbf{1}$ is a unit element for sequential composition.

Now we can prove the following theorem about unique prime decompositions of processes in $\mathsf{BPA}_{\mathbf{1}}$.

Theorem 4.3. *Every process p in $\mathsf{BPA}_\mathbf{1}$ has a sequential decomposition into processes $\langle p_1, p_2, \ldots, p_n\rangle$ in such a way that every p_i is prime, for all $1 \leq i \leq n$, and only one such prime decomposition exists modulo bisimulation.*

Proof. We prove both existence and uniqueness of a prime decomposition of p by induction on $|p|$.

1. If $|p| = 0$, then p cannot perform an action, so it must terminate (since we are in a setting without deadlock); thus $p \leftrightarrows \mathbf{1}$. Since there does not exist a prime process or composition of prime processes bisimilar to $\mathbf{1}$, and since $\mathbf{1}$ is by Definition 4.2 the sequential composition of no processes, we have a prime decomposition into the empty sequence $\langle\rangle$. This decomposition is trivially unique modulo bisimulation.

2. Suppose that $|p| > 0$. By induction we have a unique prime decomposition for all p' with $|p'| < |p|$. Now if p is prime, then clearly we have the singleton sequence $\langle p\rangle$ as decomposition for p. Regarding uniqueness, since p is prime, every prime decomposition of p is a singleton sequence of which the only element is bisimilar to p itself. Hence, this decomposition is unique modulo bisimulation.

   We now focus on the case where p is not prime, and prove existence of a sequential decomposition and uniqueness of that decomposition separately.

   If p is not prime, then by the definition of primeness (Definition 4.1), there exist processes q and r such that $p \leftrightarrows q \cdot r$, and $q \not\leftrightarrows \mathbf{1}$ and $r \not\leftrightarrows \mathbf{1}$, so $|q| > 0$ and $|r| > 0$. Since $|p| = |q| + |r|$ (by Lemmas 2.7 and 2.8), we have $|q| < |p|$ and $|r| < |p|$, so by induction we have a prime decomposition of q into $\langle q_1, q_2, \ldots, q_n\rangle$ and of r into $\langle r_1, r_2, \ldots r_m\rangle$. Because $p \leftrightarrows q \cdot r$, we have $\langle q_1, q_2, \ldots, q_n, r_1, r_2, \ldots, r_m\rangle$ as prime decomposition of p.

   Now for uniqueness, suppose we have two prime decompositions of p, namely $\langle x_1, x_2, \ldots, x_k\rangle$ and $\langle y_1, y_2, \ldots, y_l\rangle$. We name the processes associated with these decompositions as $x = x_1 \cdot x_2 \cdot \ldots \cdot x_k$ and $y = y_1 \cdot y_2 \cdot \ldots \cdot y_l$, respectively. We need to prove that the two prime decompositions are actually the same under bisimulation, i.e. that $k = l$ and $x_i \leftrightarrows y_i$ for all $1 \leq i \leq k$.

   To do this, take a longest transition sequence starting at p, i.e. one of length $|p|$. Since $p \leftrightarrows x \leftrightarrows y$, by Lemmas 2.8 and 2.7 we have $|p| = |x| = |x_1| + |x_2| + \ldots + |x_k| = |y| = |y_1| + |y_2| + \ldots + |y_l|$, so a transition sequence of length $|p|$ starting at x must start with a transition sequence of length $|x_1|$ starting at x_1; and similarly a transition sequence of length $|p|$ starting at y must start with a transition sequence of length $|y_1|$ starting at y_1. Since we have by definition of primeness (Definition 4.1) that $|x_1| > 0$ and $|y_1| > 0$, this means that such a longest transition sequence starting at x would necessarily start with a step $x \xrightarrow{a} x' = x_1' \cdot x_2 \cdot \ldots \cdot x_k$, for some $a \in \mathcal{A}$ and $x_1 \xrightarrow{a} x_1'$; and by bisimilarity also $y \xrightarrow{a} y' = y_1' \cdot y_2 \cdot \ldots \cdot y_l$, for $y_1 \xrightarrow{a} y_1'$ with $x' \leftrightarrows y'$.

   Now x_1' and y_1' both have a prime decomposition as well; $\langle x_{11}', x_{12}', \ldots, x_{1s}'\rangle$ and $\langle y_{11}', y_{12}', \ldots, y_{1t}'\rangle$ for x_1' and y_1' respectively (for which it is possible that $s = 0$ or $t = 0$, in which case we have the empty decomposition $\langle\rangle$). This means we get the following prime decomposition for x':

$\langle x'_{11}, x'_{12}, \ldots, x'_{1s}, x_2, \ldots, x_k\rangle$; and for y': $\langle y'_{11}, y'_{12}, \ldots, y'_{1t}, y_2, \ldots, y_l\rangle$. We know by Lemma 2.9 that $|x'| < |x|$ and $|y'| < |y|$, so by induction the decompositions of both x' and y' are unique. Since $x' \leftrightarrows y'$, this means that they are actually equal, so $s + (k-1) = t + (l-1)$, and the ith element of the decomposition of x' is bisimilar to the ith element of the decomposition of y', for all $1 \leq i \leq s + (k-1)$. Because x and y are not prime, we have that $k \geq 2$ and $l \geq 2$, so $x_k \leftrightarrows y_l$.

With $x_k \leftrightarrows y_l$ established, we can use the cancellation theorem (Theorem 3.2) on x and y to derive that $x_1 \cdot x_2 \cdot \ldots \cdot x_{k-1} \leftrightarrows y_1 \cdot y_2 \cdot \ldots \cdot y_{l-1}$. The size of both of these processes is smaller than $|x|$, since $|x_k| > 0$ and $|y_l| > 0$. This means that once again by induction decomposition is unique for $x_1 \cdot x_2 \cdot \ldots \cdot x_{k-1}$, so we have that $k = l$ and $x_i \leftrightarrows y_i$ for $1 \leq i < k$. We had already established that $x_k \leftrightarrows y_l$, so in conclusion, decomposition for p is also unique if p is not prime.

For both $|p| = 0$ and $|p| > 0$ we have found a prime decomposition for p and established its uniqueness. □

4.2 Always-deadlocking processes

We now consider processes which always end in deadlock, and thus can never terminate successfully. We call these processes *always-deadlocking*, and define them formally as follows.

Definition 4.4. A process p is *always-deadlocking* if there does not exist a process p' such that $p \longrightarrow^* p'$ with $p'\downarrow$.

For these processes we do not have a similar unique prime decomposition result as for processes without deadlock. This is easy to see: the process $\mathbf{0}$ can be decomposed as $\langle \mathbf{0}\rangle, \langle \mathbf{0}, \mathbf{0}\rangle, \langle \mathbf{0}, \mathbf{0}, \mathbf{0}\rangle$, etcetera, but none of these decompositions are prime, since they can always be decomposed further. The same actually holds for all processes that always end in deadlock. Take a process like $a.\mathbf{0} + b.\mathbf{0}$; this can be decomposed as $\langle a.\mathbf{1} + b.\mathbf{1}, \mathbf{0}\rangle$, or as $\langle a.\mathbf{1} + b.\mathbf{1}, \mathbf{0}, \mathbf{0}\rangle$, and so on.

Thus processes always ending in deadlock actually do not have a prime decomposition at all. However, we shall prove that all always-deadlocking processes have a decomposition in the form $\langle p_1, p_2, \ldots, p_n, \mathbf{0}\rangle$, where p_i is deadlock-free and prime, for all $1 \leq i \leq n$.

Unfortunately, this kind of decomposition is not always unique, as demonstrated by the following lemma.

Lemma 4.5. *For all processes p, it holds that $(p + \mathbf{1}) \cdot \mathbf{0} \leftrightarrows p \cdot \mathbf{0}$.*

Proof. By applying the axioms from Lemma 2.5, we calculate for every p that $(p + \mathbf{1}) \cdot \mathbf{0} \leftrightarrows (p \cdot \mathbf{0}) + (\mathbf{1} \cdot \mathbf{0}) \leftrightarrows (p \cdot \mathbf{0}) + \mathbf{0} \leftrightarrows p \cdot \mathbf{0}$. □

This means that for deadlock-free processes p where $p + \mathbf{1} \not\leftrightarrows p$, the process $p \cdot \mathbf{0}$ has at least two distinct decompositions of the form proposed above, so

decomposition is not unique. However, we can solve this by requiring that in the decomposition $\langle p_1, p_2, \ldots, p_n, \mathbf{0}\rangle$, the process p_n does not have impure termination, as stated in Definition 2.3. If this is the case, then decomposition turns out to be unique, which we shall prove. However, we first need some auxiliary lemmas.

First of all, we prove that each process can be written without the sequential composition operator. The process algebra consisting only of the successful termination and deadlock constants, and the action prefix and alternative composition operators, is known as $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, with BSP standing for Basic Sequential Processes (perhaps confusingly, since it contains no sequential composition operator). So removing the sequential composition operator from a process yields a process in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$.

Lemma 4.6. *For every process p in $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ there is a process q in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$ such that $p \leftrightarrows q$.*

Proof. We prove the lemma by structural induction on p. We distinguish for p:

1. Suppose $p \leftrightarrows \mathbf{0}$. Then p is already in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, so take $q = p$.
2. Suppose $p \leftrightarrows \mathbf{1}$. Then p is also already in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, so take $q = p$.
3. Suppose $p \leftrightarrows a.p'$ for some action $a \in \mathcal{A}$ and some process p'. Then by induction, there is a $q' \leftrightarrows p'$ that is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, so then $a.q' \leftrightarrows a.p'$ and $a.q'$ is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, since action prefix is allowed; so we can take $q = a.q'$.
4. Suppose $p \leftrightarrows p_1 + p_2$ for some processes p_1 and p_2. Then by induction, there are $q_1 \leftrightarrows p_1$ and $q_2 \leftrightarrows p_2$ such that q_1 and q_2 are in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$. Then $q_1 + q_2 \leftrightarrows p_1 + p_2$, and $q_1 + q_2$ is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, since alternative composition is allowed; so we take $q = q_1 + q_2$.
5. Suppose $p \leftrightarrows p_1 \cdot p_2$ for some processes p_1 and p_2. Then by induction, there are $q_1 \leftrightarrows p_1$ and $q_2 \leftrightarrows p_2$ such that q_1 and q_2 are in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$. Then $q_1 \cdot q_2 \leftrightarrows p_1 \cdot p_2$, but $q_1 \cdot q_2$ is not in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, so we distinguish further on q_1:
   (a) Suppose $q_1 \leftrightarrows \mathbf{0}$. Then $q_1 \cdot q_2 \leftrightarrows \mathbf{0} \cdot q_2 \leftrightarrows \mathbf{0}$; and $\mathbf{0}$ is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, so we take $q = \mathbf{0}$.
   (b) Suppose $q_1 \leftrightarrows \mathbf{1}$. Then $q_1 \cdot q_2 \leftrightarrows \mathbf{1} \cdot q_2 \leftrightarrows q_2$; and q_2 is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$, so we take $q = q_2$.
   (c) Suppose $q_1 \leftrightarrows a.q_1'$ for some action $a \in \mathcal{A}$ and some process q_1'. Then $q_1 \cdot q_2 \leftrightarrows (a.q_1') \cdot q_2 \leftrightarrows a.(q_1' \cdot q_2)$, and by induction there is a process $r \leftrightarrows q_1' \cdot q_2$, where r is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$. Then $a.r$ is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$ as well, and $a.r \leftrightarrows q_1 \cdot q_2$, so we take $q = a.r$.
   (d) Suppose $q_1 \leftrightarrows q_{11} + q_{12}$ for some processes q_{11} and q_{12}. Then $q_1 \cdot q_2 \leftrightarrows (q_{11} + q_{12}) \cdot q_2 \leftrightarrows (q_{11} \cdot q_2) + (q_{12} \cdot q_2)$. Now by induction there are processes $r_1 \leftrightarrows q_{11} \cdot q_2$ and $r_2 \leftrightarrows q_{12} \cdot q_2$ such that r_1 and r_2 are in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$. Then $r_1 + r_2$ is in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$ as well, and $r_1 + r_2 \leftrightarrows q_1 \cdot q_2$, so we take $q = r_1 + r_2$.

