Adding sequential composition and termination to the linear time branching time spectrum

Nijland, L.

Award date:
2018

Technische Universiteit
Eindhoven
University of Technology

Department of Mathematics and Computer Science
Model Driven Software Engineering Research Group

Adding sequential composition and termination to the linear time - branching time spectrum

Lois Nijland

Supervisor:
dr. S.P. Luttik

Assessment Committee:
dr. S.P. Luttik
prof.dr. H. Zantema
dr.ing. A.J. Wijs

Final version (24-07)

Eindhoven, July 2018

Abstract

Van Glabbeek presented the linear time - branching time spectrum of behavioural semantics and gave sound, ground-complete axiomatisations for the process theory BCCSP. Groote and Chen et al. proved for the semantics in the spectrum whether there exist finite axiomatisations that are ω-complete, meaning that an equation can be derived if and only if all of its closed instantiations can be derived. In this paper we add termination and sequential composition to the spectrum by studying the semantics for the process theories BSP and TSP. We prove soundness and ground-completeness of BSP modulo bisimilarity, trace semantics and ready simulation semantics and we prove that the axioms of TSP admit an elimination theorem and this can be directly exploited to extend the results for BSP with sequential composition. We provide an overview of sound, ground-complete axiomatisations for TSP for the semantics in the spectrum. We present finite bases for BSP and TSP modulo the semantics in the spectrum when there are no actions and we prove that TSP modulo bisimulation is ω-complete when there is at least one action. We prove that for BSP modulo ready simulation semantics with infinitely many actions a finite basis exists, while for BSP modulo ready simulation with a finite number of actions of at least one such a finite basis does not exist.

Preface

This thesis is made for the completion of the master's program Computer Science and Engineering at Eindhoven University of Technology. I have researched and written this master thesis from February 2018 to July 2018.

At first, the goal of the thesis was to add termination to the linear time - branching time spectrum. So, I started by creating new definitions and axioms that were necessary for this. Only after I was done with this, I discovered that these results already existed, which meant that I had not found or proved anything new yet. Therefore, my research goal was expanded to other concepts. Fortunately, I very much enjoyed researching this as well.

During the project, I received valuable guidance from my supervisor dr. Bas Luttik, so I would like to thank him for this. I would also like to thank my fellow student Astrid Belder for her input and for having enjoyable lunch breaks together.

Eindhoven Univeristy of Tehcnology
July 2018

Lois Nijland

Contents

Contents **vi**

1 Introduction **1**

2 Preliminaries **3**
2.1 The linear time - branching time spectrum . . . 3
2.2 BCCSP . . . 4
2.3 BSP . . . 8
2.4 TSP . . . 9

3 Axiomatisations and Semantics **10**
3.1 Template for BSP . . . 10
3.2 Bisimulation semantics for BSP . . . 11
3.2.1 Definition . . . 11
3.2.2 Soundness of A1-A3, A6 for BSP(A) . . . 12
3.2.3 Ground-completeness BSP(A) . . . 12
3.3 Trace semantics for BSP . . . 12
3.3.1 Step 1: Extend definition . . . 13
3.3.2 Step 2: Prove soundness . . . 14
3.3.3 Step 3: Prove ground-completeness . . . 15
3.4 Ready simulation semantics for BSP . . . 16
3.4.1 Definition . . . 16
3.4.2 Soundness . . . 17
3.4.3 Ground-completeness . . . 20
3.5 Definitions and normal forms of semantics for BSP . . . 20
3.5.1 Completed trace semantics . . . 21
3.5.2 Readiness semantics . . . 21
3.5.3 Failures semantics . . . 22
3.5.4 Ready trace semantics . . . 23
3.5.5 Failure trace semantics . . . 23
3.5.6 Simulation semantics . . . 24
3.5.7 Completed simulation semantics . . . 24
3.5.8 Possible worlds semantics . . . 25
3.6 Axiomatisations for TSP . . . 26
3.6.1 Axioms A1-A10 . . . 26
3.6.2 Overview ground-complete axiomatisations for TSP(A) . . . 26

4 ω-Completeness **28**
4.1 BSP . . . 29
4.1.1 All semantics with $|A| = 0$ . . . 29
4.1.2 Ready simulation semantics with $1 \leq |A| < \infty$ . . . 31
4.1.3 Ready simulation semantics with $|A| = \infty$ . . . 34

4.2 TSP 35
4.2.1 All semantics with $|A| = 0$ 35
4.2.2 Bisimulation semantics with $|A| \geq 1$ 39
4.2.3 Ready simulation semantics with $|A| \geq 1$ 44
4.2.4 Infinite families of equations with $1 \leq |A| < \infty$ 47

5 Conclusions **49**

Bibliography **50**

Appendix **50**

A Soundness B1-B6 **51**
A.1 BSP 51
A.1.1 Soundness B1 51
A.2 TSP 51
A.2.1 Soundness B2 51
A.2.2 Soundness B3 52
A.2.3 Soundness B4 52
A.2.4 Soundness B5 52
A.2.5 Soundness B6 53
A.3 Infinite family of equations 53
A.3.1 Completed simulation semantics 53

Chapter 1

Introduction

Labelled transition systems model processes by describing states and transitions from a state, via an action, to a state. Behavioural semantics identify states of labelled transition systems that have the same observations using criteria of the specific behavioural semantics. Since there is no general consensus on which criteria should be used to identify labelled transitions systems, several behavioural semantics exist.

In [10] van Glabbeek presented the linear time - branching time spectrum of behavioural semantics. These semantics are based on simulation notions such as simulation semantics (S), completed simulation semantics (CS), possible worlds semantics (PW), ready simulation semantics (RS) and bisimulation semantics (B), or they are based on decorated versions of execution traces such as trace semantics (T), completed trace semantics (CT), failures semantics (F), readiness semantics (R), failure trace semantics (FT) and ready trace semantics (RT).

Van Glabbeek [10] studied these semantics for the process theory BCCSP, which includes the constant 0, action prefix and non-deterministic choice. He gave sound and ground-complete axiomatisations for these semantics, which means that two closed BCCSP terms are provably equal by the axioms if and only if they are behaviourally equivalent.

In [7] an overview is given that shows for each of the semantics in the spectrum whether there exists a finite axiomatisation that is ω-complete for BCCSP. That is, if all closed instances of an equation can be derived, does this imply that the equation itself can be derived from the axiomatisation using the rules of equational logic? We also refer to an ω-complete axiom system as a *basis* for the algebra it axiomatises. Whether such a finite basis exists may depend on the cardinality of the set of actions.

Results as given in [10] and [7] do not exist for the different semantics for a process theory that includes successful termination or sequential composition, such as BSP or TSP [2]. The benefit of adding successful termination and sequential composition, is that there will be a richer language for the different semantics.

Adding termination increases the expressiveness, since BSP is more expressive than BCCSP [2]. This means that an extension of the semantics is needed and some of the existing results have to be redone. Furthermore, we add termination, because when adding sequential composition we want to distinguish successful and unsuccessful termination.

When extending closed terms that denote finite processes with sequential composition, the expressiveness does not increase. In this case much of the existing results can be reused by proving that sequential composition can be eliminated for the finest equivalence. Yet when there are recursive specifications [2] sequential composition does increase the expressiveness. Since axioms of a theory's axiomatisation are typically sound for the same theory extended with recursion, it is useful to have an axiomatisation that includes sequential composition.

In [2] it is proved that BSP and TSP modulo bisimilarity are sound and ground-complete. Furthermore, sound and ground-complete axiomatisations are given for BSP modulo simulation, completed simulation, possible worlds, ready simulation, trace, completed trace, failures, readiness, failure trace and ready trace semantics in [2]. However, ground-complete axiomatisations are not

given for TSP modulo all of the semantics and an overview such as the one in van Glabbeek's spectrum is not provided.

As already mentioned, [7] provides an overview of which semantics in the spectrum are ω-complete for BCCSP. However, as far as we know, for no semantics finite bases or the lack thereof are proved for BSP or TSP. Therefore, the goal is to reconsider the results of [10] and [7] for the extended process theories BSP, which includes successful termination, and TSP, which includes successful termination and sequential composition.

We want to obtain an overview of ground-complete axiomatisations for TSP when reconsidering the results in [10]. To achieve this we first consider BSP and change the definitions of the semantics to include successful termination, which is done in [2]. Furthermore, in [2] it is shown which axioms are needed for ground-complete axiomatisations for BSP. However, only an outline of the ground-completeness proof for BSP modulo trace semantics is given in [2] and no ground-completeness proofs are given for BSP modulo the other semantics mentioned above. Therefore, we provide elaborate ground-completeness proofs for BSP modulo trace and ready simulation semantics. We choose these semantics specifically, since we then have a ground-completeness proof for a semantics based on simulations and one for a semantics based on decorated versions of execution traces.

We show how, in general, the ground-completeness of some semantics can be proved "efficiently" for BSP, i.e. in such a way that the proof can be partially reused for other semantics. The idea of this general method is to create normal forms using the specific axiom(s) of the semantics s in such a way that terms are s-equivalent if and only if they are bisimilar, so that we can reuse the ground-completeness result of BSP modulo bisimilarity.

After this, we consider TSP, which extends BSP with sequential composition. We find that the axioms for bisimilarity, which is the finest equivalence in the spectrum, admit an elimination theorem [2]. This can be directly exploited to extend the results for BSP with sequential composition.

When reconsidering the results in [7], we prove that there are finite bases for BSP and TSP modulo all of the semantics in the spectrum when there are no actions. We prove that for BSP modulo ready simulation semantics with a finite number of actions there does not exist a finite basis, while for BSP modulo ready simulation semantics with infinitely many actions a finite basis exists, which we prove using Groote's technique of inverted substitutions [11]. Furthermore, we prove that TSP modulo bisimilarity with at least one action is ω-complete. We observe that for TSP modulo ready simulation semantics, we cannot reuse any of the previous results. Finally, we show that the infinite families of equations given in [7], which prove the lack of finite bases for BCCSP modulo completed simulation and failure trace semantics, can be derived from a sound and finite collection of equations over TSP(A).

The remainder of this report is organised as follows. We start in Chapter 2 by giving the syntax, semantics and definitions of BCCSP and TSP. In Chapter 3 we give a template for proving soundness and ground-completeness for BSP, we prove this for BSP modulo bisimulation, trace semantics and ready simulation semantics and we give definitions of the other semantics taking successful termination into account. We prove that we can eliminate sequential composition for closed TSP(A)-terms and we provide an overview of ground-complete axiomatisations for TSP. After this, ω-completeness for TSP and BSP modulo bisimulation and ready simulation is studied in Chapter 4. This is followed by conclusions in Chapter 5.

Chapter 2

Preliminaries

We introduce the linear time - branching time spectrum as explained in [10]. After this, we give the syntax and axioms of BCCSP, together with definitions that will be used throughout this report and an overview of ground-complete axiomatisations for BCCSP modulo the semantics in the linear time - branching time spectrum. We introduce the process theory BSP, which includes successful termination, and give its structural operational rules. Finally, we give the syntax and axioms of TSP, which is a process theory that includes both successful termination and sequential composition.

2.1 The linear time - branching time spectrum

Van Glabbeek presents in [10] the linear time - branching time spectrum of behavioural semantics for finitely branching, concrete processes. We can use these behavioural semantics in the context of labelled transition systems. The definition for labelled transition systems from [10] is extended below to include successful termination.

Definition 2.1.1. A *labelled transition system* is a tuple $(\mathbb{P}, \rightarrow, \downarrow)$ with $\mathbb{P}$ a set of states and $\rightarrow \subseteq \mathbb{P} \times A \times \mathbb{P}$ a transition relation. The set of terminating or final states is denoted by $\downarrow \subseteq \mathbb{P}$. Write $p \xrightarrow{a} q$ for $(p, a, q) \in \rightarrow$. The set $I(p)$ consists of those labels a for which there exists a q such that $p \xrightarrow{a} q$. Let $a_1, \ldots, a_k$ be a sequence of labels and write $p \xrightarrow{a_1,\ldots,a_k} q$ if there are states $s_0, \ldots, s_k$ such that $p = s_0 \xrightarrow{a_1} \ldots \xrightarrow{a_k} s_k = q$. Notation $p\downarrow$ is used for $p \in \downarrow$ and it is said that p has a termination option. $p \notin \downarrow$ is denoted by $p\not\downarrow$.

Figure 2.1 depicts the linear time - branching time spectrum. We refer to the semantics traces (T), completed traces (CT), failures (F), readies (R), failure traces (FT), ready traces (RT) and possible futures (PF) collectively as *decorated trace semantics*, since they are based on decorated versions of execution traces. We refer to the semantics simulation (S), completed simulation (CS), possible worlds (PW), ready simulation (RS), 2-nested simulation (2S) and bisimulation (B) collectively as *simulations*, since they are based on simulation [7]. In [10] these semantics are motivated by the observable behaviour of processes, according to some testing scenarios. These motivations are captured in terms of button pushing experiments.
In Figure 2.1 an arrow from one semantics to another means that the source of the arrow is finer (makes less identifications) than the target. To express this we use the symbol $\subseteq$, for example we state that $B \subseteq RS$, by which we mean that the equivalence associated with bisimilarity is finer than the equivalence associated with ready simulation semantics.

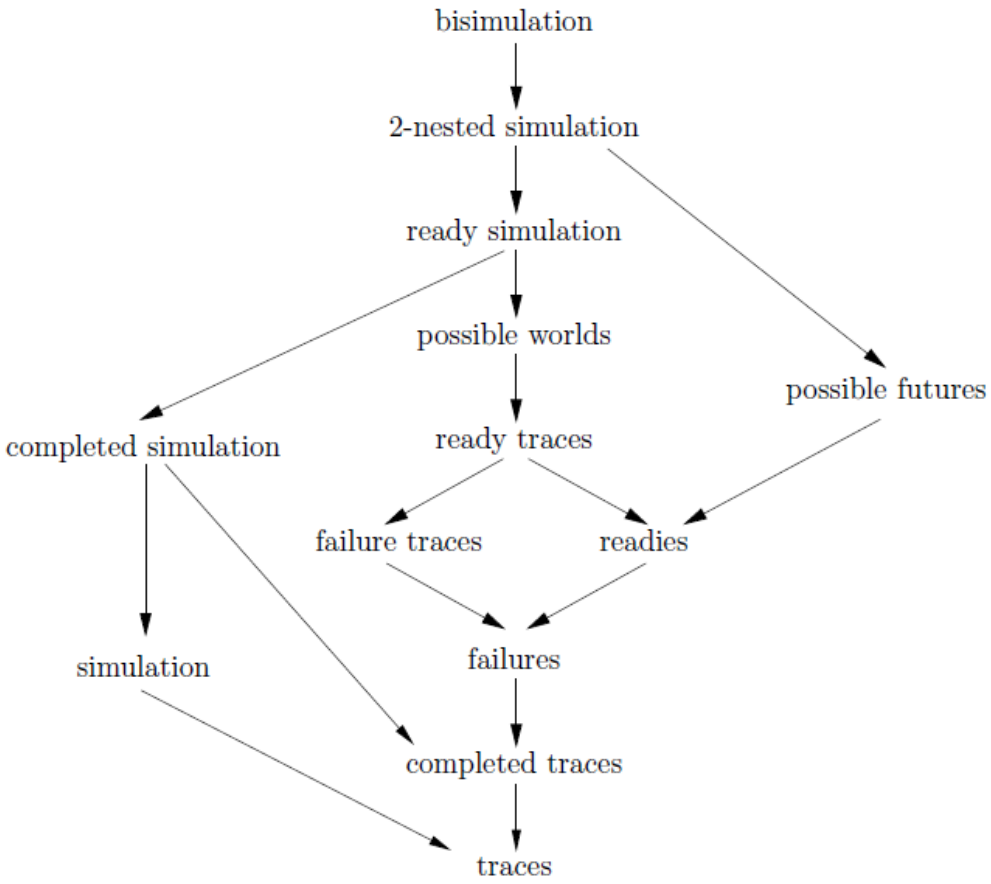


Figure 2.1: The linear time - branching time spectrum [10]

We provide a definition for bisimulation semantics below. Definitions of the other semantics are given in later sections.

Definition 2.1.2 (Bisimulation Semantics (B))**.** A **bisimulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.
2. If pRq and $q \xrightarrow{a} q'$, then $\exists p' : p \xrightarrow{a} p'$ and $p'Rq'$.

Two processes p and q are **bisimilar**, notation $p \leftrightarrows q$, if there exists a bisimulation R with pRq.

2.2 BCCSP

The process theory BCCSP is a language to specify transition systems. The semantics mentioned in Section 2.1 are applied to BCCSP and for each of them a ground-complete axiomatisation is given in [10].
BCCSP's signature consists of the constant 0, the binary operator +, and unary prefix operators $a.$, where a ranges over a non-empty set A of actions. Denote BCCSP over this set of actions by BCCSP(A). Intuitively, closed BCCSP(A)-terms, denoted by p, q, r represent finite behaviours. The constant 0 represents a process that does not exhibit any behaviour. The binary operator + represents alternative composition: $p + q$ is the process that either executes p or executes q (but

not both). The unary operator $a.$ represents prefix: $a.p$ executes action a to transform into process p.
On the set of BCCSP(A)-terms, there are binary relations $\xrightarrow{a}$, for each $a \in A$. The intuitive idea is that $t \xrightarrow{a} s$ denotes that t can execute a and thereby turn into s.
The structural operational rules for BCCSP(A), in which a ranges over A, are given in Figure 2.2. They give rise to A-labelled transitions between BCCSP(A)-terms.

$$a.x \xrightarrow{a} x \qquad \frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

Figure 2.2: BCCSP(A) Structural Operational Rules [7]

BCCSP(A) is axiomatised by the axioms in Table 2.1. These axioms allow one to reason about the equivalence of two processes in BCCSP(A) and about the behavioural equivalence of the corresponding labelled transition systems.

$x + y$	$=$	$y + x$	A1
$(x + y) + z$	$=$	$x + (y + z)$	A2
$x + x$	$=$	x	A3
$x + 0$	$=$	x	A6

Table 2.1: BCCSP(A) Axioms [2]

Let $\mathcal{O} = \{T, S, CT, CS, F, R, FT, RT, PW, RS, B\}$. For all $o \in \mathcal{O}$ we have a preorder relation $\sqsubseteq_o$, where $p \sqsubseteq_o q$ if p is in some way simulated by q, or if the decorated traces of p are included in those of q.

Definition 2.2.1. [10] For all $o \in \mathcal{O}$ p is **equivalent** to q in o-semantics, denoted by $p =_o q$, iff $p \sqsubseteq_o q \wedge q \sqsubseteq_o p$.

We consider the three process theories BCCSP (Section 2.2), BSP (Section 2.3) and TSP (Section 2.4). Since the following definitions apply to all three of these theories, let $X \in$ {BCCSP(A), BSP(A), TSP(A)}.

Definition 2.2.2. For any process theory X, write $\mathrm{X}_{a_1,\ldots,a_n}$ for the process theory consisting of the axioms of X and the specific axiom(s) a_i where $1 \leq i \leq n$ and $n \geq 1$.

For example, according to Definition 2.2.2, write $\mathrm{BCCSP}_{A4,A5}(A)$ for the process theory consisting of the axioms of BCCSP(A) (Table 2.1) and the axioms A4 and A5.

Definition 2.2.3. An X-term is closed if it does not contain any occurrence of a variable. We denote the set of closed X-terms by $T(\mathrm{X})$.

Definition 2.2.4. For any process theory X, write $\mathrm{X} \vdash p = q$ (sometimes abbreviated as $p = q$) if the equation $p = q$ can be derived from the axioms of X using the rules of equational logic.

For example, according to Definition 2.2.4, we write:

$$BCCSP(A) \vdash (a.(b.0 + c.0) + b.c.0 + d.0) + b.c.0 = a.(b.0 + c.0 + 0) + (b.c.0 + d.0)$$

This is because we have:

$$\begin{aligned}(a.(b.0 + c.0) + b.c.0 + d.0) + b.c.0 &\overset{A2}{=} a.(b.0 + c.0) + (b.c.0 + d.0 + b.c.0)\\ &\overset{A1}{=} a.(b.0 + c.0) + (b.c.0 + b.c.0 + d.0)\\ &\overset{A3}{=} a.(b.0 + c.0) + (b.c.0 + d.0)\\ &\overset{A6}{=} a.(b.0 + c.0 + 0) + (b.c.0 + d.0)\end{aligned}$$

Definition 2.2.5. For two X-terms p and q, $p \equiv q$ denotes that p and q are syntactically equal.

Definition 2.2.6. An expression p^*, where $p^* = a.p'$ or $p^* = 0$ is called a **summand** of a BCCSP(A)-term p, denoted by $p^* \preccurlyeq p$, if up to associativity and commutativity of $+$, p can be written as $\sum P$ with $p^* \in P$.

For example, according to Definition 2.2.6 the expressions $a.(b.0 + c.0)$ and 0 are summands of p if $p = a.(b.0 + c.0) + 0$.

Definition 2.2.7. The **depth** of an X-term t, denoted by $depth(t)$ is the length of the longest trace that t can exhibit, i.e.

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

Definition 2.2.8. For any process theory X and $o \in \mathcal{O}$, the process theory X is **sound** for the algebra of closed X-terms modulo o-semantics if, for all closed X-terms p and q, $X \vdash p = q$ implies $p =_o q$.

Definition 2.2.9. For any process theory X and $o \in \mathcal{O}$, the process theory X is **ground-complete** for the algebra of closed X-terms modulo o-semantics if, for all closed X-terms p and q, $p =_o q$ implies $X \vdash p = q$.

In [10] axiomatisations are given for all of the semantics in Figure 2.1 except for 2-nested simulation and possible futures. According to van Glabbeek the reason for this is that the axioms for these semantics are more cumbersome and the corresponding testing scenarios are less plausible. Therefore, we will not consider 2-nested simulation and possible futures in this report. The ground-complete axiomatisations for the semantics are given in Table 2.2.