   In each case there is a q in $\mathsf{BSP}_{\mathbf{0},\mathbf{1}}$ such that $q \leftrightarrows p_1 \cdot p_2$.

All cases have been covered, so the lemma holds. □

The following auxiliary lemma is used by the two lemmas immediately following this one.

Lemma 4.7. *If p is a process that does not have impure termination, and $p\downarrow$, then $p \leftrightarrows \mathbf{1}$.*

Proof. Since $p\downarrow$, p can terminate. Hence, since p does not have impure termination, p is not allowed to perform an action. The only processes that cannot perform an action are bisimilar either to $\mathbf{0}$ or $\mathbf{1}$; and since $\mathbf{0}$ cannot terminate, it must be that $p \leftrightarrows \mathbf{1}$. □

The next lemma states that it is possible to write every always-deadlocking process in such a way that the deadlock part is "split off" on the right side, i.e., in the form that is required by the decomposition we want to prove.

Lemma 4.8. *For every process p in $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ where all transition sequences starting at p end in deadlock, there exists a deadlock-free process r such that $p \leftrightarrows r \cdot \mathbf{0}$ and r has no impure termination.*

Proof. We prove this by structural induction on p. First of all we can assume (by Lemma 4.6) without loss of generality that p does not contain the sequential composition operator. Then we distinguish the following cases for p:

1. Suppose $p \leftrightarrows \mathbf{0}$. Then we need an r such that $\mathbf{0} \leftrightarrows r \cdot \mathbf{0}$, so we can take $r = \mathbf{1}$ (which is deadlock-free and has no impure termination).

2. Suppose $p \leftrightarrows \mathbf{1}$. This breaks the assumption that all transition sequences starting at p end in deadlock, so this case does not occur.

3. Suppose $p \leftrightarrows a.p'$ for some action $a \in \mathcal{A}$ and some process p', where all transition sequences starting at $a.p'$ end in deadlock. Then all transition sequences starting at p' end in deadlock as well, so by induction there is a deadlock-free process r' without impure termination such that $p' \leftrightarrows r' \cdot \mathbf{0}$. Now $a.p' \leftrightarrows a.(r' \cdot \mathbf{0}) \leftrightarrows (a.r') \cdot \mathbf{0}$; and $a.r'$ is deadlock-free (because r' is deadlock-free), and it has no impure termination (since $a.r'$ cannot terminate, but can only make a step to r', which has no impure termination), so we can take $r = a.r'$.

4. Suppose $p \leftrightarrows p_1 + p_2$ for some processes p_1 and p_2, where all transition sequences starting at $p_1 + p_2$ end in deadlock. Then all transition sequences starting at p_1 and all transition sequences starting at p_2 end in deadlock as well, so by induction there are deadlock-free processes r_1 and r_2, both without impure termination, such that $p_1 \leftrightarrows r_1 \cdot \mathbf{0}$ and $p_2 \leftrightarrows r_2 \cdot \mathbf{0}$. Now $p_1 + p_2 \leftrightarrows (r_1 \cdot \mathbf{0}) + (r_2 \cdot \mathbf{0}) \leftrightarrows (r_1 + r_2) \cdot \mathbf{0}$; and $r_1 + r_2$ is deadlock-free (since both r_1 and r_2 are deadlock-free). Since r_1 and r_2 individually are without impure termination, the only way for $r_1 + r_2$ to have impure termination is if $r_1\downarrow$ and $r_2 \xrightarrow{a} r_2'$ for some $a \in \mathcal{A}$, or vice versa.

   Now one of the following three cases holds:

(a) Suppose that $r_1\downarrow$ and $r_2 \xrightarrow{a} r_2'$. Since r_1 has no impure termination, by Lemma 4.7 it can only terminate if $r_1 \leftrightarrows \mathbf{1}$. Then we have $p \leftrightarrows (\mathbf{1} + r_2) \cdot \mathbf{0}$, and by Lemma 4.5, $p \leftrightarrows r_2 \cdot \mathbf{0}$. So we can take $r = r_2$, which is deadlock-free and without impure termination.

(b) Suppose that $r_1 \xrightarrow{a} r_1'$ and $r_2\downarrow$. Then this case is symmetrical to the one above, so we take $r = r_1$.

(c) Suppose neither $r_1\downarrow$ and $r_2 \xrightarrow{a} r_2'$, nor $r_1 \xrightarrow{a} r_1'$ and $r_2\downarrow$. Then $r_1 + r_2$ has no impure termination, and since $p \leftrightarrows (r_1 + r_2) \cdot \mathbf{0}$, we can take $r = r_1 + r_2$.

In all three cases we have a suitable r.

All cases have been covered, so the lemma holds. □

The following lemma is a variant of the cancellation theorem (Theorem 3.2), for the case where $r \leftrightarrows \mathbf{0}$. Since this case does not fall under the assumptions present in the general cancellation theorem (i.e., that p, q deadlock-free and $r \not\leftrightarrows \mathbf{0}$), it is proved separately here. Note that it is required that both processes are without impure termination, since otherwise the lemma does not hold: e.g. $(a.\mathbf{1} + \mathbf{1}) \cdot \mathbf{0} \leftrightarrows a.\mathbf{1} \cdot \mathbf{0}$, but clearly $a.\mathbf{1} + \mathbf{1} \not\leftrightarrows a.\mathbf{1}$ (see also Lemma 4.5). So this lemma might seem trivial at first, but it is not.

Lemma 4.9. *If for some deadlock-free processes p and q, both without impure termination, it holds that $p \cdot \mathbf{0} \leftrightarrows q \cdot \mathbf{0}$, then $p \leftrightarrows q$.*

Proof. This proof largely follows the structure of the proof of Theorem 3.2. Consider the following relation $\mathcal{R}$:

$$\mathcal{R} = \{(p, q) \mid p, q \text{ deadlock-free, without impure termination} \wedge p \cdot \mathbf{0} \leftrightarrows q \cdot \mathbf{0}\} .$$

If $\mathcal{R}$ is a bisimulation relation, then the theorem holds, so we need to prove that each of the four properties of a bisimulation relation hold for $\mathcal{R}$. For each property, assume $(p, q) \in \mathcal{R}$ with p, q deadlock-free and without impure termination.

1. Suppose $p \xrightarrow{a} p'$ for some $a \in \mathcal{A}$; we need to show that there exists a q' such that $q \xrightarrow{a} q'$ and $(p', q') \in \mathcal{R}$.

   By SOS rule 7 (see Table 2.1) on $p \xrightarrow{a} p'$ we have $p \cdot \mathbf{0} \xrightarrow{a} p' \cdot \mathbf{0}$, so by bisimilarity of $p \cdot \mathbf{0}$ and $q \cdot \mathbf{0}$ there exists an $s \leftrightarrows p' \cdot \mathbf{0}$ such that $q \cdot \mathbf{0} \xrightarrow{a} s$. Fix a derivation of $q \cdot \mathbf{0} \xrightarrow{a} s$; we distinguish cases according to which SOS rule has been applied last in this derivation. This can either be rule 7 or rule 8:

   (a) By SOS rule 7 there is a q' such that $q \xrightarrow{a} q'$; then $s = q' \cdot \mathbf{0}$, so $p' \cdot \mathbf{0} \leftrightarrows q' \cdot \mathbf{0}$. By definition of $\mathcal{R}$, since $p' \cdot \mathbf{0} \leftrightarrows q' \cdot \mathbf{0}$, and p', q' deadlock-free and without impure termination (since p, q deadlock-free and without impure termination), this means that $(p', q') \in \mathcal{R}$.

   (b) By SOS rule 8 it holds that $q\downarrow$ and there is a process r such that $\mathbf{0} \xrightarrow{a} r$; but obviously $\mathbf{0}$ cannot perform an action, so this case cannot occur.

Only the first case can occur, so there is a q' such that $q \xrightarrow{a} q'$ with $(p', q') \in \mathcal{R}$; this satisfies the bisimulation relation requirement.

2. Suppose $q \xrightarrow{a} q'$ for some $a \in \mathcal{A}$; we need to show that there exists a p' such that $p \xrightarrow{a} p'$ and $(p', q') \in \mathcal{R}$.

   This is analogous to the case for $p \xrightarrow{a} p'$ above.

3. Suppose $p\downarrow$; we need to show that $q\downarrow$.

   If $p\downarrow$, then because p has no impure termination, by Lemma 4.7 we have $p \leftrightarrows \mathbf{1}$, so $p \cdot \mathbf{0} \leftrightarrows \mathbf{0}$; thus we also have $q \cdot \mathbf{0} \leftrightarrows \mathbf{0}$. This is only possible if q cannot perform any action, and since q is deadlock-free, it must be that $q \leftrightarrows \mathbf{1}$. So we also have $q\downarrow$; this satisfies the bisimulation relation requirement.

4. Suppose $q\downarrow$; we need to show that $p\downarrow$.

   This is analogous to the case for $p\downarrow$ above.

Summing up, the four requirements for a bisimulation relation are met, so $\mathcal{R}$ is indeed a bisimulation relation. Thus the lemma holds. □

Now we can prove the following theorem about unique prime decompositions of always-deadlocking processes.

Theorem 4.10. *Every process p in $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$, where all transition sequences starting at p end in deadlock, has a sequential decomposition $\langle p_1, p_2, \ldots, p_n, \mathbf{0}\rangle$ in such a way that every p_i is deadlock-free and prime, for all $1 \leq i \leq n$, and p_n does not have impure termination; and only one such decomposition exists modulo bisimulation.*

Proof. The existence of a decomposition in this form is readily established: by Lemma 4.8 there exists a deadlock-free r without impure termination such that $p \leftrightarrows r \cdot \mathbf{0}$. Since r is deadlock-free, it can be decomposed by Theorem 4.3 into the unique prime decomposition $\langle r_1, r_2, \ldots, r_n\rangle$. Now suppose that r_n has impure termination. Since r can reach r_n, then r would have impure termination as well. But r does not have impure termination, so neither does r_n.[1] Now $\langle r_1, r_2, \ldots, r_n, \mathbf{0}\rangle$ is a decomposition of p satisfying the above requirements.

As for uniqueness, suppose we have two decompositions of p that satisfy the properties as stated in the Theorem: $\langle x_1, x_2, \ldots, x_k, \mathbf{0}\rangle$ and $\langle y_1, y_2, \ldots, y_l, \mathbf{0}\rangle$. We name the processes associated with these decompositions, *without* the deadlock part, as $x = x_1 \cdot x_2 \cdot \ldots \cdot x_k$ and $y = y_1 \cdot y_2 \cdot \ldots \cdot y_l$, respectively. Since $p \leftrightarrows x \cdot \mathbf{0} \leftrightarrows y \cdot \mathbf{0}$ and x and y do not have impure termination, we know by Lemma 4.9 that $x \leftrightarrows y$. Since they are both deadlock-free, we have by Theorem 4.3 that the prime decompositions of x and y are equal, since prime decomposition is unique for processes without deadlock; so $k = l$ and $x_i \leftrightarrows y_i$ for all $1 \leq i \leq k$. Hence the two decompositions of p are also equal; so the required decomposition of p is unique. □

[1] Note that $r_1, \ldots, r_{n-1}$ can still have impure termination, since that does not result in impure termination for r.

4.3 Sometimes-deadlocking processes

We now turn our attention to processes which have deadlock, but also have successful termination; i.e. processes which sometimes – but not always – end in deadlock. We call these processes *sometimes-deadlocking*, and define them formally as follows.

Definition 4.11. A process p is *sometimes-deadlocking* if, for some processes p' and p'', we have $p \longrightarrow^* p'$ with $p'\downarrow$ and also $p \longrightarrow^* p''$ with $p'' \leftrightarrows \mathbf{0}$.

An example of such a process is $a.\mathbf{0} + a.\mathbf{1}$, which either performs an a-action and then deadlocks, or performs an a-action and then successfully terminates.

When decomposing a sometimes-deadlocking process, at least one of the resulting processes will contain deadlock. However, to obtain a unique decomposition result we will require that the resulting decomposition contains only one deadlocking process, where this process is the rightmost one in the decomposition. That is, in order to obtain this decomposition, we take a sometimes-deadlocking process and repeatedly "split off" a deadlock-free prime process on the left-hand side, until this is not possible anymore. For an argument as to why we only split off deadlock-free processes, see Section 4.4.

Similar to the case of always-deadlocking processes, it is not possible to establish a unique prime decomposition result where all elements of the decomposition are prime: see e.g. the process $a.\mathbf{0}+\mathbf{1}$, which is sometimes-deadlocking (since it is possible to terminate immediately, or perform an a-action and then deadlock), but it does not have a prime decomposition, by the following lemma.

Lemma 4.12. *The sometimes-deadlocking process $a.\mathbf{0}+\mathbf{1}$ does not have a prime decomposition.*

Proof. We prove the lemma by contradiction. Suppose there are processes $p_1, p_2, \ldots, p_n$ such that p_i is prime for all $1 \leq i \leq n$ and $a.\mathbf{0}+\mathbf{1} \leftrightarrows p_1 \cdot p_2 \cdot \ldots \cdot p_n$. It cannot be the case that $n = 0$, since that would yield the empty decomposition, which is bisimilar to $\mathbf{1}$, and clearly not bisimilar to $a.\mathbf{0} + \mathbf{1}$. So $n \geq 1$, thus at least p_1 exists. Now take a process q such that $q \leftrightarrows p_2 \cdot \ldots \cdot p_n$, then $p \leftrightarrows p_1 \cdot q$. Note that since $a.\mathbf{0}+\mathbf{1}$ can terminate, it must be the case that both p_1 and q have a termination option, so $p_1\downarrow$ and $q\downarrow$ must hold.

Now since p can perform an a-action to $\mathbf{0}$, $p_1 \cdot q$ must also be able to do so. It cannot be the case that $p_1 \leftrightarrows \mathbf{0}$ or $p_1 \leftrightarrows \mathbf{1}$, since $\mathbf{0}$ and $\mathbf{1}$ are not prime, so it must be the case that the a-action (which is the only action) comes from p_1; so we have $p_1 \xrightarrow{a} p_1'$ for some process p_1'.

Then we have $p_1 \cdot q \xrightarrow{a} p_1' \cdot q$, where $p_1' \cdot q$ should deadlock. This is the case if none of the SOS rules are applicable on $p_1' \cdot q$, i.e. when $p_1' \leftrightarrows \mathbf{1}$ and $q \leftrightarrows \mathbf{0}$, or when $p_1' \leftrightarrows \mathbf{0}$. The first case is not possible, since we should have that $q\downarrow$, and $\mathbf{0}\downarrow$ does not hold. This leaves us with the second case, where $p_1' \leftrightarrows \mathbf{0}$.

We now have that $p_1\downarrow$ and $p_1 \xrightarrow{a} \mathbf{0}$; this is the only behaviour that p_1 can have, since any additional behaviour would also be apparent in p. Hence it holds that $p_1 \leftrightarrows a.\mathbf{0} + \mathbf{1}$. However, this process is not prime, since it can easily be shown that $a.\mathbf{0} + \mathbf{1} \leftrightarrows (a.\mathbf{0} + \mathbf{1}) \cdot (a.\mathbf{0} + \mathbf{1})$.

There is no way to derive a process $p_1 \cdot p_2 \cdot \ldots \cdot p_n$ bisimilar to $a.\mathbf{0}+\mathbf{1}$ such that p_1 is prime, hence no prime decomposition for $a.\mathbf{0}+\mathbf{1}$ exists. □

Since $a.\mathbf{0}+\mathbf{1}$ is a sometimes-deadlocking process, we can conclude that not all sometimes-deadlocking processes have a prime decomposition, just like in the case of always-deadlocking processes. Hence we propose an alternative decomposition that does hold for all sometimes-deadlocking processes. Like in the always-deadlocking case, this decomposition is one where all but the last component is prime; in this case, we require that the last component is *deadlock-prime*.

Definition 4.13. A process p is called *deadlock-prime* if $p \not\leftrightarrows \mathbf{1}$, and whenever we have that $p \leftrightarrows q \cdot r$ for some processes q and r, then either $q \leftrightarrows \mathbf{1}$ or $r \leftrightarrows \mathbf{1}$, or q has deadlock.

With this notion of deadlock-primeness, we shall prove that all sometimes-deadlocking processes in $\mathsf{BPA}_{\mathbf{0,1}}$ can be decomposed as $\langle p_1, p_2, \ldots, p_n \rangle$, where p_i is deadlock-free and prime, for all $1 \leq i < n$, and p_n is deadlock-prime; and that this decomposition is unique.

Theorem 4.14. *Every sometimes-deadlocking process p in $\mathsf{BPA}_{\mathbf{0,1}}$ has a sequential decomposition into processes $\langle p_1, p_2, \ldots, p_n \rangle$ in such a way that every p_i is deadlock-free and prime, for all $1 \leq i < n$, and p_n is deadlock-prime; and only one such decomposition exists modulo bisimulation.*

Proof. We prove by induction on $|p|$ that if p is sometimes-deadlocking, then the decomposition exists and it is unique. The proof structure largely follows that of the proof of Theorem 4.3.

1. If $|p| = 0$, then either $p \leftrightarrows \mathbf{0}$ or $p \leftrightarrows \mathbf{1}$, both of which are not sometimes-deadlocking, so we do not need to prove that a decomposition exists.

2. Suppose $|p| > 0$. Then we know by induction for all p' with $|p'| < |p|$, that if p' is sometimes-deadlocking, then it has a unique decomposition in the required format. Suppose that p is sometimes-deadlocking, then we need to prove that such a decomposition for p exists, and that it is unique. We now distinguish on p.

   (a) If there are no processes q and r with $q \not\leftrightarrows \mathbf{1}$ and $r \not\leftrightarrows \mathbf{1}$ such that $p \leftrightarrows q \cdot r$ and q is deadlock-free, then p is deadlock-prime by Definition 4.13, and thus $\langle p \rangle$ is a decomposition of p in the required format. Since there is only one such decomposition, it is unique.

   (b) If there are processes q and r with $q \not\leftrightarrows \mathbf{1}$ and $r \not\leftrightarrows \mathbf{1}$, such that $p \leftrightarrows q \cdot r$ and q is deadlock-free, then q can be decomposed by Theorem 4.3 into $\langle q_1, q_2, \ldots, q_n \rangle$, where q_i is deadlock-free and prime for all $1 \leq i \leq n$. Because q is deadlock-free, we have by Lemmas 2.8 and 2.7 that $|p| = |q \cdot r| = |q| + |r|$. We know that $q \not\leftrightarrows \mathbf{1}$ and, since q is deadlock-free, also $q \not\leftrightarrows \mathbf{0}$, so it must be the case that q can perform an action; hence $|q| > 0$ and thus $|r| < |p|$. This means we can apply the induction hypothesis on r. Since p is sometimes-deadlocking and q is deadlock-free, it must be the case that r is sometimes-deadlocking,

otherwise $q \cdot r$ would not be sometimes-deadlocking. Thus r has a unique decomposition into $\langle r_1, r_2, \ldots, r_m \rangle$, where r_i is deadlock-free and prime for all $1 \leq i < m$, and r_m is deadlock-prime. Combining these results, we have $\langle q_1, q_2, \ldots, q_n, r_1, r_2, \ldots, r_m \rangle$ as a decomposition for p in the required format.

Now for uniqueness, suppose we have two decompositions of p, namely $\langle x_1, x_2, \ldots, x_k \rangle$ and $\langle y_1, y_2, \ldots, y_l \rangle$, with x_i prime for all $1 \leq i < k$ and y_i prime for all $1 \leq i < l$, and x_k and y_l deadlock-prime. We name the processes associated with these decompositions as $x = x_1 \cdot x_2 \cdot \ldots \cdot x_k$ and $y = y_1 \cdot y_2 \cdot \ldots \cdot y_l$, respectively. We need to prove that the two prime decompositions are actually the same under bisimulation, i.e. that $k = l$ and $x_i \leftrightarrows y_i$ for all $1 \leq i \leq k$.

To do this, take the longest transition sequence starting at p, i.e. one of length $|p|$. Since $p \leftrightarrows x \leftrightarrows y$, by Lemmas 2.8 and 2.7 (which are applicable here, since only the rightmost component of the decomposition contains deadlock) we have $|p| = |x| = |x_1| + |x_2| + \ldots + |x_k| = |y| = |y_1| + |y_2| + \ldots + |y_l|$, so a transition sequence of length $|p|$ starting at x must start with a transition sequence of length $|x_1|$ starting at x_1; and similarly a transition sequence of length $|p|$ starting at y must start with a transition sequence of length $|y_1|$ starting at y_1. Since we have by definition of primeness (Definition 4.1) that $|x_1| > 0$ and $|y_1| > 0$, this means that such a longest transition sequence starting at x would necessarily start with a step $x \xrightarrow{a} x' = x_1' \cdot x_2 \cdot \ldots \cdot x_k$, for some $a \in \mathcal{A}$ and $x_1 \xrightarrow{a} x_1'$; and by bisimilarity also $y \xrightarrow{a} y' = y_1' \cdot y_2 \cdot \ldots \cdot y_l$, for $y_1 \xrightarrow{a} y_1'$ with $x' \leftrightarrows y'$.

Since we are in the case where $p \leftrightarrows q \cdot r$, every decomposition must have at least two components, thus $k \geq 2$ and $l \geq 2$; so we know by definition of the decomposition that x_1 and y_1 are deadlock-free and prime. Because x_1 and y_1 do not contain deadlock, neither do x_1' and y_1'; hence x_1' has a prime decomposition into $\langle x_{11}', x_{12}', \ldots, x_{1s}' \rangle$ and y_1' into $\langle y_{11}', y_{12}', \ldots, y_{1t}' \rangle$ (for which it is possible that $s = 0$ or $t = 0$, in which case we have the empty decomposition $\langle\rangle$). This means we get as decomposition for x': $\langle x_{11}', x_{12}', \ldots, x_{1s}', x_2, \ldots, x_k \rangle$, and as decomposition for y': $\langle y_{11}', y_{12}', \ldots, y_{1t}', y_2, \ldots, y_l \rangle$. We know by Lemma 2.9 that $|x'| < |x|$ and $|y'| < |y|$, so by induction the decompositions of both x' and y' are unique. Since $x' \leftrightarrows y'$, this means that they are actually equal, so $s + (k-1) = t + (l-1)$, and the ith element of the decomposition of x' is bisimilar to the ith element of the decomposition of y', for all $1 \leq i \leq s + (k-1)$, and in particular also $x_k \leftrightarrows y_l$.

With $x_k \leftrightarrows y_l$ established, we can use the cancellation theorem (Theorem 3.2) on x and y to derive that $x_1 \cdot x_2 \cdot \ldots \cdot x_{k-1} \leftrightarrows y_1 \cdot y_2 \cdot \ldots \cdot y_{l-1}$. The size of both of these processes is smaller than $|x|$, since $|x_k| > 0$ and $|y_l| > 0$. This means that once again by induction the required decomposition of $x_1 \cdot x_2 \cdot \ldots \cdot x_{k-1}$ is unique, so we have that $k = l$ and $x_i \leftrightarrows y_i$ for $1 \leq i < k$. We had already established that $x_k \leftrightarrows y_l$, so in conclusion, the decomposition of p into the format required by the theorem is unique.