Theorem 2.2.10. *For each of the semantics $o \in \mathcal{O}$, two closed BCCSP(A)-terms p, q are o-equivalent iff they can be proved equal from the axioms marked with "+" in the column for o in Table 2.2. The axioms marked with "v" or "ω" are valid in o-semantics but not needed for the proof.*

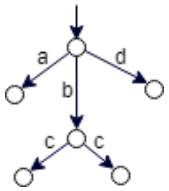
Proof. The soundness and ground-completeness proofs are given in [10]. □

Axiom name	Axiom	B	RS	PW	RT	FT	R	F	CS	CT	S	T
A1	$x + y = y + x$	+	+	+	+	+	+	+	+	+	+	+
A2	$(x + y) + z = x + (y + z)$	+	+	+	+	+	+	+	+	+	+	+
A3	$x + x = x$	+	+	+	+	+	+	+	+	+	+	+
A6	$x + 0 = x$	+	+	+	+	+	+	+	+	+	+	+
Rs	$I(x) = I(y) \Rightarrow a.(x + y) = a.(x + y) + a.y$		+	v	v	v	v	v	v	v	v	v
Pw	$a.(b.x + b.y + z) = a.(b.x + z) + a.(b.y + z)$			+	v	v	v	v		v		v
Ft_2, Rt	$I(x) = I(y) \Rightarrow a.x + a.y = a.(x + y)$				+	+	v	v		v		v
Ft_1	$a.x + a.y = a.x + a.y + a.(x + y)$					+		v		v		v
Fa_2, Re	$a.(b.x + u) + a.(b.y + v) = a.(b.x + b.y + u) + a.(b.y + v)$						+	+		v		v
Fa_1	$a.x + a.(y + z) = a.x + a.(x + y) + a.(y + z)$							+		ω		v
Cs	$a.(x + b.y + z) = a.(x + b.y + z) + a.(b.y + z)$								+	v	v	v
Ct	$a.(b.x + u) + a.(c.y + v) = a.(b.x + c.y + u + v)$									+		v
Si	$a.(x + y) = a.(x + y) + a.y$										+	v
Tr	$a.x + a.y = a.(x + y)$											+
	$I(0) = 0$	+	+	+	+	+	+	+	+	+	+	+
	$I(a.x) = a.0$	+	+	+	+	+	+	+	+	+	+	+
	$I(x + y) = I(x) + I(y)$	+	+	+	+	+	+	+	+	+	+	+

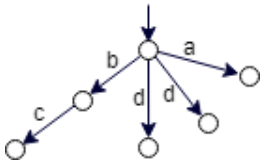
Table 2.2: Ground-complete axiomatisations for BCCSP(A) modulo the semantics [10]

We give an example to illustrate how labelled transition systems, processes, axioms, and behavioural semantics are related to each other.

Example 2.2.11. Process $p = (a.0 + b.(c.0 + c.0)) + d.0$ and process $q = (b.(c.0) + d.0 + d.0) + a.0$ are provably equal modulo bisimilarity. This can be seen by applying the axioms A1-A3 from Table 2.2. The labelled transition systems of process p and process q are shown in Figure 2.3a and Figure 2.3b respectively.



(a) $(a.0 + b.(c.0 + c.0)) + d.0$



(b) $(b.(c.0) + d.0 + d.0) + a.0$

Figure 2.3: Equivalent processes modulo bisimilarity

By soundness it should hold that these processes are also bisimilar. We can verify this by using Definition 2.1.2. Note that since BCCSP is ground-complete modulo bisimilarity we could also reason the other way around: we have processes p, q which are bisimilar, then they should also be provably equal from the axioms of BCCSP.

2.3 BSP

The process theory BSP (for Basic Sequential Processes) is a language to specify transition systems and is described in [2]. The signature of BSP consists of the constants 0 and 1, the unary prefix operators $a.$, where a ranges over a non-empty set A of actions, and the binary operator $+$. Denote BSP over this set of actions by BSP(A).
The constant 0, the unary operator $a.$ and the binary operator $+$ represent the same behaviour as explained in Section 2.2. The constant 1 represents a process that has a termination option.
On the set of BSP(A)-terms, there are binary relations $\xrightarrow{a}$ for each $a \in A$. The intuitive idea of $\xrightarrow{a}$ is the same as in Section 2.2. There is also a termination predicate $\downarrow$, $s\downarrow$ denotes that s has a termination option.
The structural operational rules for BSP(A), in which a ranges over A, are given in Figure 2.4. They give rise to A-labelled transitions between BSP(A)-terms.

$$a.x \xrightarrow{a} x \qquad \frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

$$1\downarrow \qquad \frac{x\downarrow}{(x+y)\downarrow} \qquad \frac{y\downarrow}{(x+y)\downarrow}$$

Figure 2.4: BSP(A) Structural Operational Rules [2]

BSP(A) is axiomatised by axioms A1-A3 and A6 in Table 2.1. These axioms allow one to reason about the equivalence of two processes in BSP(A) and about the behavioural equivalence of the corresponding labelled transition systems.

Definition 2.3.1. An expression p^*, where $p^* = a.p'$, $p^* = 1$ or $p^* = 0$ is called a **summand** of a BSP(A)-term p, denoted by $p^* \trianglelefteq p$, if up to associativity and commutativity of $+$, p can be written as $\sum P$ with $p^* \in P$.

An example of a labelled transition system corresponding to process $a.b.0+a.(c.0+d.1+1)$ is shown in Figure 2.5.

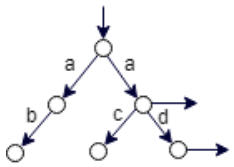


Figure 2.5: Process $a.b.0 + a.(c.0 + d.1 + 1)$

2.4 TSP

The process theory TSP (for Theory of Sequential Processes) is a language to specify transition systems and is described in [2]. The signature of TSP consists of the constants 0 and 1, the unary prefix operators $a.$, where a ranges over a non-empty set A of actions, and two binary operators, $+$ and $\cdot$. Denote TSP over this set of actions by TSP(A).
The constants 0 and 1, the unary operator $a.$ and the binary operator $+$ represent the same behaviour as explained in Section 2.2 and Section 2.3. The binary operator $\cdot$ represents sequential composition: $x \cdot y$ is the process that first executes x, and upon completion of x starts y.
On the set of TSP(A)-terms, there are binary relations $\xrightarrow{a}$ for each $a \in A$ and termination predicates $\downarrow$. The intuitive ideas of $\xrightarrow{a}$ and $\downarrow$ are the same as in Section 2.2 and Section 2.3.
The structural operational rules for TSP(A), in which a ranges over A, are given in Figure 2.6. They give rise to A-labelled transitions between TSP(A)-terms.

$$a.x \xrightarrow{a} x \qquad \frac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x+y \xrightarrow{a} y'}$$

$$1\downarrow \qquad \frac{x\downarrow}{(x+y)\downarrow} \qquad \frac{y\downarrow}{(x+y)\downarrow}$$

$$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y} \qquad \frac{x\downarrow \quad y \xrightarrow{a} y'}{x \cdot y \xrightarrow{a} y'} \qquad \frac{x\downarrow \quad y\downarrow}{(x \cdot y)\downarrow}$$

Figure 2.6: TSP(A) Structural Operational Rules [2]

TSP(A) is axiomatised by axioms A1-A10 in Table 2.1 and Table 2.3. These axioms allow one to reason about the equivalence of two processes in TSP(A) and about the behavioural equivalence of the corresponding labelled transition systems.

$(x+y)\cdot z$	$=$	$x\cdot z + y\cdot z$	A4
$(x\cdot y)\cdot z$	$=$	$x\cdot(y\cdot z)$	A5
$0\cdot x$	$=$	0	A7
$x\cdot 1$	$=$	x	A8
$1\cdot x$	$=$	x	A9
$a.x\cdot y$	$=$	$a.(x\cdot y)$	A10

Table 2.3: TSP(A) Axioms [2]

Definition 2.4.1. An expression p^*, where $p^* = p_1 \cdot p_2$, $p^* = a.p'$, $p^* = 1$ or $p^* = 0$ is called a **summand** of a TSP(A)-term p, denoted by $p^* \trianglelefteq p$, if up to associativity and commutativity of $+$, p can be written as $\sum P$ with $p^* \in P$.

Chapter 3

Axiomatisations and Semantics

In this section we first consider BSP, which contains successful termination. We provide a template that uses BSP(A)-terms to prove soundness and ground-completeness. We use the template to prove soundness and ground-completeness of TSP modulo trace semantics and we use a method given in [10] to prove soundness and ground-completeness of ready simulation semantics. Furthermore, for all $o \in \mathcal{O}$ we give definitions of the semantics o taking successful termination into account. After this, we consider TSP, which contains successful termination as well as sequential composition, and we show that the results of BSP can be reused. Finally, we provide an overview of sound and ground-complete axiomatisations for TSP.

3.1 Template for BSP

A general technique to prove soundness and ground-completeness for BSP(A) modulo o-semantics, where $o \in \mathcal{O}$, is described below. The idea is that we first change the definition of o-semantics to incorporate termination and prove soundness of the axiom(s) for BSP(A) modulo o-semantics. After this, we prove ground-completeness of BSP(A) modulo o-semantics by defining a normal form for o-semantics and proving that two terms p, q in o normal form are o-equivalent iff the terms p, q are o'-equivalent, for some finer semantics $o' \in \mathcal{O}$ such that BSP(A) modulo o'-semantics is ground-complete.

Step 1: Extend definition
Since there is no distinction between successful and unsuccessful termination in van Glabbeek's spectrum and since this is relevant for BSP(A), we extend the definition of o-semantics to transition systems that take this distinction into account. These definitions are provided in [2].

Step 2: Prove soundness
Let the sound axiomatisation for BSP(A) modulo o-semantics as defined in the definition including termination consist of axioms A1-A3, A6 and Sa. Sa denotes the set of specific axioms that are sound for BSP(A) modulo o-semantics. Soundness of axiom $A_i \in Sa$ is proved by showing that the left-hand side of A_i is o-equivalent to the right-hand side for all closed BSP(A)-terms. Soundness of some axiom Ax for BSP(A) modulo o-semantics follows from soundness of Ax for BSP(A) modulo o'-semantics when $o' \subseteq o$.

Step 3: Prove ground-completeness
To prove ground-completeness we follow the steps.

1. Define a set of terms in o *normal form* using the axioms in Sa.
2. Prove that for every BSP(A)-term p there is a p' in o normal form such that $\text{BSP}_{Sa}(\text{A}) \vdash p = p'$.

3. Let o' be some semantics such that $o' \subseteq o$ and BSP(A) modulo o' is proved to be ground-complete. Prove that if p, q are in o normal form, then $p =_o q \Leftrightarrow p =_{o'} q$. This is split into two parts.
   - Prove that $p =_o q \Rightarrow p =_{o'} q$.
   - $p =_{o'} q \Rightarrow p =_o q$ follows from the fact that $o' \subseteq o$.

So, we have that for every BSP(A)-term there is an equivalent term in o normal form. Furthermore, we have that two terms in o normal form are o-equivalent iff they are o'-equivalent for some $o' \subseteq o$. From this and from the fact that BSP(A) is ground-complete for the algebra of closed BSP(A)-terms modulo o'-semantics it follows that $\mathrm{BSP}_{Sa}(\mathrm{A})$ is ground-complete for the algebra of closed BSP(A)-terms modulo o-semantics.