In both cases we have established that if p is sometimes-deadlocking, then it has a unique decomposition following the definition in the theorem.

Both cases $|p| = 0$ and $|p| > 0$ have been covered, so the theorem holds. □

While some processes (such as $a.\mathbf{0} + \mathbf{1}$) can be infinitely decomposed into copies of themselves, this is not the case for all sometimes-deadlocking processes: there exist processes which are prime (so cannot be further decomposed) and yet have deadlock as well. We show this by giving an example of such a process.

Lemma 4.15. *The process $p = a.(b.\mathbf{0} + \mathbf{1}) + c.\mathbf{1}$ has deadlock and is prime.*

Proof. Obviously this process has deadlock; we prove by contradiction that it is prime. Suppose that p is not prime, then we have $p \leftrightarrows q \cdot r$ for some processes q and r, with $q \not\leftrightarrows \mathbf{1}$ and $r \not\leftrightarrows \mathbf{1}$. We also have $q \not\leftrightarrows \mathbf{0}$ and $r \not\leftrightarrows \mathbf{0}$, since then all transition sequences starting at p would end in deadlock, which is not the case. So both q and r should be able to perform at least one action. In p the actions that can be taken are a and c, so q should be able to perform at least one of those actions. We now distinguish on the actions that q can perform.

1. Suppose q can perform both actions a and c. Since p can terminate both after a and c, q must provide a termination option after both a and c. Now r must be able to perform b, since r must be able to perform at least one action; but by the termination in q, this b will be reachable both after a and after c, which is not the case in p.

2. Suppose q can perform action a, but not c. Then c must be performed by r. Since p must be able to terminate after a, q must also be able to terminate after a; so it is possible to take action a from q and then action c from r. But this is not possible in p.

3. Suppose q can perform action c, but not a. Then a must be performed by r. By the same reasoning as above, it is possible to perform action c from q and then action a from r, but this is not possible in p.

Neither of the cases is possible, so no such q exists. Hence it must be the case that p is prime. □

The existence of processes which have deadlock and are prime is important, because it shows us that it might be possible to establish a decomposition for sometimes-deadlocking processes where deadlock is allowed to occur in components other than the last one. However, for now we leave the decompositions as is, and give some motivation why we do so in the next section.

4.4 Analysis of decompositions with deadlock

While the decompositions of always- and sometimes-deadlocking processes are certainly useful due to their uniqueness property, we might question whether

only allowing deadlock in the last component of the decomposition is not too strict.

For example, the process $a.\mathbf{0}+a.(b.\mathbf{1}+c.\mathbf{1})$ has – according to the decomposition definition for sometimes-deadlocking processes – a decomposition into just the single component $\langle a.\mathbf{0}+a.(b.\mathbf{1}+c.\mathbf{1})\rangle$, but one could argue that a decomposition into multiple components, like $\langle a.\mathbf{0}+\mathbf{1}, a.\mathbf{1}, b.\mathbf{1}+c.\mathbf{1}\rangle$, would be more natural. Even though this might be the case, there are several arguments against allowing deadlock in more than just the last component, because they all result in several decompositions, which invalidates the uniqueness property. The following cases illustrate this:

1. Every component consisting of deadlock can be infinitely decomposed: $\mathbf{0}$ can be decomposed into $\langle\mathbf{0}\rangle$, $\langle\mathbf{0},\mathbf{0}\rangle$, $\langle\mathbf{0},\mathbf{0},\mathbf{0}\rangle$, ...

2. Some processes can be infinitely decomposed into copies of themselves: $a.\mathbf{0}+\mathbf{1}$ can be decomposed into $\langle a.\mathbf{0}+\mathbf{1}\rangle$, $\langle a.\mathbf{0}+\mathbf{1}, a.\mathbf{0}+\mathbf{1}\rangle$, ...

3. A transition sequence ending in deadlock in an earlier component can often be repeated in the next component: $a.\mathbf{0}+b.c.\mathbf{1}$ can be decomposed into $\langle a.\mathbf{0}+\mathbf{1}, b.\mathbf{1}, c.\mathbf{1}\rangle$ but also into $\langle a.\mathbf{0}+\mathbf{1}, a.\mathbf{0}+b.\mathbf{1}, c.\mathbf{1}\rangle$.

4. A transition sequence ending in deadlock in a later component can sometimes also be repeated in an earlier component: $a.\mathbf{0}+a.\mathbf{1}+a.(a.\mathbf{0}+a.\mathbf{1})$ can be decomposed into $\langle a.\mathbf{1}+\mathbf{1}, a.\mathbf{0}+a.\mathbf{1}\rangle$, but also into $\langle a.\mathbf{0}+a.\mathbf{1}+\mathbf{1}, a.\mathbf{0}+a.\mathbf{1}\rangle$ (or into $\langle a.\mathbf{1}+\mathbf{1}, a.\mathbf{0}+\mathbf{1}, a.\mathbf{1}\rangle$).

5. Some processes can even be decomposed in commutative ways: the process $a.\mathbf{0}+b.\mathbf{0}+\mathbf{1}$ can be decomposed into $\langle a.\mathbf{0}+\mathbf{1}, b.\mathbf{0}+\mathbf{1}\rangle$, but also into $\langle b.\mathbf{0}+\mathbf{1}, a.\mathbf{0}+\mathbf{1}\rangle$.

All these cases are barred by only allowing deadlock to appear in the last component, hence we have chosen for decompositions following this rule. However, this is not an exhaustive proof that there might not be some way to unique decomposition with deadlock also occurring in other components.

Chapter 5

Generalisation of decomposition to monoids

We can generalise our unique prime decomposition result for processes in $\mathsf{BPA}_\mathbf{1}$ to *monoids*: abstract sets which have a distinguished identity element and an associative binary operator. We do this by giving a number of properties that should hold for a monoid, in order for each of its elements to have a unique prime decomposition.

Providing such a generalisation to monoids is useful in several ways. For one, by finding a minimal set of properties for which unique prime decomposition holds, we can clearly see which properties of $\mathsf{BPA}_\mathbf{1}$ are actually required for the decomposition result, and which are not. This shows us, for example, that in $\mathsf{BPA}_\mathbf{1}$ only the sequential composition operator itself is needed for decomposition, and none of the other operators play a role (as long as their semantics do not interfere with the required properties). This brings us to the next advantage, namely that we can now change the algebra in any way which preserves the properties, and also retain the decomposition result. Hence, we could add or remove operators from the algebra, change the information present in the transition steps, or even come up with some structure that is not a process algebra at all; as long as the properties given here are satisfied, unique prime decomposition is automatically proven.

The results in this chapter are inspired by Luttik and Van Oostrom's paper on Decomposition Orders [17], which gives a decomposition result for commutative monoids. Our work here differs in that we focus on monoids which are non-commutative (or do not have to be commutative) instead.

A monoid can formally be defined as follows [14]:

Definition 5.1. Consider a set M with a distinguished element $e \in M$ and a total binary operator $\_\cdot\_ : M \times M \to M$ such that for all $p, q, r \in M$ the following properties hold:

1. $p \cdot (q \cdot r) = (p \cdot q) \cdot r$ (associativity) ,
2. $p \cdot e = e \cdot p = p$ (identity) .

Then $(M, \cdot, e)$ is a *monoid*.

Note. We normally apply associativity implicitly, so we write e.g. $p \cdot q \cdot r$ instead of either $p \cdot (q \cdot r)$ or $(p \cdot q) \cdot r$.

In Section 5.1 we formulate a number of properties sufficient to make a *unique decomposition monoid*; a monoid for which a unique prime decomposition property is provable. We then prove that this is the case in Section 5.2. In Section 5.3 we provide an argument for why cancellation has to be stated as one of the unique decomposition monoid properties, and cannot be derived from the other properties, thereby proving that such a property is necessary. Finally, in Section 5.4 we prove that $\mathsf{BPA}_\mathbf{1}$ processes modulo bisimilarity constitute a unique decomposition monoid.

5.1 Unique decomposition monoids

We are now going to constrain the definition of a general monoid M as defined above in such a way that we get a *unique decomposition monoid*, for which it is possible to define a notion of unique prime decomposition in a similar way as it has been defined for processes in $\mathsf{BPA}_\mathbf{1}$ in Theorem 4.3; i.e., that each of the elements of the monoid has a unique prime decomposition. We shall then proceed to prove that the set of all processes in $\mathsf{BPA}_\mathbf{1}$ form a unique decomposition monoid, thereby proving once more the existence of a unique prime decomposition for processes in $\mathsf{BPA}_\mathbf{1}$. However, the result obtained here is stronger, as any other system satisfying the criteria for a unique decomposition monoid automatically gets a unique prime decomposition result as well.

To be able to define unique prime decomposition on monoids, we also need a size function $|\_| : M \to \mathbb{N}$, which gives each element a natural size, and a binary relation $\to \subseteq M \times M$ which relates each element except for e to one or more strictly smaller elements. For example, one might take the natural numbers with addition and 0 as identity element, the element itself as its size, and with each element except for 0 relating to the one-smaller element; or take a set of strings with the empty string as identity element, the length of a string as its size, and with each non-empty string relating to a string which is equal except with its first element removed.

The properties required for a monoid to have a unique prime decomposition are then as follows.

Definition 5.2. Consider a monoid $(M, \cdot, e)$, with an additional size function $|\_| : M \to \mathbb{N}$, and a binary relation $\to \subseteq M \times M$, for which the following properties hold, in addition to the two monoid properties stated in Definition 5.1, for all $p, q, r \in M$:

3. $|p| = 0$ iff $p = e$,
4. $|p \cdot q| = |p| + |q|$,
5. if $p \neq e$, then there is a $p' \in M$ such that $p \to p'$ with $|p'| = |p| - 1$,

6. if $p \to q$, then $|q| < |p|$,
7. if $p \cdot q \to r$, then either $q \to r$,
   or there is a $p' \in M$ such that $p \to p'$ and $r = p' \cdot q$,
8. if $p \cdot r = q \cdot r$, then $p = q$ (cancellation) .

Then $(M, \cdot, |\_|, \to, e)$ is a *unique decomposition monoid.*

The following observations can be made about these properties. First of all, note that by Properties 3 and 6 there can never exist a $q \in M$ for which $e \to q$, since that would make the size of q be smaller than zero. Also note that it is possible by Property 6 for an element $p \in M$ to have a transition $p \to p'$ (for some $p' \in M$) such that $|p| - |p'| > 1$, i.e., that the size of an element decreases by more than one – as long as there is (by Property 5) at least one transition which decreases the size of p by exactly one. This is used to make possible impure termination in $\mathsf{BPA}_{\mathbf{1}}$, as we shall see in Section 5.4.

5.2 Decomposition proof for monoids

We can now prove that a unique prime decomposition exists for each element of a unique decomposition monoid; but first we define primeness and decomposition on elements of a monoid by analogy of primeness and sequential decompositions of processes (Definitions 4.1 and 4.2 respectively).

Definition 5.3. An element $p \in M$ of a monoid $(M, \cdot, e)$, with $p \neq e$, is called *prime* if, for all elements $q, r \in M$, we have that $p = q \cdot r$ implies either $q = e$ or $r = e$.

Definition 5.4. A *decomposition* of an element $p \in M$ of a monoid $(M, \cdot, e)$ consists of a sequence of elements $\langle p_1, p_2, \ldots, p_n \rangle$ such that $p = p_1 \cdot p_2 \cdot \ldots \cdot p_n$; call $p_1 \cdot p_2 \cdot \ldots \cdot p_n$ the element associated with decomposition $\langle p_1, p_2, \ldots, p_n \rangle$. We define e to be the element associated with the empty sequence $\langle\rangle$, because e is a unit element under composition (by Property 2).