3.2 Bisimulation semantics for BSP

We give a definition of bisimilarity incorporating termination and show that BSP(A) is sound and ground-complete for the algebra of closed BSP(A)-terms modulo bisimilarity. We do this for bisimulation because $\forall_{o \in \mathcal{O}} : B \subseteq o$, which means that we can use the soundness and ground-completeness results for o-semantics.

3.2.1 Definition

To distinguish between successful and unsuccessful termination, we change the definition of bisimilarity (Definition 2.1.2) for BSP(A) to the following [2].

Definition 3.2.1 (Bisimulation Semantics (B) - Termination)**.** A **bisimulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.
2. If pRq and $q \xrightarrow{a} q'$, then $\exists p' : p \xrightarrow{a} p'$ and $p'Rq'$.
3. If pRq and $p\downarrow$, then $q\downarrow$.
4. If pRq and $q\downarrow$, then $p\downarrow$.

Two closed BSP(A)-terms p and q are **bisimilar**, notation $p \leftrightarrows q$, if there exists a bisimulation R with pRq, satisfying conditions 1-4.

Definition 3.2.1 differs from Definition 2.1.2 because it includes conditions about termination (conditions 3 and 4). This distinguishes, for example, processes $a.0$ and $a.1$.

The following two lemmas are presented because they are useful when proving ground-completeness of BSP modulo other semantics in later sections.

Lemma 3.2.2. *If $p' \leftrightarrows q'$, $p = a.p'$, $q = a.q'$, then $p \leftrightarrows q$.*

Proof. Since $p' \leftrightarrows q'$, there is a bisimulation relation R' such that $p'R'q'$. We prove that $R = R' \cup \{(p, q)\}$ is a bisimulation relation as defined in Definition 3.2.1 by showing that the conditions 1-4 are satisfied.

1. We have pRq and $p \xrightarrow{a} p'$. We have $q \xrightarrow{a} q'$ and $p'R'q'$. Since R includes R' this holds.
2. We have pRq and $q \xrightarrow{a} q'$. We have $p \xrightarrow{a} p'$ and $p'R'q'$. Since R includes R' this holds.
3. Since pRq, we have if $p\downarrow$, then also $q\downarrow$. However, $p = a.p'$, which means that $p \not\downarrow$ and so this holds.

4. Since pRq, we have if $q\downarrow$, then also $p\downarrow$. However, $q = a.q'$, which means that $q \not\downarrow$ and so this holds.

□

Lemma 3.2.3. *If $p_1 \leftrightarrows q_1$, $p_2 \leftrightarrows q_2$, $p = p_1 + p_2$ and $q = q_1 + q_2$, then $p \leftrightarrows q$.*

Proof. Since $p_1 \leftrightarrows q_1$ and $p_2 \leftrightarrows q_2$, there are bisimulation relations R_1 and R_2 such that $p_1 R_1 q_1$ and $p_2 R_2 q_2$. We prove that $R = R_1 \cup R_2 \cup \{(p, q)\}$ is a bisimulation relation as defined in Definition 3.2.1 by showing that the conditions 1-4 are satisfied.

1. Suppose that we have pRq and $p \xrightarrow{a} p'$. According to the operational rules in Figure 2.6 this can be because $p_1 \xrightarrow{a} p'$, in which case also $q \xrightarrow{a} q'$ such that $p' R q'$ since $p_1 R_1 q_1$ and R includes R_1, or because $p_2 \xrightarrow{a} p'$, in which case also $q \xrightarrow{a} q'$ such that $p' R q'$ since $p_2 R_2 q_2$ and R includes R_2.

2. Suppose that we have pRq and $q \xrightarrow{a} q'$. According to the operational rules in Figure 2.6 this can be because $q_1 \xrightarrow{a} q'$, in which case also $p \xrightarrow{a} p'$ such that $p' R q'$ since $p_1 R_1 q_1$ and R includes R_1, or because $q_2 \xrightarrow{a} q'$, in which case also $p \xrightarrow{a} p'$ such that $p' R q'$ since $p_2 R_2 q_2$ and R includes R_2.

3. Suppose that we have pRq and $p\downarrow$. According to the operational rules in Figure 2.6 this can be because $p_1\downarrow$, in which case also $q\downarrow$ since $p_1 R_1 q_1$ and R includes R_1, or because $p_2\downarrow$, in which case also $q\downarrow$ since $p_2 R_2 q_2$ and R includes R_2.

4. Suppose that we have pRq and $q\downarrow$. According to the operational rules in Figure 2.6 this can be because $q_1\downarrow$, in which case also $p\downarrow$ since $p_1 R_1 q_1$ and R includes R_1, or because $q_2\downarrow$, in which case also $p\downarrow$ since $p_2 R_2 q_2$ and R includes R_2.

□

3.2.2 Soundness of A1-A3, A6 for BSP(A)

The axiomatisation of BSP(A) modulo bisimilarity consists of axioms A1-A3 and A6 in Table 2.1.

Theorem 3.2.4. *BSP(A) is sound for the algebra of closed BSP(A)-terms modulo bisimilarity.*

Proof. This is proved in [2]. □

Corollary 3.2.5 follows from Theorem 3.2.4 and from the fact that $\forall_{o\in\mathcal{O}} : B \subseteq o$.

Corollary 3.2.5. *For all $o \in \mathcal{O}$ BSP(A) is sound for the algebra of closed BSP(A)-terms modulo o-semantics.*

3.2.3 Ground-completeness BSP(A)

Theorem 3.2.6. *BSP(A) is ground-complete for the algebra of closed BSP(A)-terms modulo bisimilarity.*

Proof. This is proved in [2]. □

3.3 Trace semantics for BSP

In the following sections we give a definition of trace semantics for BSP(A) and we prove soundness and ground-completeness using the technique presented in Section 3.1.

3.3.1 Step 1: Extend definition

The following definition is given in [10] for trace semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.3.1 (Trace Semantics (T))**.** $\sigma \in A^+$ is a **trace** of a process p if there is a process q such that $p \xrightarrow{\sigma} q$. Let $T(p)$ denote the set of traces of p. Two processes p and q are **trace equivalent**, notation $p =_T q$, if $T(p) = T(q)$.

In [10] a testing scenario is given, which states that trace semantics can be explained with the following *tracachine machine*.

"The process is modelled as a black box that contains as its interface to the outside world a display on which the name of the action is shown that is currently carried out by the process. The process autonomously chooses an execution path that is consistent with its position in the labelled transition system. During this execution an action name is always visible on the display and the last action name remains visible in the display if a deadlock occurs (unless a deadlock occurs at the start)."

"On this machine traces can be recorded, but stagnation cannot be detected, since in case of deadlock the observer may think that the last action is still continuing. Two processes are identified if they allow the same set of observations in this sense."

To distinguish between successful and unsuccessful termination, we would change the trace machine described above to display a 1 in case the process is terminating. To incorporate this, we change the definition of trace semantics for BSP(A). The definition is essentially the same as the definition given in [2], but the notation is slightly different. We use the following definition because it corresponds more closely to Definition 3.3.1.

Definition 3.3.2 (Trace Semantics (T) - Termination)**.** $\sigma \in A^+$ is a **trace** of a process p if there is a process q such that $p \xrightarrow{\sigma} q$. Let $T(p)$ denote the set of traces of p. $\sigma \in A^+$ is a **1-trace** of a process p if there is a process q such that $p \xrightarrow{\sigma} q$ and $q\downarrow$. Let $T_1(p)$ denote the set of 1-traces of p. Two processes p and q are **trace equivalent**, notation $p =_T q$, if $T(p) = T(q)$ and $T_1(p) = T_1(q)$.

Definition 3.3.2 differs from Definition 3.3.1 because it includes the notion of 1-traces. This is used to distinguish traces that have a termination option, for example to distinguish processes $a.0$ and $a.1$. Moreover, we still use the notion of traces, since otherwise the processes $a.1 + b.0$ and $a.1$ would be identified.

Example 3.3.3. In Figure 3.1 a labelled transition system corresponding to process $a.(b.0 + c.0 + d.1 + 1)$ is shown. We have $T(a.(b.0 + c.0 + d.1 + 1)) = \{a, ab, ac, ad\}$ and $T_1(a.(b.0 + c.0 + d.1 + 1)) = \{a, ad\}$. Note that these sets of traces and 1-traces are equal to the sets of traces and 1-traces of the process shown in Figure 2.5.

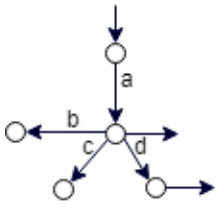


Figure 3.1: Process $a.(b.0 + c.0 + d.1 + 1)$

For trace semantics (T) we have axiom Tr (Table 2.2). The following definition and two lemmas are presented because they are useful to prove soundness of axiom Tr for BSP(A) modulo trace semantics.

Definition 3.3.4. Let $a \in A$ and let T be a set of traces, then $a.T = \{a\sigma | \sigma \in T\}$.

Lemma 3.3.5. *$T_1(a.s) = a.T_1(s)$ and $T(a.s) = a.T(s)$ for all closed BSP(A)-terms s and actions $a \in A$.*

Proof. $T_1(a.s) = \{\sigma | a.s \xrightarrow{\sigma} s' \downarrow\}$. From $a.s \xrightarrow{\sigma} s' \downarrow$ it follows that $a.s \xrightarrow{b} s'' \xrightarrow{\sigma'} s' \downarrow$ where $\sigma = b\sigma'$. Using the operational semantics of BSP(A) in Figure 2.4, we derive $b = a$ and $s'' = s$. So we have $T_1(a.s) = \{a\sigma' | s \xrightarrow{\sigma'} s' \downarrow\} = a.\{\sigma' | s \xrightarrow{\sigma'} s' \downarrow\} = a.T_1(s)$.
$T(a.s) = a.T(s)$ follows using similar reasoning. □

Lemma 3.3.6. *$T_1(s+t) = T_1(s) \cup T_1(t)$ and $T(s+t) = T(s) \cup T(t)$ for all closed BSP(A)-terms s, t.*

Proof. We need to prove that $\sigma \in T_1(s+t) \Leftrightarrow \sigma \in T_1(s) \cup T_1(t)$ for all σ.
($\Rightarrow$)
$T_1(s+t) = \{\sigma | s+t \xrightarrow{\sigma} u \downarrow\}$. From this it follows that $s+t \xrightarrow{a} u' \xrightarrow{\sigma'} u \downarrow$ where $\sigma = a\sigma'$. Using the operational semantics of BSP(A) in Figure 2.4, we derive that a must be an action of s or t. Hence, $\sigma \in T_1(s) \cup T_1(t)$.
($\Leftarrow$)
$T_1(s) = \{\sigma | s \xrightarrow{\sigma} u \downarrow\}$. From this it follows that $s \xrightarrow{a} u' \xrightarrow{\sigma'} u \downarrow$ where $\sigma = a\sigma'$. Using the operational semantics of BSP(A) in Figure 2.4, we know that $s+t$ can also do this a action. Hence, $\sigma \in T_1(s+t)$. Using similar reasoning, we find if $\sigma \in T_1(t)$ then $\sigma \in T_1(s+t)$.
$T(s+t) = T(s) \cup T(t)$ follows using similar reasoning □

3.3.2 Step 2: Prove soundness

The axiomatisation of BSP(A) modulo trace semantics consists of axioms A1-A3, A6 and Tr (Table 2.2). Axiom Tr is sound for BSP(A) modulo trace semantics as defined in Definition 3.3.2. This is proved in the following lemma.

Lemma 3.3.7 (Tr-Traces-BSP)**.** *$a.p + a.q =_T a.(p+q)$ for all closed BSP(A)-terms p, q and $a \in A$.*

Proof. We need to prove that for every 1-trace $\sigma \in T_1(a.p + a.q)$, also $\sigma \in T_1(a.(p+q))$ and vice versa. So, suppose there is a 1-trace $\sigma \in T_1(a.p + a.q) = T_1(a.p) \cup T_1(a.q)$ (by Lemma 3.3.6). There are two cases: $\sigma \in T_1(a.p)$ or $\sigma \in T_1(a.q)$.
First suppose that $\sigma \in T_1(a.p) = a.T_1(p)$ (by Lemma 3.3.5). This means that σ starts with a and then continues with a trace $\tau \in T_1(p)$. Since $\tau \in T_1(p)$, also $\tau \in T_1(p+q)$, and by Lemma 3.3.6, $\tau \in T_1(p) \cup T_1(q)$. Hence, since $\sigma = a\tau \in a.(T_1(p) \cup T_1(q)) = a.T_1(p+q) = T_1(a.(p+q))$, this proves that if $\sigma \in T_1(a.p)$ then $\sigma \in T_1(a.(p+q))$. Similarly, one can derive that if $\sigma \in T_1(a.q)$ then $\sigma \in T_1(a.(p+q))$. It follows that if $\sigma \in T_1(a.p + a.q)$ then $\sigma \in T_1(a.(p+q))$. The proof for the converse is similar and therefore omitted.
Furthermore, the proof that $\sigma \in T(a.p + a.q) \Leftrightarrow \sigma \in T(a.(p+q))$ is similar and therefore omitted. Hence, $T(a.p + a.q) = T(a.(p+q))$ and $T_1(a.p + a.q) = T_1(a.(p+q))$. From this we conclude that $a.p + a.q =_T a.(p+q)$. □

Theorem 3.3.8. *$BSP_{Tr}(A)$ is sound for the algebra of closed BSP(A)-terms modulo trace semantics.*

Proof. From Corollary 3.2.5 soundness of axioms A1-A3 and A6 for BSP(A) modulo trace semantics follows. Furthermore, soundness of Tr is proved in Lemma 3.3.7. □

3.3.3 Step 3: Prove ground-completeness

As described in the technique from Section 3.1 we start by introducing the notion of T normal form. Intuitively, terms in T normal form are BSP(A)-terms to which axiom Tr cannot be applied from left to right at any depth.

Definition 3.3.9. Let $t = \sum_{i=1}^{n} a_i.t_i$ (+1), modulo A1 and A2, be a BSP(A)-term, where $a_i \in A$. If $n = 0$ and t does not contain the summand 1 then $t = 0$. In all other cases, the constant 0 is removed using A6. We say that t is in **T normal form** if for all a_i, a_j, where $1 \leq i, j \leq n$ and $i \neq j$ it holds that $a_i \not\equiv a_j$ and if we recursively have that t_i (at any depth of t) is in T normal form.

Lemma 3.3.10. *For every BSP(A)-term p, there is a p' in T normal form such that $BSP_{Tr}(A) \vdash p = p'$.*

Proof. The proof is by induction on the depth of p.

Base case
If $depth(p) = 0$, then $p \equiv 0$ or $p \equiv 1$, and hence p is in T normal form.

Inductive step
For BSP(A)-term p with $depth(p) = n$ there is a p' in T normal form such that $\text{BSP}_{Tr}(\text{A}) \vdash p = p'$ (IH). We now prove that this also holds for BSP(A)-term p with $depth(p) = n + 1$.
Suppose $p = \sum_{i=1}^{m} a_i.p_i$ (+1) and $depth(p) = n + 1$. We know that for all $1 \leq i \leq m$, $depth(p_i) \leq n$, since otherwise $depth(p) > n + 1$. Hence, by IH we may assume that for all $1 \leq i \leq m$, p_i is in T normal form. For any a_j, a_k, where $1 \leq j, k \leq m$, such that $a_j \equiv a_k$, then by axiom Tr we write this as $a_j.(p_j + p_k)$ and we remove the terms $a_j.p_j$ and $a_k.p_k$. The resulting p is in T normal form. □

Before we prove that two terms p, q in T normal form are trace equivalent iff p, q are bisimilar, we prove two lemmas that will be useful. For this, recall Definition 2.3.1 of summand.

Lemma 3.3.11. *If $p = a.p'$, $p =_T q$ and p and q are in T normal form then $q = a.q'$ such that $p' =_T q'$.*

Proof. Since $p =_T q$, and p has a summand that starts with an a action, then q must at least contain a summand of the form $a.q'$. Since q is in T normal form, it has the form $q = \sum_{i=1}^{n} a_i.q_i$ (+1) (where $a_i \not\equiv a_j$ when $i \neq j$ and for any i, q_i is in T normal form), so it cannot have two summands starting with action a. Furthermore, it cannot be the case that $a_j \equiv b$ for some $b \not\equiv a$, since then q would have a trace starting with b, which would contradict $p =_T q$. Hence, $q = a.q'$.
We have $T(p) = T(q)$ and so $T(a.p') = T(a.q')$. By Lemma 3.3.5 and because a is an action, we find that $T(p') = T(q')$. By similar reasoning $T_1(p') = T_1(q')$ is obtained. We conclude that $p' =_T q'$. □

Lemma 3.3.12. *If $p = p_1 + p_2$, $p =_T q$ and p and q are in T normal form then $q = q_1 + q_2$ such that $p_1 =_T q_1$ and $p_2 =_T q_2$.*

Proof. Since p is in T normal form, it has the form $p = \sum_{i=1}^{n} a_i.p_i$ (+1). Suppose $I(p_1) = \mathcal{A}$ and $I(p_2) = \mathcal{B}$. Then since p is in T normal form, we have $\mathcal{A} \cap \mathcal{B} = \varnothing$.
Because $p =_T q$, q must have summands that start with all $a \in \mathcal{A}$ and all $b \in \mathcal{B}$. So, modulo A1 and A2, q has at least the summands q_1, q_2 such that $I(q_1) = \mathcal{A}$ and $I(q_2) = \mathcal{B}$. Note that if $1 \preccurlyeq p_1$ or $1 \preccurlyeq p_2$, then because $p =_T q$ we have $1 \preccurlyeq q_1$ or $1 \preccurlyeq q_2$. Moreover, note that it cannot be the case that p_1 or p_2 equals 0, since then p would not be in T normal form.
Since $q = \sum_{i=1}^{n} a_i.q_i$ (+1) (where $a_i \not\equiv a_j$ when $i \neq j$ and for any i, q_i is in T normal form), there cannot exist a summand q_3 such that $I(q_3) = a$, where $a \in \mathcal{A}$ or $a \in \mathcal{B}$. Furthermore, it cannot be the case that $I(q_3) = c$, such that $c \notin \mathcal{A}$ and $c \notin \mathcal{B}$, since then c would be a trace of q, but c would not be a trace of p, which would contradict $p =_{CT} q$. Hence, $q = q_1 + q_2$.
We have $T(p) = T(p_1) + T(p_2)$ and $T(q) = T(q_1) + T(q_2)$ and by $p =_T q$, $T(p_1) + T(p_2) = T(q_1) + T(q_2)$. We know that $I(p_1) = I(q_1) = \mathcal{A}$, $I(p_2) = I(q_2) = \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B} = \varnothing$. Therefore, all traces

starting with an action $b \in \mathcal{B}$ (or in case $\mathcal{B} = \varnothing$, then the summand 1) can be removed from the equation $T(p_1) + T(p_2) = T(q_1) + T(q_2)$, resulting in $T(p_1) = T(q_1)$. Similarly, all traces starting with action $a \in \mathcal{A}$ (or in case $\mathcal{A} = \varnothing$, then the summand 1) can be removed, resulting in $T(p_2) = T(q_2)$. By similar reasoning $T_1(p_1) = T_1(q_1)$ and $T_1(p_2) = T_1(q_2)$ is obtained. From this we conclude that $p_1 =_T q_1$ and $p_2 =_T q_2$. □

Lemma 3.3.13. *If p, q are in T normal form then $p =_T q \Leftrightarrow p \leftrightarrows q$.*

Proof. ($\Rightarrow$)
The fact that $p =_T q$ implies $p \leftrightarrows q$ is proved with induction on $\#(p) + \#(q)$, which is the sum of the number of symbols in p and q.
Suppose that $p =_T q$ and that $p' =_T q'$ implies $p' \leftrightarrows q'$, for all p', q' such that $\#(p') + \#(q') < \#(p) + \#(q)$ (IH).
$p \leftrightarrows q$ is derived with a case distinction on the structure of p:

- If $p \equiv 0$, then since $p =_T q$, also $q = 0$ and clearly $p = 0 \leftrightarrows 0 = q$.
- If $p \equiv 1$, then since $p =_T q$, also $q = 1$ and clearly $p = 1 \leftrightarrows 1 = q$.
- If $p \equiv a.p'$, then by Lemma 3.3.11 also $q = a.q'$ such that $p' =_T q'$. Since $\#(p') + \#(q') < \#(p) + \#(q)$, it holds by IH that $p' \leftrightarrows q'$. Therefore, by Lemma 3.2.2 $p \leftrightarrows q$.
- If $p \equiv p_1 + p_2$ then by Lemma 3.3.12 also $q = q_1 + q_2$ such that $p_1 =_T q_1$ and $p_2 =_{CT} q_2$. Since $\#(p_1) + \#(q_1) < \#(p) + \#(q)$ and $\#(p_2) + \#(q_2) < \#(p) + \#(q)$, it follows by IH that $p_1 \leftrightarrows q_1$ and $p_2 \leftrightarrows q_2$. Therefore, by Lemma 3.2.3 $p \leftrightarrows q$.

($\Leftarrow$)
From $B \subseteq T$ it follows that $p \leftrightarrows q$ implies $p =_T q$. □

Theorem 3.3.14. *$BSP_{Tr}(A)$ is ground-complete for the algebra of closed BSP(A)-terms modulo trace semantics.*

Proof. By Lemma 3.3.10 for every BSP(A)-term there is an equivalent term in T normal form and by Lemma 3.3.13 two terms in T normal form are trace equivalent iff they are bisimilar. From this and from the fact that BSP(A) is ground-complete for the algebra of closed BSP(A)-terms modulo bisimilarity (Theorem 3.6.4) it follows that $\mathrm{BSP}_{Tr}(\mathrm{A})$ is ground-complete for the algebra of closed BSP(A)-terms modulo trace semantics. □

3.4 Ready simulation semantics for BSP

In BCCSP we use the set of initial actions $I(p)$ of a process p to reason about the initial observations of process p. In BSP we also want to be able to reason about the initial observations of process p, which includes successful termination in addition to actions. For this reason, we introduce the notion of $menu(p)$. Let $menu(p) \subseteq A \cup \{1\}$ be the *menu* of p, i.e. its set of outgoing actions including 1 if p has a termination option.
We give a definition of ready simulation semantics for BSP(A) and we prove soundness and ground-completeness.