In order to prove that a unique prime decomposition exists, we first need one additional lemma.

Lemma 5.5. *Consider a unique decomposition monoid $(M, \cdot, |\_|, \to, e)$, and elements $p, q, r \in M$; and suppose $p \cdot q \to r$ with $|r| = |p \cdot q| - 1$, and $p \neq e$ and $q \neq e$. Then there exists an element $p' \in M$ such that $p \to p'$ with $r = p' \cdot q$.*

Proof. Since $p \cdot q \to r$, we have by Property 7 that either $q \to r$ or there is a $p' \in M$ such that $p \to p'$ and $r = p' \cdot q$.

Now suppose $q \to r$; then by Property 6 we have $|r| < |q|$. Now we have $|r| = |p \cdot q| - 1 = |p| + |q| - 1$ by Property 4, so $|p| + |q| - 1 < |q|$, so $|p| - 1 < 0$, hence $|p| < 1$, i.e., $|p| = 0$. But since $p \neq e$, by Property 3 we have $|p| \neq 0$. This is a contradiction, so we do *not* have $q \to r$. Hence, by Property 7 it must be that there is a $p' \in M$ such that $p \to p'$ with $r = p' \cdot q$. □

We now prove that every unique decomposition monoid has a unique prime decomposition result. This proof follows roughly the same structure as that of Theorem 4.3, the unique prime decomposition theorem for deadlock-free processes.

Theorem 5.6. *Consider a unique decomposition monoid $(M, \cdot, |\_|, \rightarrow, e)$. Every element $p \in M$ has a decomposition into a sequence of elements $\langle p_1, p_2, \ldots, p_n \rangle$, with $n \in \mathbb{N}$ and $p_i \in M$ for all $1 \leq i \leq n$, such that each p_i is prime, and $p = p_1 \cdot p_2 \cdot \ldots \cdot p_n$. For every p, only one such prime decomposition exists.*

Proof. We prove both existence and uniqueness of a prime decomposition of p by induction on $|p|$.

1. If $|p| = 0$, then by Property 3 we have $p = e$, so we have $\langle\rangle$ as the (empty) prime decomposition of p by Definition 5.4. As for uniqueness, suppose we have another decomposition of p into $\langle p_1, p_2, \ldots, p_n \rangle$, with p_i prime for all $1 \leq i \leq n$. Since $|p| = 0$ we have by substitution that $|p_1 \cdot p_2 \cdot \ldots \cdot p_n| = 0$ as well, so by Property 4 we have $|p_1|+|p_2|+\ldots+|p_n| = 0$; then it must be that $|p_i| = 0$ for all $1 \leq i \leq n$. This means that again by Property 3 we have $p_i = e$ for all $1 \leq i \leq n$, and e is not prime by Definition 5.3, so this would mean that this decomposition is not prime. Hence, no decomposition other than the empty one exists, i.e., that decomposition is unique.

2. Suppose that $|p| > 0$. By induction we have a unique prime decomposition for all p' with $|p'| < |p|$.

   If p is prime, then by Definition 5.4 we have $\langle p \rangle$ as prime decomposition of p. As for uniqueness, suppose we have another decomposition of p into $\langle p_1, p_2, \ldots, p_n \rangle$, with p_i prime for all $1 \leq i \leq n$. We now perform case distinction on n.

   (a) If $n = 0$ then we have the empty decomposition $\langle\rangle$, which would mean by Property 3 that $p = e$. But p is prime and e is not prime by Definition 5.3, so this cannot be the case.

   (b) If $n = 1$ then we have $\langle p_1 \rangle$ as prime decomposition with $p_1 = p$; this decomposition of p into $\langle p \rangle$ was already established above.

   (c) If $n > 1$ then there exist $q, r \in M$ such that $p = q \cdot r$, e.g. take $q = p_1$ and $r = p_2 \cdot \ldots \cdot p_n$, so then p would not be prime, so this cannot be the case.

   Hence, the only decomposition of p is $\langle p \rangle$ itself, so that decomposition is unique.

   If p is not prime, then by definition of primeness (Definition 5.3) there exist $q, r \in M$ such that $p = q \cdot r$, with $q \neq e$ and $r \neq e$, so by Property 3 we have $|q| > 0$ and $|r| > 0$. Since $|p| = |q \cdot r| = |q|+|r|$ by Property 4, we have $|q| < |p|$ and $|r| < |p|$, so by induction we have a prime decomposition of q into $\langle q_1, q_2, \ldots, q_n \rangle$, with q_i prime for all $1 \leq i \leq n$, and a prime decomposition of r into $\langle r_1, r_2, \ldots, r_m \rangle$, with r_i prime for all $1 \leq i \leq m$. Because $p = q \cdot r$ we have by substitution that $\langle q_1, q_2, \ldots, q_n, r_1, r_2, \ldots, r_m \rangle$ is a prime decomposition of p.

Now for uniqueness, suppose we have two prime decompositions of p, namely $\langle x_1, x_2, \ldots, x_k \rangle$ and $\langle y_1, y_2, \ldots, y_l \rangle$, with x_i prime for all $1 \leq i \leq k$ and y_i prime for all $1 \leq i \leq l$; so $p = x_1 \cdot x_2 \cdot \ldots \cdot x_k = y_1 \cdot y_2 \cdot \ldots \cdot y_l$ by Definition 5.4. We need to prove that these two decompositions are the same, i.e. that $k = l$ and $x_i = y_i$ for all $1 \leq i \leq k$. Since $|p| > 0$ we have by Property 3 that $p \neq e$, and thus by Property 5 that there exists a $p' \in M$ such that $p \to p'$ with $|p'| = |p| - 1$.

Now we know that $p = x_1 \cdot x_2 \cdot \ldots \cdot x_k \to p'$, and $x_1 \neq e$ (because x_1 is prime) and $x_2 \cdot \ldots \cdot x_k \neq e$ (because first of all $k > 1$, since $k = 0$ would yield an empty decomposition and $k = 1$ would mean that p is prime, both of which are not the case; and all of $x_2, \ldots, x_k$ are prime and thus not equal to e, so by repeated application of Property 4 and by Property 3 we have $x_2 \cdot \ldots \cdot x_k \neq e$), hence we can apply Lemma 5.5 on x_1 and $x_2 \cdot \ldots \cdot x_k$ to get that $p' = x'_1 \cdot x_2 \cdot \ldots \cdot x_k$, with $x_1 \to x'_1$ for some $x'_1 \in M$. Now x'_1 does not have to be prime, so we can decompose it into $\langle x'_{11}, x'_{12}, \ldots, x'_{1s} \rangle$, with x_{1i} prime for all $1 \leq i \leq s$; then we have $\langle x'_{11}, x'_{12}, \ldots, x'_{1s}, x_2, \ldots, x_k \rangle$ as decomposition of p', and since $|p'| < |p|$ by Property 6, this decomposition is unique by induction.

Now we also have that $p = y_1 \cdot y_2 \cdot \ldots \cdot y_l \to p'$, so by the same reasoning as applied above, we also have $p' = y'_1 \cdot y_2 \cdot \ldots \cdot y_l$, with $y_1 \to y'_1$ for some $y'_1 \in M$, and we have $\langle y'_{11}, y'_{12}, \ldots, y'_{1t}, y_2, \ldots, y_l \rangle$ as decomposition for p', with y_{1i} prime for all $1 \leq i \leq t$, which is once again unique by induction.

We now have two decompositions of p', but since decomposition for p' is unique, it must be that $s + k = t + l$, and that the ith element of the decomposition of x' is equal to the ith element of the decomposition of y', for $1 \leq i \leq s + k$; so in particular we have that $x_k = y_l$. With this fact established, we can use Property 8 to derive that $x_1 \cdot x_2 \cdot \ldots \cdot x_{k-1} = y_1 \cdot y_2 \cdot \ldots \cdot y_{l-1}$. The size of both of these elements is smaller than $|p|$, since $|x_k| > 0$ and $|y_l| > 0$ (because both are prime). This means that once again by induction, decomposition is unique for $x_1 \cdot x_2 \cdot \ldots \cdot x_{k-1}$, so we have that $k = l$ and $x_i = y_i$ for $1 \leq i < k$. We had already established that $x_k = y_l$, so in conclusion, decomposition for p is also unique if p is not prime.

For both $|p| = 0$ and $|p| > 0$ we have found a prime decomposition for p and established its uniqueness. □

Note that to merely prove that a prime decomposition exists, instead of also proving its uniqueness, we only need the monoid properties of associativity and identity (Properties 1 and 2) from Definition 5.1, and Properties 3 and 4 from Definition 5.2.

5.3 Cancellation for monoids

We have now proved that a monoid has a unique prime decomposition result if a number of properties hold for that monoid. However, we have not proved that all these properties are actually necessary to have.

In the $\mathsf{BPA}_{\mathbf{1}}/\mathsf{BPA}_{\mathbf{0,1}}$ case, it was possible to prove cancellation directly from the SOS rules and bisimulation relation imposed on the algebra. We would have liked to do the same in the case of monoids, i.e. to derive Property 8 in Definition 5.2 from Properties 1–7 in Definitions 5.1 and 5.2. However, it is not possible to do so, since cancellation is not guaranteed to hold using only these properties. We prove this by providing an example for which every property *except* for cancellation holds.

Theorem 5.7. *Property 8 is not derivable from Properties 1–7, i.e. there exist monoids for which Properties 1–7 hold, but Property 8 does not hold.*

Proof. Consider a monoid $(M, \cdot, \epsilon)$ which is defined as follows:

- The set M consists of strings of arbitrary finite length from the alphabet $\{\mathsf{a}, \mathsf{b}\}$; we can recursively define M using the grammar $S ::= \mathsf{a} \mid \mathsf{b} \mid SS$. We denote the empty string with the symbol ϵ, and take this element as the unit element. Furthermore, we use the notation p^n, with $p \in M$ and $n \in \mathbb{N}$, for the string p concatenated n times; so $p^0 = \epsilon$, and $p^{n+1} = pp^n$.
- Take a size function $|\_| : M \to \mathbb{N}$ such that for all $p \in M$, $|p|$ is the length of the string p; so $|\epsilon| = 0$, and $|cp| = 1 + |p|$ for $c \in \{\mathsf{a}, \mathsf{b}\}$.
- We define the composition function $\_\cdot\_ : M \times M \to M$ as follows, for all $p, q \in M$:

$$p \cdot q = \begin{cases} p & \text{if } q = \epsilon \\ pq & \text{if } q = \mathsf{a}q' \text{ for some } q' \in M \\ \mathsf{b}^n q & \text{if } q = \mathsf{b}q' \text{ for some } q' \in M, \text{ and } |p| = n\ . \end{cases}$$

  Hence, the character b at the beginning of the right-hand string causes all characters in the left-hand string to be replaced by the character b.
- Finally a transition relation $\to \subseteq M \times M$ is defined as the smallest set such that $p \to p'$ if $p = cp'$, with $c \in \{\mathsf{a}, \mathsf{b}\}$ and $p' \in M$; i.e. each step removes one character from the left-hand side of the string, until the empty string is reached.

We now prove for $(M, \cdot, \epsilon)$ that Properties 1 and 2 hold (thereby proving that it is a monoid) and Properties 3–7 hold, but Property 8 does *not* hold, so it is a unique decomposition monoid except for cancellation.

First of all, it is easy to see that cancellation does not hold: we have both $\mathsf{a} \cdot \mathsf{b} = \mathsf{bb}$ and $\mathsf{b} \cdot \mathsf{b} = \mathsf{bb}$, so $\mathsf{a} \cdot \mathsf{b} = \mathsf{b} \cdot \mathsf{b}$, but not $\mathsf{a} = \mathsf{b}$, since a and b are distinct elements. We now prove that all the other properties *do* hold, for all $p, q, r \in M$.