3.4.1 Definition

The following definition is given in [10] for ready simulation semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.4.1 (Ready Simulation Semantics (RS))**.** A **ready simulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.

2. If pRq then $I(p) = I(q)$.

Two processes p and q are **ready simulation equivalent**, notation $p =_{RS} q$, if there exists a ready simulation R with pRq and a ready simulation S with qSp.

In [10] a testing scenario is given, which states that ready simulation semantics can be explained with the following *ready trace machine* with undo-button.

"The ready trace machine contains as its interface to the outside world not only the display of the trace machine as described in Section 3.3.1, but also a switch for each action $a \in A$. By means of these switches the observer may determine which actions are free and which are blocked. This situation may be changed any time during a run of the process. It is also equipped with a lamp for each action $a \in Act$. Each time the process idles, the lamps of all actions the process is ready to engage in are lit. All these actions are blocked by the observer, otherwise the process wouldn't idle. Now the observer can see which actions could be released in order to let the process proceed. During the execution of an action no lamps are lit. Furthermore, the machine has an undo-button. It is assumed that all intermediate states that are passed through during a run of a process are stored in a memory inside the black box. Pressing the undo-button causes the machine to shift one state backwards. In the initial state pressing the button has no effect."

"A ready trace of a process consists of a sequence of members and subsets of A, the actions representing information obtained from the display, and the sets of actions representing information obtained from the lights. The information about the free and blocked actions is redundant. An observation now consists of a ready trace, enriched with undo-actions. Such observations can easily be translated into finite ready simulation formulas."

To distinguish between successful and unsuccessful termination, we would change the trace machine (Section 3.3.1) to display a 1 in case the process is terminating, we would create a switch for 1 and we would create a lamp for 1 which would be lit if the process can terminate. To incorporate this, the definition of ready simulation semantics for BSP(A) is changed to the following [2].

Definition 3.4.2 (Ready Simulation Semantics (RS) - Termination)**.** Let $menu(p)$ be the *menu* of p, as defined in Table 3.1. A **ready simulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.
2. If pRq then $menu(p) = menu(q)$.

Two processes p and q are **ready simulation equivalent**, notation $p =_{RS} q$, if there exists a ready simulation R with pRq and a ready simulation S with qSp satisfying conditions 1 and 2.

Definition 3.4.2 differs from Definition 3.4.1 because, in condition 2, $menu(p)$ takes termination into account. This is done to make a distinction between processes that can terminate and processes that cannot terminate, for example the processes $a.0$ and $a.1$.

3.4.2 Soundness

To compute the *menu* of a process, the idea of $menu(p)$ is axiomatised. We do this by defining the unary operator *menu* that calculates the set of initial observations (actions or successful termination) of a process, which is expressed as a process again. The axiomatisation of *menu* can be found in Table 3.1.

$menu(0)$	=	0	M1
$menu(1)$	=	1	M2
$menu(a.x)$	=	$a.0$	M3
$menu(x+y)$	=	$menu(x) + menu(y)$	M4

Table 3.1: Axiomatisation of *menu*

Lemma 3.4.3. *For all TSP(A)-terms p and q, $menu(p)$ is provably equal to $menu(q)$ (i.e. using axioms M1-M4 from Table 3.1) if and only if for all $m \in A \cup \{1\}$ we have $m \in menu(p) \Leftrightarrow m \in menu(q)$.*

Proof. Straightforward. □

For ready simulation (RS) we have axiom Rs (Table 2.2). Since we introduced a different definition for ready simulation equivalence, we need to change this axiom. This is illustrated by the following example.

Example 3.4.4. By Rs we have $p = a.(b.1 + 1 + b.c.1) + a.(b.c.1) \overset{\text{Rs}}{=} a.(b.1 + 1 + b.c.1) = q$. We show that, according to Definition 3.4.2, the processes p and q are not ready simulation equivalent by assuming that p and q are ready simulation equivalent and deriving a contradiction.
Suppose p is ready simulation equivalent to q. Then we know that there is a ready simulation relation R such that $p \ R \ q$ and since $p \xrightarrow{a} b.c.1$ we have $b.c.1 \ R \ b.1 + 1 + b.c.1$. However, since $1 \in menu(b.1 + 1 + b.c.1)$ but $1 \notin menu(b.c.1)$ we have $menu(b.c.1) \neq menu(b.1 + 1 + b.c.1)$. Hence, this contradicts that p and q are ready simulation equivalent as defined in Definition 3.4.1.

If we use the condition $menu(x) = menu(y)$ for axiom Rs, then the processes p and q as defined in Example 3.4.4 are no longer identified. So, we obtain axiom Rs_2:

$$menu(x) = menu(y) \Rightarrow a.(x + y) = a.(x + y) + a.x \qquad Rs_2$$

For convenience, we use $a.(x + y) = a.(x + y) + a.x$ instead of $a.(x + y) = a.(x + y) + a.y$ for Rs_2, which is equivalent by A1.
In [6] it is proved that axiom Rs can be replaced by axiom Rs'.

$$a.(b.x + b.y + z) = a.(b.x + b.y + z) + a.(b.x + z) \qquad Rs'$$

We show that each closed instance of Rs_2 can be derived from $\text{BSP}_{Rs'}$. The proof for this is based on the proof given in [6] for replacing Rs by Rs'.

Lemma 3.4.5. *For all closed BSP(A)-terms p and q, if $menu(p) = menu(q)$, then $a.(p + q) = a.(p + q) + a.p$ can be derived from $BSP_{Rs'}$.*

Proof. Let $menu(p) = menu(q)$, where q is of the form $\sum_{i=1}^{m} b_i.q_i(+1)$. We prove that $a.(p + q) = a.(p + q) + a.p$ can be derived from $\text{BSP}_{Rs'}$ by induction on m.

- If $m = 0$, then there are two cases, either $q = 0$ or $q = 1$. In case $q = 0$, then since $menu(p) = menu(q)$, we derive by axioms M1-M4 that we must also have $p = 0$. Similarly, in case $q = 1$ we must have $p = 1$. In both cases we have $a.(p + q) = a.(p + p) \overset{\text{A3}}{=} a.p \overset{\text{A3}}{=} a.p + a.p \overset{\text{A3}}{=} a.(p + p) + a.p = a.(p + q) + a.p$
- If $m > 0$, then q contains a summand $b_m.q_m$ and since $menu(p) = menu(q)$, p also contains a summand $b_m.p'$. Hence, we have:

$$\begin{aligned} a.(p + q) &\overset{\text{Rs'}}{=} a.(p + q) + a.(p + \sum_{i=1}^{m-1} b_i.q_i) \\ &\overset{\text{IH}}{=} a.(p + q) + a.(p + \sum_{i=1}^{m-1} b_i.q_i) + a.p \\ &\overset{\text{Rs'}}{=} a.(p + q) + a.p \end{aligned}$$

  Note that it is irrelevant for this case whether p and q contain the summand 1. This is because we have $menu(p) = menu(q)$ and so if p contains the summand 1 then also q contains the summand 1 and vice versa and we can remove duplicate summands by A3.

Hence, each closed instance of Rs_2 can be derived from $\text{BSP}_{Rs'}$. □

We prefer Rs' over Rs_2 because Rs' is not a conditional axiom, which allows easier reasoning in the cases where we use this axiom. The axiomatisation of BSP(A) modulo ready simulation semantics consists of axioms A1-A3, A6 (Table 2.2) and axiom Rs'. Axiom Rs' is sound for BSP(A) modulo ready simulation semantics as defined in Definition 3.4.2. This is proved in the following lemma.

Lemma 3.4.6 (Rs'-Ready Simulation-BSP)**.** $a.(b.p + b.q + r) =_{RS} a.(b.p + b.q + r) + a.(b.p + r)$ *for all closed BSP(A)-terms* p, q, r.

Proof. To show that $a.(b.p + b.q + r)$ is ready simulation equivalent to $a.(b.p + b.q + r) + a.(b.p + r)$, we give two ready simulations R and S, with $a.(b.p + b.q + r) + a.(b.p + r)\ R\ a.(b.p + b.q + r)$ and $a.(b.p + b.q + r)\ S\ a.(b.p + b.q + r) + a.(b.p + r)$. We start by giving R.

$$R = \{(a.(b.p + b.q + r) + a.(b.p + r), a.(b.p + b.q + r)) | p, q, r \in T(BSP(A))\} \cup \{(p, p) | p \in T(BSP(A))\} \cup \{(b.p + r, b.p + b.q + r) | p, q, r \in T(BSP(A))\}$$

It is verified that R is a ready simulation relation. Since ready simulation equivalence on pairs of the shape (p, p) is straightforward, this part is omitted.

Condition 1
Pair $(a.(b.p + b.q + r) + a.(b.p + r),\ a.(b.p + b.q + r))$
Suppose $a.(b.p + b.q + r) + a.(b.p + r) \xrightarrow{a} s$. There are two cases:

- If $a.(b.p + b.q + r) \xrightarrow{a} b.p + b.q + r$, $s = b.p + b.q + r$, then clearly $(b.p + b.q + r, b.p + b.q + r) \in R$.
- If $a.(b.p + r) \xrightarrow{a} b.p + r$, $s = b.p + r$, then $a.(b.p + b.q + r) \xrightarrow{a} (b.p + b.q + r)$ and clearly $(b.p + r, b.p + b.q + r) \in R$.

Pair $(b.p + r,\ b.p + b.q + r)$
Suppose $b.p + r \xrightarrow{a} s$. There are two cases:

- If $b.p \xrightarrow{a} s$, $a = b$, $s = p$. Then also $b.p + b.q + r \xrightarrow{b} p$ and clearly $(p, p) \in R$.
- If $r \xrightarrow{a} r'$, $s = r'$. Then also $b.p + b.q + r \xrightarrow{a} r'$ and clearly $(r', r') \in R$.

Condition 2
Pair $(a.(b.p + b.q + r) + a.(b.p + r),\ a.(b.p + b.q + r))$
$menu(a.(b.p + b.q + r) + a.(b.p + r)) = a = menu(a.(b.p + b.q + r))$

Pair $(b.p + r,\ b.p + b.q + r)$
$menu(b.p + r) = menu(r) \cup \{b\} = menu(b.p + b.q + r)$

Condition 3
Pair $(a.(b.p + b.q + r) + a.(b.p + r),\ a.(b.p + b.q + r))$
$a.(b.p + b.q + r) + a.(b.p + r) \not\downarrow$, so this vacuously holds.

Pair $(b.p + r,\ b.p + b.q + r)$
If $b.p + r \downarrow$, since $b.p \not\downarrow$ it must be that $r \downarrow$. Then also $b.p + b.q + r \downarrow$.