1. We prove by case distinction that $p \cdot (q \cdot r) = (p \cdot q) \cdot r$.
   (a) Suppose $r = \epsilon$. Then $p \cdot (q \cdot r) = p \cdot q = (p \cdot q) \cdot r$, twice by definition of $\_\cdot\_$.
   (b) Suppose $r = \mathsf{b}r'$ for some $r' \in M$. Then $p \cdot (q \cdot r) = p \cdot \mathsf{b}^m r = \mathsf{b}^{n+m} r$, twice by definition of $\_\cdot\_$, for $|p| = n$ and $|q| = m$; and $(p \cdot q) \cdot r = \mathsf{b}^k r$ by definition of $\_\cdot\_$, for $|p \cdot q| = k$, and by Property 4 (which we prove below) $k = |p| + |q| = n + m$, so also $(p \cdot q) \cdot r = \mathsf{b}^{n+m} r$.

(c) Suppose $r = \mathsf{a}r'$ for some $r' \in M$, then we distinguish on q:

i. Suppose $q = \epsilon$. Then $p \cdot (q \cdot r) = p \cdot r = (p \cdot q) \cdot r$, twice by definition of $\_ \cdot \_$.

ii. Suppose $q = \mathsf{b}q'$ for some $q' \in M$. Then $p \cdot (q \cdot r) = p \cdot qr = \mathsf{b}^n qr$, twice by definition of $\_ \cdot \_$, for $|p| = n$; and $(p \cdot q) \cdot r = \mathsf{b}^n q \cdot r = \mathsf{b}^n qr$, also twice by definition of $\_ \cdot \_$.

iii. Suppose $q = \mathsf{a}q'$ for some $q' \in M$. Then $p \cdot (q \cdot r) = p \cdot qr = pqr = p \cdot qr = p \cdot (q \cdot r)$, four times by definition of $\_ \cdot \_$.

All cases have been covered, so associativity holds.

2. By definition of $\_ \cdot \_$ we have $p \cdot \epsilon = p$; it also follows from the definition that $\epsilon \cdot q = q$ for all three cases of q in the definition.

3. By definition of $|\_|$ we have $|\epsilon| = 0$, and the only string of length 0 is the empty string.

4. By definition of $\_ \cdot \_$ we have for all three cases of $p \cdot q$ that the resulting string has the same length as that of p and q combined. Since $|p|$ is defined as the length of p for all p, we have that $|p \cdot q| = |p| + |q|$.

5. If $p \neq \epsilon$, then by Property 3 we have $|p| > 0$, so by definition of $|\_|$ we have that p is a string of at least length one, so $p = cp'$ for some $c \in \{\mathsf{a}, \mathsf{b}\}$ and $p' \in M$, so by definition of $\rightarrow$ we have $p \rightarrow p'$, and by definition of $|\_|$, $|p'| = |p| - 1$.

6. By definition of $\rightarrow$ and $|\_|$, every step $p \rightarrow p'$ for some $p' \in M$ has $|p'| = |p| - 1$, so $|p'| < |p|$.

7. Suppose that $p \cdot q \rightarrow r$. We now prove by case distinction on p.

(a) Suppose $p = \epsilon$. Then by Property 2, $p \cdot q = q$, so $q \rightarrow r$.

(b) Suppose $p \neq \epsilon$, then we distinguish on q.

i. Suppose $q = \epsilon$. Then $p \cdot q = p$ by definition of $\_ \cdot \_$, so $p \rightarrow r$, and because $q = \epsilon$, also $r = r \cdot q$; then we can take $p' = r$ for $r = p' \cdot q$.

ii. Suppose $q = \mathsf{b}q'$ for some $q' \in M$. Then $p \cdot q = \mathsf{b}^n q$ by definition of $\_ \cdot \_$, for $|p| = n$; and since $p \neq \epsilon$, by Property 3 we have $|p| > 0$, so $n > 0$. Hence by definition of $\rightarrow$ we have $\mathsf{b}^n q \rightarrow \mathsf{b}^{n-1} q$, so $r = \mathsf{b}^{n-1} q$.
Also since $n > 0$ we have $p = cp'$ for some $c \in \{\mathsf{a}, \mathsf{b}\}$ and $p' \in M$, hence by definition of $\rightarrow$ we have $p \rightarrow p'$ with $|p'| = n - 1$ by definition of $|\_|$. Then $p' \cdot q = \mathsf{b}^{n-1} q$ by definition of $\_ \cdot \_$, so $p' \cdot q = r$.

iii. Suppose $q = \mathsf{a}q'$ for some $q' \in M$. Then $p \cdot q = pq$ by definition of $\_ \cdot \_$; and since $p \neq \epsilon$, by Property 3 we have $|p| > 0$, so $p = cp'$ for some $c \in \{\mathsf{a}, \mathsf{b}\}$ and $p' \in M$. Hence by definition of $\rightarrow$ we have $pq \rightarrow p'q$, so $r = p'q$.
Also by definition of $\rightarrow$ we have $p \rightarrow p'$; then $p' \cdot q = p'q$ by definition of $\_ \cdot \_$, so $p' \cdot q = r$.

In each case we have either $q \to r$ or there is a $p' \in M$ such that $p \to p'$ and $r = p' \cdot q$, so the property holds.

All properties except for cancellation hold, so the given monoid M is a unique decomposition monoid except for cancellation. □

Now that we have shown that cancellation is not derivable from the other properties in a general setting, the only question remaining is whether cancellation is actually required in order to prove unique decomposition. We can easily show that this is the case: if we have unique decomposition without cancellation, then there exist elements $p, q, r \in M$ such that $p \cdot r = q \cdot r$, but $p \neq q$. Suppose the decompositions of p, q and r are

$$\langle p_1, p_2, \ldots, p_n \rangle \text{ with } p_i \in M \text{ for } 1 \leq i \leq n$$
$$\langle q_1, q_2, \ldots, q_m \rangle \text{ with } q_i \in M \text{ for } 1 \leq i \leq m$$
$$\langle r_1, r_2, \ldots, r_k \rangle \text{ with } r_i \in M \text{ for } 1 \leq i \leq k$$

respectively. Then we have as the unique decomposition of $p \cdot r = q \cdot r$ that $\langle p_1, p_2, \ldots, p_n, r_1, r_2, \ldots, r_k \rangle = \langle q_1, q_2, \ldots, q_m, r_1, r_2, \ldots, r_k \rangle$, but this would imply that the decompositions of p and q are the same, which in turn would imply that $p = q$; and we assumed that this is not the case.

5.4 $\mathsf{BPA}_{\mathbf{1}}$ as unique decomposition monoid

Now that we have established that unique decomposition monoids are actually uniquely decomposable, we can apply this result to processes in $\mathsf{BPA}_{\mathbf{1}}$ by showing that $\mathsf{BPA}_{\mathbf{1}}$ is in fact a unique decomposition monoid; we do this by proving that Properties 1 and 2 from Definition 5.1 and Properties 3–8 from Definition 5.2 hold for $\mathsf{BPA}_{\mathbf{1}}$.

For reasoning about equality on processes we use the bisimulation semantics as established before, so we divide $\mathsf{BPA}_{\mathbf{1}}$ into *equivalence classes* modulo bisimulation. We use the notation $\mathcal{P}$ to indicate the set of processes in $\mathsf{BPA}_{\mathbf{1}}$ divided using bisimulation; for every process p from $\mathsf{BPA}_{\mathbf{1}}$ we write $[p] \in \mathcal{P}$ for the equivalence class containing the process p. This is allowed because bisimulation is a *congruence relation* on closed terms with respect to each of the operators in $\mathsf{BPA}_{\mathbf{1}}$. This fact follows from the shape of the SOS rules, which adheres to the tyft/tyxt format as presented by Groote and Vaandrager [13]; for rules in this format, the bisimulation relation is automatically a congruence.

This results in the following definition.

Definition 5.8. We define a monoid $(\mathcal{P}, \cdot, [\mathbf{1}])$ with the following properties:

- The set $\mathcal{P}$ consists of the equivalence classes of processes in $\mathsf{BPA}_{\mathbf{1}}$, with $[\mathbf{1}]$ as unit element.
- For the composition function $\_ \cdot \_ : \mathcal{P} \times \mathcal{P} \to \mathcal{P}$ we use the sequential composition function defined in $\mathsf{BPA}_{\mathbf{1}}$, so for all processes p and q we have $[p] \cdot [q] = [p \cdot q]$.

- We define a transition relation $\rightarrow \subseteq \mathcal{P} \times \mathcal{P}$ such that for all processes p and p' we have $[p] \rightarrow [p']$ iff there exist processes q and q' and an $a \in \mathcal{A}$ such that $p \leftrightarrows q$, $p' \leftrightarrows q'$ and $q \xrightarrow{a} q'$.
- Finally, take a size function $|\_| : \mathcal{P} \rightarrow \mathbb{N}$ such that for every process p we have $|[p]| = |p|$.

We now prove that this definition actually results in a unique decomposition monoid.

Theorem 5.9. *Take the monoid $(\mathcal{P}, \cdot, [\mathbf{1}])$ with properties as defined in Definition 5.8 above. Then $(\mathcal{P}, \cdot, |\_|, \rightarrow, [\mathbf{1}])$ is a unique decomposition monoid.*

Proof. We need to prove that Properties 1 and 2 from Definition 5.1 and Properties 3–8 from Definition 5.2 hold for all $P, Q, R \in \mathcal{P}$. Since for all $P \in \mathcal{P}$ and all $p \in P$ we have $[p] = P$, we take processes p, q, r and prove the properties for all $[p]$, $[q]$ and $[r]$.

1. By Lemma 2.5 it holds that $p \cdot (q \cdot r) \leftrightarrows (p \cdot q) \cdot r$; hence they are in the same equivalence class, so $[p \cdot (q \cdot r)] = [(p \cdot q) \cdot r]$; then by repeated application of the definition of $\_ \cdot \_$ we have $[p] \cdot ([q] \cdot [r]) = ([p] \cdot [q]) \cdot [r]$.
2. By Lemma 2.5 it holds that $p \cdot \mathbf{1} \leftrightarrows p$ and $\mathbf{1} \cdot p \leftrightarrows p$; hence $p \cdot \mathbf{1}$, p and $\mathbf{1} \cdot p$ are all in the same equivalence class, so $[p \cdot \mathbf{1}] = [p] = [\mathbf{1} \cdot p]$, so by definition of $\_ \cdot \_$ we have $[p] \cdot [\mathbf{1}] = [\mathbf{1}] \cdot [p] = [p]$.
3. By definition of $|\_|$ we have $|[p]| = 0$ iff $|p| = 0$, which is the case if $\max\{n \in \mathbb{N} \mid \exists p' : p \longrightarrow^n p'\} = 0$, i.e. if there is no process p' and action $a \in \mathcal{A}$ such that $p \xrightarrow{a} p'$, if p cannot perform an action. According to the SOS rules (Table 2.1) this is the case iff $p \leftrightarrows \mathbf{1}$, hence p and $\mathbf{1}$ are in the same equivalence class, so $[p] = [\mathbf{1}]$.
4. We have $|[p] \cdot [q]| = |[p \cdot q]| = |p \cdot q| = |p| + |q| = |[p]| + |[q]|$ by congruence, definition of $|\_|$, Lemma 2.7 and again definition of $|\_|$, respectively.
5. Suppose $[p] \neq [\mathbf{1}]$. Then by Property 3 we have $|[p]| \neq 0$, hence $|p| \neq 0$, so by definition of size for $\mathsf{BPA}_{\mathbf{1}}$ – since the size is defined as the maximum number of actions that can be performed – it must be the case that there is an action $a \in \mathcal{A}$ and process p' for which $p \xrightarrow{a} p'$, with the number of actions that can be performed in p' is one less than that of p. Hence $p \xrightarrow{a} p'$ with $|p'| = |p| - 1$; thus by definition of $|\_|$ also $|[p']| = |[p]| - 1$.
6. If $[p] \rightarrow [q]$, then by definition of $\rightarrow$ there is an $a \in \mathcal{A}$ such that $p \xrightarrow{a} q$. Now by Lemma 2.9 we have $|q| < |p|$, so also $|[q]| < |[p]|$.
7. Suppose $[p] \cdot [q] \rightarrow [r]$, then by definition of $\rightarrow$ and because $[p] \cdot [q] = [p \cdot q]$, there is an $a \in \mathcal{A}$ such that $p \cdot q \xrightarrow{a} r$. By the SOS rules in Table 2.1 this is possible either by Rule 7 or Rule 8, so we distinguish on those.
   (a) By Rule 7 we have that $p \cdot q \xrightarrow{a} r$ if $p \xrightarrow{a} p'$; then $r \leftrightarrows p' \cdot q$, so by congruence $[r] = [p'] \cdot [q]$.