This proves that there exists a ready simulation R with $a.(b.p+b.q+r)+a.(b.p+r)\ R\ a.(b.p+b.q+r)$. We define

$$S = \{(a.(b.p + b.q + r), a.(b.p + b.q + r) + a.(b.p + r)) | p, q, r \in T(BSP(A))\} \cup \{(p, p) | p \in T(BSP(A))\}$$

Verifying that S is a ready simulation relation is similar to the verification above and is therefore omitted. Hence, since $a.(b.p + b.q + r) + a.(b.p + r)\ R\ a.(b.p + b.q + r)$ and $a.(b.p + b.q + r)\ S\ a.(b.p + b.q + r) + a.(b.p + r)$, it is concluded that $a.(b.p + b.q + r) + a.(b.p + r) =_{RS} a.(b.p + b.q + r)$. □

Theorem 3.4.7. $BSP_{Rs'}(A)$ *is sound for the algebra of closed BSP(A)-terms modulo ready simulation semantics.*

Proof. From Corollary 3.2.5 soundness of axioms A1-A3 and A6 for BSP(A) modulo ready simulation follows. Furthermore, soundness of Rs' is proved in Lemma 3.4.6. □

3.4.3 Ground-completeness

The following proof is based on the proof for ground-completeness of BCCSP modulo ready simulation in [10]. We do not use the technique presented in Section 3.1 since the following technique yields a simpler proof.

Lemma 3.4.8. *For any BSP(A)-terms p, q it holds that $p =_{RS} q \Rightarrow BSP_{Rs'}(A) \vdash p = q$.*

Proof. We show that for any BSP(A)-terms p, q it holds that

$$p \sqsubseteq_{RS} q \Rightarrow BSP_{Rs'}(A) \vdash q = q + p \tag{3.1}$$

If we have proved Equation 3.1, then by symmetry we also have $q \sqsubseteq_{RS} p \Rightarrow BSP_{Rs'}(A) \vdash p = p + q$. So, when we assume $p =_{RS} q$, then we have $p \sqsubseteq_{RS} q \wedge q \sqsubseteq_{RS} p$. From this, it follows that $BSP_{Rs'}(A) \vdash q = q + p$ and $BSP_{Rs'}(A) \vdash p = p + q$, which can be rewritten as $BSP_{Rs'}(A) \vdash p = p + q \overset{\text{A1}}{=} q + p = q$. So, if we prove Equation 3.1, the lemma follows immediately.
By axioms A1-A3, A6 we have $\text{BSP}_{Rs'}(\text{A}) \vdash q = q + p$ iff $\text{BSP}_{Rs'}(\text{A}) \vdash q = q + p^*$ for every summand p^* of p. We prove Equation 3.1 with structural induction on p. We assume $p \sqsubseteq_{RS} q$ and that Equation 3.1 has been proved for all pairs of smaller BSP(A)-terms p', q' (IH).
So, suppose $p \sqsubseteq_{RS} q$ and let $p^* \trianglelefteq p$. Since p is a BSP(A)-term, the summand p^* of p can be of the form $a.p'$, 0 or 1.

- If $p^* \equiv 0$, then by A6 we have $\text{BSP}_{Rs'}(\text{A}) \vdash q = q + 0 = q + p^*$.
- If $p^* \equiv 1$, then $1 \in \mathit{menu}(p)$. Since $p \sqsubseteq_{RS} q$, we must have that $1 \in \mathit{menu}(q)$. Hence, by the axiomatisation of *menu* as given in Table 3.1, we have $1 \trianglelefteq q$ and so $\text{BSP}_{Rs'}(\text{A}) \vdash q = q + 1 = q + p^*$.
- If $p^* \equiv a.p'$, then $p \xrightarrow{a} p'$. So there exists a q' such that $q \xrightarrow{a} q'$ and $p' \sqsubseteq_{RS} q'$. Now $\mathit{menu}(p') = \mathit{menu}(q')$ and by Lemma 3.4.3 we have $\text{BSP}_{Rs'}(\text{A}) \vdash \mathit{menu}(p') = \mathit{menu}(q')$. By induction $\text{BSP}_{Rs'}(\text{A}) \vdash q' = q' + p'$, so $\text{BSP}_{Rs'}(\text{A}) \vdash a.q' \overset{\text{IH}}{=} a.(q' + p') \overset{\text{A1}}{=} a.(p' + q') \overset{\text{Rs}_2}{=} a.(p' + q') + a.p' \overset{\text{A1}}{=} a.(q' + p') + a.p' \overset{\text{IH}}{=} a.q' + a.p'$. Note that we can use axiom Rs_2 by Lemma 3.4.5. Furthermore, $a.q' \trianglelefteq q$, so $\text{BSP}_{Rs'}(\text{A}) \vdash q = q + a.q' = q + a.q' + a.p' = q + a.p' = q + p^*$.

From the case distinction above, it follows that $\text{BSP}_{Rs'}(\text{A}) \vdash q = q + p$. □

Theorem 3.4.9. *$BSP_{Rs'}(A)$ is ground-complete for the algebra of closed BSP(A)-terms modulo ready simulation semantics.*

Proof. This follows directly from Lemma 3.4.8. □

3.5 Definitions and normal forms of semantics for BSP

Definitions of the semantics $o \in \mathcal{O} \setminus \{T, RS, B\}$ for BSP(A) are given below. These definitions already appear in [2] as well as ground-complete axiomatisations for BSP modulo o-semantics. To give a complete overview of all semantics, we repeat these definitions here. Moreover, in [2] no ground-completeness proofs are given for BSP modulo o-semantics. Therefore, we present normal forms for the semantics $\{CT, F, R, FT, RT\}$ such that the technique described in Section 3.1 can be further applied to prove soundness and ground-completeness for BSP(A) modulo these semantics. To prove soundness and ground-completeness for BSP(A) modulo the semantics $\{S, CS\}$ and to prove soundness for BSP(A) modulo PW semantics we can use a similar technique as applied in Section 3.4. For the ground-completeness proof of BSP(A) modulo PW we can use a similar proof as given for BCCSP(A) modulo PW in [10].

3.5.1 Completed trace semantics

The following definition is given in [10] for completed trace semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.1 (Completed Trace Semantics (CT)). $\sigma \in A$ is a **complete trace** of a process p if there is a process q such that $p \xrightarrow{\sigma} q$ and $I(q) = \varnothing$. Let $CT(p)$ denote the set of complete traces of p. Two processes p and q are **completed trace equivalent**, notation $p =_{CT} q$, if $T(p) = T(q)$ and $CT(p) = CT(q)$.

To distinguish between successful and unsuccessful termination, we change the definition of completed trace semantics for BSP(A) to the following. This is essentially the same as the definition given in [2], but the notation is slightly different. We use the following definition because it corresponds more closely to Definition 3.5.1.

Definition 3.5.2 (Completed Trace Semantics (CT) - Termination). $\sigma \in A$ is a **trace** of a process p if there is a process q such that $p \xrightarrow{\sigma} q$. Let $T(p)$ denote the set of traces of p. $\sigma \in A$ is a **1-trace** of a process p if there is a process q such that $p \xrightarrow{\sigma} q$ and $q\downarrow$. Let $T_1(p)$ denote the set of 1-traces of p. $\sigma \in A$ is a **complete 0-trace** of a process p if there is a process q such that $p \xrightarrow{\sigma} q$, $menu(q) = \varnothing$. Let $CT_0(p)$ denote the set of complete 0-traces of p.
Two processes p and q are **completed trace equivalent**, notation $p =_{CT} q$ if $T(p) = T(q)$, $T_1(p) = T_1(q)$ and $CT_0(p) = CT_0(q)$.

Example 3.5.3. The processes $a.1$ and $a.1 + a.0$ are distinguished by the complete 0-trace a.

For completed trace semantics (CT) we have axiom Ct (Table 2.2), to which we now refer as Ct_1. Since we need to take termination into account, we add axiom Ct_2 [2]:

$$a.(b.x + u) + a.1 = a.(b.x + u + 1) \qquad Ct_2$$

Note that this corresponds to axiom Ct_1 with $c.y + v$ substituted by 1.

The axiomatisation of BSP(A) modulo completed trace semantics consists of axioms A1-A3, A6, Ct_1 (Table 2.2) and Ct_2.
We introduce the notion of CT normal form, to be able to apply the technique presented in Section 3.1. Intuitively, terms in CT normal form are BSP(A)-terms to which axioms Ct_1 and Ct_2 cannot be applied from left to right at any depth.

Definition 3.5.4. Let $t = \sum_{i=1}^{n} a_i.t_i$ (+1), modulo A1 and A2, be a BSP(A)-term, where $a_i \in A$. If $n = 0$ and t does not contain the summand 1 then $t = 0$. In all other cases, the constant 0 is removed using A6. We say that t is in **CT normal form** if for all a_i, t_i, it holds that there does not exist a $j \neq i$, where $1 \leq i, j \leq n$, such that $a_j \equiv a_i \ \wedge \ t_j = (c.r + v) \ \wedge \ t_i = (b.s + u)$, or such that $a_j \equiv a_i \ \wedge \ t_j = 1 \ \wedge \ t_i = (b.s + u)$ for some terms u, v, r, s and actions $b, c \in A$, and if we recursively have that t_i (at any depth of t) is in CT normal form.

3.5.2 Readiness semantics

The following definition is given in [10] for readiness semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.5 (Readiness Semantics (R)). A pair (σ, B) with $B \subseteq A$ is a **ready pair** of process p if there is a process p' such that $p \xrightarrow{\sigma} p'$ with $I(p') = B$. Let $R(p)$ denote the set of ready pairs of p. Two processes p and q are **ready equivalent**, notation $p =_R q$, if $R(p) = R(q)$.

To distinguish between successful and unsuccessful termination, the definition of readiness semantics for BSP(A) is changed to the following [2].

Definition 3.5.6 (Readiness Semantics (R) - Termination)**.** [2] Let p be a process such that $p \xrightarrow{\sigma} p'$. The pair $(\sigma, menu(p'))$ is a **ready pair** of p. The ready set of p is the set of all ready pairs of p. Two processes p and q are **ready equivalent**, notation $p =_R q$ if they have the same ready set.

Example 3.5.7. The processes $a.1$ and $a.1 + a.0$ are distinguished by the ready pair $(a, 0)$. [2]

For readiness semantics (R) we have axiom Re (Table 2.2). The axiomatisation of BSP(A) modulo readiness semantics consists of axioms A1-A3, A6 and Re (Table 2.2).
We introduce the notion of R normal form, to be able to apply the technique presented in Section 3.1. Intuitively, terms in R normal form are BSP(A)-terms to which axiom Re cannot be applied from right to left at any depth.

Definition 3.5.8. Let $t = \sum_{i=1}^{n} a_i.t_i$ (+1), modulo A1 and A2, be a BSP(A)-term, where $a_i \in A$. If $n = 0$ and t does not contain the summand 1 then $t = 0$. In all other cases, the constant 0 is removed using A6. We say that t is in **R normal form** if for all a_i, t_i, it holds that there does not exist a $j \neq i$, where $1 \leq i, j \leq n$, such that $a_j \equiv a_i \ \wedge \ t_j = (b.w + v) \ \wedge \ t_i = (b.s + b.r + u) \ \wedge \ w = r$, for some terms u, v, w, r, s and action $b \in A$, and if we recursively have that t_i (at any depth of t) is in R normal form.

3.5.3 Failures semantics

The following definition is given in [10] for failures semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.9 (Failures Semantics (F))**.** A pair (σ, B) with $B \subseteq A$ is a **failure pair** of process p if there is a process p' such that $p \xrightarrow{\sigma} p'$ with $I(p') \cap B = \varnothing$. Let $F(p)$ denote the set of failure pairs of p. Two processes p and q are **failures equivalent**, notation $p =_F q$, if $F(p) = F(q)$.