(b) By Rule 8 we have that $p \cdot q \xrightarrow{a} r$ if $p{\downarrow}$ and $q \xrightarrow{a} q'$; then $r \leftrightarroweq q'$. This means that $[q] \to [q']$ by definition of $\to$, and since $r \leftrightarroweq q'$ we have also that $[r] = [q']$, hence $[q] \to [r]$.

In both cases we have either $[q] \to [r]$ or there is a $[p']$ such that $[p] \to [p']$ and $[r] = [p'] \cdot [q]$.

8. Suppose $[p] \cdot [r] = [q] \cdot [r]$. Then by congruence we have $[p \cdot r] = [q \cdot r]$, so $p \cdot r \leftrightarroweq q \cdot r$. Then by the $\mathsf{BPA}_{\mathbf{1}}$ cancellation theorem (Theorem 3.2) we have $p \leftrightarroweq q$, so $[p] = [q]$.

All required properties hold, so $(\mathcal{P}, \cdot, |\_|, \to, [\mathbf{1}])$ is a unique decomposition monoid. □

Note that Property 3 would not hold in case we added deadlock, since $|\mathbf{0}| = 0$ as well; hence this definition only works for $\mathsf{BPA}_{\mathbf{1}}$ and not for $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$. However, it might also be possible to establish another definition of unique decomposition monoids in which $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ can be defined as well.

Chapter 6

Conclusion

In this thesis, we have established a number of results. We have proven that all processes in $\mathsf{BPA}_{\mathbf{1}}$ have a unique sequential prime decomposition, and we have shown that such a decomposition does not exist in general for $\mathsf{BPA}_{\mathbf{0,1}}$. To alleviate this, we have established two decomposition results for processes containing deadlock, both as close as possible to a prime decomposition. Taken together, these three forms of decomposition provide a unique decomposition for all processes in $\mathsf{BPA}_{\mathbf{0,1}}$ as well.

Aside from establishing a unique decomposition for processes in $\mathsf{BPA}_{\mathbf{1}}/\mathsf{BPA}_{\mathbf{0,1}}$, we have also established a sufficient criterion to be able to provide a unique prime decomposition result for monoids, and have proven that a reduction of $\mathsf{BPA}_{\mathbf{1}}$ to monoids satisfies this criterion. Furthermore we have shown that a cancellation property is necessary to be able to define unique decomposition on monoids in this fashion.

There is a lot of future work that could still be done in this area, mostly in terms of trying to obtain a unique decomposition result for different process algebras. Mostly lacking right now is an attempt at decomposition for $\mathsf{BPA}_{\mathbf{1}}/\mathsf{BPA}_{\mathbf{0,1}}$ where infinite processes are allowed. A possible starting point for this is given in the Appendix. Another possibility would be to first add the *Kleene star* operator to the algebra, which provides a limited form of infinity which might be easier to handle.

Another candidate for decomposition would be $\mathsf{BPP}_{\mathbf{0,1}}$, the Basic Parallel Processes with deadlock and (impure) termination – which could derive its result from the unique decomposition of $\mathsf{BPP}_{\mathbf{1}}$, which was proven by Luttik and Van Oostrom [17].

Furthermore, we could try to refine the decomposition result for $\mathsf{BPA}_{\mathbf{0,1}}$ in such a way that it becomes possible – under some circumstances – to also allow deadlock in other components than the last one. This should be possible in theory, since there are processes containing deadlock which are prime, and thus do not give rise to infinite decomposition, as shown by Lemma 4.15.

As already stated in Section 5.4, we conjecture that it is also possible to define a monoid in such a way that it captures the essence of all decompositions of $\mathsf{BPA}_{\mathbf{0,1}}$. Hence, constructing such a monoid would also belong to future work.

Finally, we have shown that the properties for a unique decomposition monoid are sufficient for proving unique prime decomposition, but have not proven that they are all necessary, except for cancellation. Proving the necessity of the other properties would show that the collection of properties as a whole is minimal.

References

[1] L. Aceto, W.J. Fokkink, and C. Verhoef. Structural operational semantics. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*, chapter 3, pages 197–292. Elsevier, 2001.

[2] J.C.M. Baeten. Embedding untimed into timed process algebra: the case for explicit termination. *Mathematical Structures in Computer Science*, 13(04):589–618, 2003.

[3] J.C.M. Baeten. Calculating with Automata. In P. Degano, R. De Nicola, and J. Meseguer, editors, *Concurrency, Graphs and Models*, number 5065 in LNCS, pages 747–756. Springer-Verlag, 2008.

[4] J.C.M. Baeten, T. Basten, and M.A. Reniers. *Process Algebra: Equational Theories of Communicating Processes*. Number 50 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 2009. To appear.

[5] J.C.M Baeten, P.J.L Cuijpers, B. Luttik, and P.J.A. van Tilburg. Models of computation: automata and processes. In *Proceedings of FSEN 2009*, LNCS. Springer-Verlag, 2009. To appear.

[6] J.C.M. Baeten, P.J.L. Cuijpers, and P.J.A. van Tilburg. A Context-Free Process as a Pushdown Automaton. In F. van Breugel and M. Chechik, editors, *Proceedings of CONCUR 2008*, number 5201 in LNCS, pages 98–113. Springer-Verlag, 2008.

[7] J.C.M. Baeten, P.J.L. Cuijpers, and P.J.A. van Tilburg. A Basic Parallel Process as a Parallel Pushdown Automaton. In T. Hildebrandt and D. Gorla, editors, *Proceedings of EXPRESS 2008*, number 242 in ENTCS, pages 35–48. Elsevier, 2009.

[8] J.A. Bergstra and J.W. Klop. Process Algebra for Synchronous Communication. *Information and Control*, 1/3(60):109–137, 1984.

[9] O. Burkart, D. Caucal, F. Moller, and B. Steffen. Verification on Infinite Structures. In J. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, pages 545–623. Elsevier, 2001.

[10] I. Castellani and M. Hennessy. Distributed Bisimulations. *Journal of the ACM*, 36(4):887–911, 1989.

[11] S. Christensen. *Decidability and Decomposition in Process Algebras.* PhD thesis, University of Edinburgh, 1993. ECS-LFCS-93-278.

[12] S. Christensen, Y. Hirshfeld, and F. Moller. Decomposability, Decidability and Axiomatisability for Bisimulation equivalence on basic parallel processes. In *Proceedings of LICS '93*, pages 386–396. IEEE Computer Society Press, 1993.

[13] J.F. Groote and F. Vaandrager. Structured Operational Semantics and Bisimulation as a Congruence. In G. Ausiello, M. Dezani-Ciancaglini, and S. Ronchi Della Rocca, editors, *Proceedings of ICALP '89*, number 372 in LNCS, pages 423–438. Springer-Verlag, 1989.

[14] J.M. Howie. *Fundamentals of Semigroup Theory.* Number 12 in London Mathematical Society Monographs New Series. Oxford University Press, 1995.

[15] C.P.J. Koymans and J.L.M Vrancken. Extending process algebra with the empty process ε. Logic Group Preprint Series 1, Utrecht University, Department of Philosophy, 1985.

[16] B. Luttik. What is algebraic in process theory? *Bulletin of the EATCS*, 88:65–83, 2006.

[17] B. Luttik and V. van Oostrom. Decomposition Orders – another generalisation of the fundamental theorem of arithmetic. *Theoretical Computer Science*, 335(2-3):147–186, 2005.

[18] R. Milner and F. Moller. Unique Decomposition of Processes. *Theoretical Computer Science*, 107(2):357–363, 1993.

[19] F. Moller. *Axioms for Concurrency.* PhD thesis, University of Edinburgh, 1989. ECS-LFCS-89-84.

[20] F. Moller. The Importance of the Left Merge Operator in Process Algebras. In M.S. Paterson, editor, *Proceedings of ICALP '90*, number 443 in LNCS, pages 752–764. Springer-Verlag, 1990.

[21] F. Moller. The Nonexistence of Finite Axiomatisations for CCS Congruences. In *Proceedings of LICS '90*, pages 142–153. IEEE Computer Society Press, 1990.

[22] J.L.M. Vrancken. The algebra of communicating processes with empty process. *Theoretical Computer Science*, 177:287–328, 1997.

Appendix A

Some preliminary results regarding recursion

Originally it was our aim for this thesis to include the notion of infinite processes in $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ by allowing *recursive specifications* to be defined. However, infinite processes prove to be a problem with regard to unique prime decomposition. To show this, we first modify the basic syntax of processes to allow for recursive specifications.

Definition A.1. A *recursive specification* over $\mathcal{V}$, where $\mathcal{V}$ is a (finite) set of variables, is defined as a set of equations of the form $X = t_X$, one for each variable $X \in \mathcal{V}$. Here t_X is a term over the signature of $\mathsf{BPA}_{\mathbf{1}}$ or $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$, which may contain elements of $\mathcal{V}$.

If we now allow these recursive specification variables to occur in the syntax of our algebras, we get the following modified grammar for $\mathsf{BPA}_{\mathbf{1}}$:

$$P ::= \mathbf{1} \mid a.P \mid X \mid P + P \mid P \cdot P \ ,$$

and for $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$:

$$P ::= \mathbf{0} \mid \mathbf{1} \mid a.P \mid X \mid P + P \mid P \cdot P \ .$$

This addition also slightly changes the semantics for $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$.

Definition A.2. The operational semantics for $\mathsf{BPA}_{\mathbf{1}}$ and $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ are given by the SOS rules in Table 2.1 with the addition of the two rules for recursion given in Table A.1. Here $\mathcal{A}$ is the set of possible actions, and $\mathcal{V}$ is the set of recursive specification variables.

It is now possible to specify a plethora of processes which were previously inexpressible. Examples are $X = a.X$, a process which can only perform an infinite number of a-actions and never terminate, or $X = (a.X \cdot b.\mathbf{1}) + \mathbf{1}$, a process which can perform an unbounded number of a-actions, then the same number of b-actions, and then terminate.

$$10\ \frac{t_X \xrightarrow{a} x \quad X = t_X}{X \xrightarrow{a} x} \qquad 11\ \frac{t_X\downarrow \quad X = t_X}{X\downarrow}$$

Table A.1: Additional SOS rules for recursion ($a \in \mathcal{A}, X \in \mathcal{V}$)

Now a number of these processes do not have a unique prime decomposition. An example of this is the process $X = a.X$, which can be decomposed into $\langle X\rangle, \langle X, X\rangle, \langle X, X, X\rangle$, etcetera, but none of these decompositions are prime, since X can always be decomposed further.

At the time we did not see a way of keeping recursive specifications in the theory without significantly hampering the overall progress of the project, so we decided not to include recursive specifications in our main results. However, at this moment we already had proof that cancellation *does* hold for infinite processes, as long as the component to be split off is finite. Since this result might aid a future endeavour into decomposition of infinite processes (or more generally, processes with recursion), we present it here. The cancellation proof itself is mostly the same as the one present in Theorem 3.2; most differences arise in the definitions and lemmas on which the theorem is based.

First of all we make a significant change to the definition of the size of a process, and redefine it from Definition 2.6 to the following.

Definition A.3. We define the *size* of a process p, written $|p|$, to be the length of the longest transition sequence (t.s.) starting at p that ends with a termination, unless there are transition sequences starting at p that do not end in termination, in which case $|p|$ is defined to be ∞:

$$|p| = \begin{cases} \sup\{n \in \mathbb{N} \mid p \longrightarrow^n p' \wedge p'\downarrow\} & \text{if all t.s. starting at } p \text{ end in termination} \\ \infty & \text{otherwise .} \end{cases}$$

We say a process p is *finite-sized*, or write the predicate *finite-sized*(p), if $|p| = n$ for some $n \in \mathbb{N}$, i.e., it has a finite size, and we say it is *infinite-sized* otherwise.

Now there are three distinct ways in which a process can be infinite-sized:

1. If a process in $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$ has deadlock then there is a non-terminating transition sequence, so the process is infinite-sized by definition. Note that this differs greatly from the original definition of size as per Definition 2.6, where deadlock was treated the same as successful termination.

2. If a process contains a transition sequence of infinite length, as is the case in e.g. the process equation $X = a.X$, then this results in a non-terminating transition sequence, so it is also infinite-sized by definition.

3. If a process contains only terminating transition sequences with a finite length, and if that process has the characteristic that given a transition sequence with a fixed length (say n) it is always possible to find a transition sequence with a length greater than n, then $\sup\{n \in \mathbb{N} \mid p \longrightarrow^n p' \wedge p\downarrow\} = \infty$, so a process of such a kind is infinite-sized as well.

Now the size of finite-sized processes can easily be calculated, as per the following lemma (contrast this with Lemma 2.7).

Lemma A.4. *For finite-sized processes p and q and for $a \in \mathcal{A}$, the following equalities can be derived:*

$$\begin{aligned} |\mathbf{0}| &= \infty \\ |\mathbf{1}| &= 0 \,, \\ |a.p| &= 1 + |p| \,, \\ |p + q| &= \max(|p|, |q|) \,, \\ |p \cdot q| &= |p| + |q| \,. \end{aligned}$$

Proof. By definition of size (Definition A.3) and the SOS rules in Table 2.1. □

We now state some auxiliary lemmas on arbitrary processes in $\mathsf{BPA}_{\mathbf{1}}$ or $\mathsf{BPA}_{\mathbf{0,1}}$, which are needed to prove cancellation. These lemmas correspond to Lemmas 2.8, 2.9 and 3.1 respectively.

Lemma A.5. *For processes p and q, if $p \leftrightarrows q$, then $|p| = |q|$.*

Proof. First of all we prove that if $p \leftrightarrows q$ then either both p and q are finite-sized or both are infinite-sized. To this end, suppose without loss of generality that p is infinite-sized and q is finite-sized, with $|q| = n$. Then by definition of size (Definition A.3) there either is a non-terminating transition sequence starting at p (which cannot be simulated by q), or there are transition sequences of unbounded length starting at p; in which case there is also a transition sequence of length m, with $m > n$, from p, which cannot be simulated in q. This is a contradiction of $p \leftrightarrows q$.

So either of the two following cases holds:

1. Processes p and q are both finite-sized. Assume without loss of generality that (for $p \leftrightarrows q$) we have $|p| > |q|$. Then there is a transition sequence of length $|p|$ starting at p which is longer than any sequence in q, so q cannot simulate this sequence. But then $p \not\leftrightarrows q$, which is a contradiction. Hence it must be that $|p| = |q|$.
2. Processes p and q are both infinite-sized. Then by definition $|p| = |q| = \infty$.

In either case we have $|p| = |q|$. □

Lemma A.6. *If for some finite-sized process p we have $p \xrightarrow{a} p'$ for some process p' and $a \in \mathcal{A}$, then $|p'| < |p|$.*

Proof. Suppose $|p'| = n$, so the longest transition sequence starting at p' takes n steps. By $p \xrightarrow{a} p'$ it is possible to get from p to p' in one step, so there is a transition sequence of $n + 1$ steps starting at p. Thus the longest transition sequence starting at p must also be at least $n + 1$ steps long. Hence, $|p| \geq n + 1 > n = |p'|$, so $|p'| < |p|$. □

Lemma A.7. *If we have a finite-sized process r for which $r \xrightarrow{a} r'$ for some $a \in \mathcal{A}$ and process r', then there is no process p, finite-sized or infinite-sized, such that $p \cdot r \leftrightarrows r'$.*

Proof. We prove the lemma by contradiction. Suppose that there is a process p for which $p \cdot r \leftrightarrows r'$. We distinguish two cases:

1. Process p is finite-sized. Then by Lemma A.5, $|p \cdot r| = |r'|$ and by Lemma A.4, $|p \cdot r| = |p| + |r|$, so $|p| + |r| = |r'|$ and $|p| = |r'| - |r|$. But by Lemma A.6, $|r'| < |r|$, so $|r'| - |r| < 0$ and thus $|p| < 0$. But there cannot exist a process with a longest transition sequence smaller than zero, so this is a contradiction.
2. Process p is infinite-sized. Then $|p| = \infty$, so $|p \cdot r| = \infty$ as well. Since $p \cdot r \leftrightarrows r'$, then by Lemma A.5 we have $|r'| = \infty$ too. But since r was supposed to be finite-sized, by Lemma A.6 it holds that r' is finite-sized as well, so this is a contradiction.

In both cases we have a contradiction, so no such p exists. □

Note that if r is infinite-sized then such a p can exist, e.g. take $r = a.r + \mathbf{1}$ and $p = a.\mathbf{1} + \mathbf{1}$, then $r \xrightarrow{a} r$ and $p \cdot r \leftrightarrows r$.

Using these lemmas, we can prove the following cancellation theorem for processes in $\mathsf{BPA}_{\mathbf{1}}$ or $\mathsf{BPA}_{\mathbf{0},\mathbf{1}}$, as opposed to the one in Theorem 3.2.

Theorem A.8. *If for some (possibly infinite-sized) processes p and q there exists a finite-sized process r such that $p \cdot r \leftrightarrows q \cdot r$, then $p \leftrightarrows q$.*

Proof. Consider the following relation $\mathcal{R}$:

$$\mathcal{R} = \{(p, q) \mid (\exists r : \text{finite-sized}(r) \wedge p \cdot r \leftrightarrows q \cdot r)\} .$$

Obviously, if $\mathcal{R}$ is a bisimulation relation, then the theorem holds. Recall the definition of a bisimulation (Definition 2.4); we need to prove that each of the four properties of a bisimulation relation hold for $\mathcal{R}$, to establish that $\mathcal{R}$ is a bisimulation relation. For each property, assume $(p, q) \in \mathcal{R}$, so take a finite-sized r such that $p \cdot r \leftrightarrows q \cdot r$.

1. Suppose $p \xrightarrow{a} p'$ for some $a \in \mathcal{A}$; we need to show that there exists a q' such that $q \xrightarrow{a} q'$ and $(p', q') \in \mathcal{R}$.

   By SOS rule 7 (see Table 2.1) on $p \xrightarrow{a} p'$ we have $p \cdot r \xrightarrow{a} p' \cdot r$, so by bisimilarity of $p \cdot r$ and $q \cdot r$ there exists an $s \leftrightarrows p' \cdot r$ such that $q \cdot r \xrightarrow{a} s$. Fix a derivation of $q \cdot r \xrightarrow{a} s$; we distinguish cases according to which SOS rule has been applied last in this derivation. This can either be rule 7 or rule 8:

   (a) By SOS rule 7 there is a q' such that $q \xrightarrow{a} q'$; then $s = q' \cdot r$, so $p' \cdot r \leftrightarrows q' \cdot r$. By definition of $\mathcal{R}$, since finite-sized$(r) \wedge p' \cdot r \leftrightarrows q' \cdot r$, this means that $(p', q') \in \mathcal{R}$.

(b) By SOS rule 8 it holds that $q\downarrow$ and there is an r' such that $r \xrightarrow{a} r'$; then $s = r'$, so $p' \cdot r \leftrightarrows r'$. But by Lemma A.7 this is a contradiction, so this case cannot occur.

Only the first case can occur, so there is a q' such that $q \xrightarrow{a} q'$ with $(p', q') \in \mathcal{R}$; this satisfies the bisimulation relation requirement.

2. Suppose $q \xrightarrow{a} q'$ for some $a \in \mathcal{A}$; we need to show that there exists a p' such that $p \xrightarrow{a} p'$ and $(p', q') \in \mathcal{R}$.

   This is analogous to the case for $p \xrightarrow{a} p'$ above.

3. Suppose $p\downarrow$; we need to show that $q\downarrow$.

   To show this, we apply case distinction on r. Since r is finite-sized, it is deadlock-free, so r must be able to either perform an action and/or terminate. We distinguish between these two options:

   (a) Process r can perform an action: $r \xrightarrow{a} r'$ for some $a \in \mathcal{A}$. Then by SOS rule 8, $p \cdot r \xrightarrow{a} r'$, and by bisimilarity of $p \cdot r$ and $q \cdot r$ there exists an $s \leftrightarrows r'$ with $q \cdot r \xrightarrow{a} s$. Fix a derivation of $s \cdot r \xrightarrow{a} s$; we again distinguish cases according to which SOS rule has been applied last in this derivation, which is either SOS rule 7 or rule 8:

      i. By SOS rule 7 there is a q' such that $q \xrightarrow{a} q'$; then $s = q' \cdot r$, so $r' \leftrightarrows q' \cdot r$. But by Lemma A.7 this is a contradiction, so this case cannot occur.

      ii. By SOS rule 8 it holds that $q\downarrow$ and $s = r'$; so we have $q\downarrow$.

      Only the second case can occur, and in this case we have $q\downarrow$.

   (b) Process r can terminate: $r\downarrow$. Since $p\downarrow$ holds, by SOS rule 9 it holds that $p \cdot r\downarrow$, and by bisimilarity of $p \cdot r$ and $q \cdot r$ it also holds that $q \cdot r\downarrow$. This can only be satisfied by SOS rule 9, so $q\downarrow$.

   In either of the above cases we have $q\downarrow$; this satisfies the bisimulation relation requirement.

4. Suppose $q\downarrow$; we need to show that $p\downarrow$.

   This is analogous to the case for $p\downarrow$ above.

Summing up, the four requirements for a bisimulation relation are met, so $\mathcal{R}$ is indeed a bisimulation relation. This concludes our proof of the theorem. □

Note that it is required that r is deadlock-free when p and q are allowed to contain deadlock: take process $r = a.\mathbf{0} + \mathbf{1}$, then it can easily be shown that $\mathbf{1} \cdot r \leftrightarrows (a.\mathbf{0} + \mathbf{1}) \cdot r$, but obviously $\mathbf{1} \not\leftrightarrows a.\mathbf{0} + \mathbf{1}$. A counterexample without $\mathbf{1}$ is also possible: $a.\mathbf{0} \cdot a.\mathbf{0} \leftrightarrows a.\mathbf{0} \cdot \mathbf{0}$, but $a.\mathbf{0} \not\leftrightarrows \mathbf{0}$. Or take a counterexample without actions: $\mathbf{1} \cdot \mathbf{0} \leftrightarrows \mathbf{0} \cdot \mathbf{0}$, but $\mathbf{1} \not\leftrightarrows \mathbf{0}$.

This cancellation theorem concludes our preliminary results regarding recursion.