To distinguish between successful and unsuccessful termination, the definition of failures semantics for BSP(A) is changed to the following [2].

Definition 3.5.10 (Failures Semantics (F) - Termination)**.** [2] Let p be a process such that $p \xrightarrow{\sigma} p'$. For each $X \subseteq (A \cup \{1\}) \backslash menu(p')$, pair $[\sigma, X]$ is a **failure pair** of p. The failure set of p is the set of all failure pairs of p. Two processes p and q are **failures equivalent**, notation $p =_F q$ if they have the same failure set.

Example 3.5.11. The processes $a.1$ and $a.1 + a.0$ are distinguished by the failure pairs $[a, X]$ with $1 \in X$. [2]

For failures semantics (F) we have axioms Fa_1 and Fa_2 (Table 2.2). The axiomatisation of TSP(A) modulo failures semantics consists of axioms A1-A3, A6, Fa_1 and Fa_2 (Table 2.2).
We introduce the notion of F normal form, to be able to apply the technique presented in Section 3.1. Intuitively, terms in F normal form are BSP(A)-terms to which axioms Fa_1 and Fa_2 cannot be applied from right to left at any depth.

Definition 3.5.12. Let $t = \sum_{i=1}^{n} a_i.t_i$ (+1), modulo A1 and A2, be a BSP(A)-term, where $a_i \in A$. If $n = 0$ and t does not contain the summand 1 then $t = 0$. In all other cases, the constant 0 is removed using A6. We say that t is in **F normal form** if for all a_i, t_i, it holds that there does not exist a $j \neq i$, where $1 \leq i, j \leq n$, such that $a_j \equiv a_i \ \wedge \ t_j = (b.w + v) \ \wedge \ t_i = (b.s + b.r + u) \ \wedge \ w = r$ and there do not exist $j, k \neq i$, where $j \neq k$ and $1 \leq i, j, k \leq n$, such that $a_j \equiv a_i \ \wedge \ a_k \equiv a_i \ \wedge \ t_i = r \ \wedge \ t_j = (s + o) \ \wedge \ t_k = (u + v) \ \wedge \ r = s \ \wedge \ o = u$, for some terms u, v, w, r, s, o and action $b \in A$, and if we recursively have that t_i (at any depth of t) is in F normal form.

3.5.4 Ready trace semantics

The following definition is given in [10] for ready trace semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.13 (Ready Trace Semantics (RT))**.** A sequence $B_0a_1B_1 \dots a_kB_k$ with $k \geq 0$ and $B_0, \dots, B_k \subseteq A$ is a **ready trace** of a process s_0 if there is a sequence of transitions $s_0 \xrightarrow{a_1} \dots \xrightarrow{a_k} s_k$ with $I(s_i) = B_i$ for $i = 0, \dots, k$. Let $RT(p)$ denote the set of ready traces of p. Two processes p and q are **ready trace equivalent**, notation $p =_{RT} q$, if $RT(p) = RT(q)$.

To distinguish between successful and unsuccessful termination, the definition of ready trace semantics for BSP(A) is changed to the following [2].

Definition 3.5.14 (Ready Trace Semantics (RT) - Termination)**.** [2] A **ready trace** of process p is a trace σ, together with, for every process p' that σ passes through $menu(p')$. The ready trace set of p is the set of all ready traces of p. Two processes p and q are **ready trace equivalent**, notation $p =_{RT} q$ if they have the same ready trace set.

For ready trace semantics (RT) we have axiom Rt (Table 2.2). For BSP(A) the condition of this axiom changes to $menu(x) = menu(y)$, since we do not consider the processes $a.1 + a.0$ and $a.1$ to be ready trace equivalent. So axiom Rt becomes:

$$menu(x) = menu(y) \Rightarrow a.x + a.y = a.(x + y) \qquad Rt$$

The axiomatisation of BSP(A) modulo ready trace semantics consists of axioms A1-A3, A6 (Table 2.2) and Rt.
We introduce the notion of RT normal form, to be able to apply the technique presented in Section 3.1. Intuitively, terms in RT normal form are BSP(A)-terms to which axiom Rt cannot be applied from left to right at any depth.

Definition 3.5.15. Let $t = \sum_{i=1}^{n} a_i.t_i$ (+1), modulo A1 and A2, be a BSP(A)-term, where $a_i \in A$. If $n = 0$ and t does not contain the summand 1 then $t = 0$. In all other cases, the constant 0 is removed using A6. We say that t is in **RT normal form** if for all a_i, t_i, it holds that there does not exist a $j \neq i$, where $1 \leq i, j \leq n$, such that $a_j \equiv a_i \wedge menu(t_j) = menu(t_i)$, and if we recursively have that t_i (at any depth of t) is in RT normal form.

3.5.5 Failure trace semantics

The following definition is given in [10] for failure trace semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.16 (Failure Trace Semantics (FT))**.** A sequence $B_0a_1B_1 \dots a_kB_k$ with $k \geq 0$ and $B_0, \dots, B_k \subseteq A$ is a **failure trace** of a process s_0 if there is a sequence of transitions $s_0 \xrightarrow{a_1} \dots \xrightarrow{a_k} s_k$ with $I(s_i) \cap B_i = \varnothing$ for $i = 0, \dots, k$. Let $FT(p)$ denote the set of failure traces of p. Two processes p and q are **failure trace equivalent**, notation $p =_{FT} q$, if $FT(p) = FT(q)$.

To distinguish between successful and unsuccessful termination, the definition of failure trace semantics for BSP(A) is changed to the following [2].

Definition 3.5.17 (Failure Trace Semantics (FT) - Termination)**.** [2] A **failure trace** of process p is a trace σ, together with, for every process p' that σ passes through, a set $X \subseteq A \cup \{\downarrow\}$ disjoint from $menu(p')$, i.e. $X \cap menu(p') = \varnothing$. The failure trace set of p is the set of all failure traces of p. Two processes p and q are **failure trace equivalent**, notation $p =_{FT} q$ if they have the same failure trace set.

For failure trace semantics (FT) we have axioms Ft_1 and Ft_2 (Table 2.2). Using similar reasoning as in Section 3.5.4, the condition of axiom Ft_2 changes to $menu(x) = menu(y)$. So axiom Ft_2 becomes:

$$menu(x) = menu(y) \Rightarrow a.x + a.y = a.(x + y) \qquad Ft_2$$

The axiomatisation of BSP(A) modulo failure trace semantics consists of axioms A1-A3 and A6, Ft_1 (Table 2.2) and Ft_2.
We introduce the notion of FT normal form, to be able to apply the technique presented in Section 3.1. Intuitively, terms in FT normal form are BSP(A)-terms to which axiom Ft_1 cannot be applied from right to left at any depth and to which axiom FT_2 cannot be applied from left to right at any depth.

Definition 3.5.18. Let $t = \sum_{i=1}^{n} a_i.t_i$ (+1), modulo A1 and A2, be a BSP(A)-term, where $a_i \in A$. If $n = 0$ and t does not contain the summand 1 then $t = 0$. In all other cases, the constant 0 is removed using A6. We say that t is in **FT normal form** if for all a_i, t_i, it holds that there does not exist a $j \neq i$, where $1 \leq i, j \leq n$, such that $a_j \equiv a_i \ \wedge \ menu(t_j) = menu(t_i)$ and there do not exist $j, k \neq i$, where $j \neq k$ and $1 \leq i, j, k \leq n$, such that $a_j \equiv a_i \wedge a_k \equiv a_i \wedge t_i = r \wedge t_j = s \wedge t_k = (u+v) \wedge r = u \wedge s = v$, for some terms u, v, r, s, and if we recursively have that t_i (at any depth of t) is in FT normal form.

3.5.6 Simulation semantics

The following definition is given in [10] for simulation semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.19 (Simulation Semantics (S))**.** A **simulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.

Process p **can be simulated by** q, notation $p \overset{\subseteq}{\to} q$, if there is a simulation R with pRq. Processes p and q are **similar**, notation $p \rightleftarrows q$, if $p \overset{\subseteq}{\to} q$ and $q \overset{\subseteq}{\to} p$.

To distinguish between successful and unsuccessful termination, the definition of simulation semantics for BSP(A) is changed to the following [2].

Definition 3.5.20 (Simulation Semantics (S) - Termination)**.** A **simulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.
2. If pRq and $p \downarrow$, then $q \downarrow$.

Process p **can be simulated by** q, notation $p \overset{\subseteq}{\to} q$, if there is a simulation R with pRq satisfying condition 1 and 2. Processes p and q are **similar**, notation $p \rightleftarrows q$ if $p \overset{\subseteq}{\to} q$ and $q \overset{\subseteq}{\to} p$.

For simulation semantics (S) we have axiom Si (Table 2.2). The axiomatisation of BSP(A) modulo simulation semantics consists of axioms A1-A3, A6 and Si (Table 2.2).

3.5.7 Completed simulation semantics

The following definition is given in [10] for completed simulation semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.21 (Completed Simulation Semantics (CS))**.** A **complete simulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.
2. If pRq then $I(p) = \varnothing \Leftrightarrow I(q) = \varnothing$.

Two processes p and q are **completed simulation equivalent**, notation $p =_{CS} q$, if there exists a complete simulation R with pRq and a complete simulation S with qSp.

To distinguish between successful and unsuccessful termination, the definition of completed simulation semantics for BSP(A) is changed to the following [2].

Definition 3.5.22 (Completed Simulation Semantics (CS) - Termination)**.** A **complete simulation** is a binary relation R on processes, satisfying, for all $a \in A$:

1. If pRq and $p \xrightarrow{a} p'$, then $\exists q' : q \xrightarrow{a} q'$ and $p'Rq'$.
2. If pRq then $menu(p)\backslash\{1\} = \varnothing \Leftrightarrow menu(q)\backslash\{1\} = \varnothing$.
3. If pRq and $p\downarrow$, then $q\downarrow$.

Two processes p and q are **completed simulation equivalent**, notation $p =_{CS} q$ if there exist complete simulations R and S with pRq and qSp satisfying conditions 1-3.

For completed simulation semantics (CS) we have axiom Cs (Table 2.2), to which we now refer as Cs_1. Since we need to take termination into account, we add axiom Cs_2 [2]:

$$a.(x+1) = a.(x+1) + a.1 \qquad Cs_2$$

Note that this corresponds to axiom Cs_1 with $b.y + z$ substituted by 1.

The axiomatisation of BSP(A) modulo completed simulation semantics consists of axioms A1-A3, A6, Cs_1 (Table 2.2) and Cs_2 .

3.5.8 Possible worlds semantics

The following definition is given in [10] for possible worlds semantics. It does not make a distinction between successful and unsuccessful termination.

Definition 3.5.23 (Possible Worlds Semantics (PW))**.** The set D of deterministic processes is the largest set such that for each $p \in D$ and $a \in I(p)$ there is exactly one process q such that $p \xrightarrow{a} q$. Moreover, whenever we have that $p \xrightarrow{a} q$, then $q \in D$. A process p is a **possible world** of a process q if p is deterministic and pRq for some ready simulation R as defined in Definition 3.4.1. Let $PW(q)$ denote the class of possible worlds of q. Two processes q and r are **possible worlds equivalent**, notation $q =_{PW} r$, if $PW(q) = PW(r)$.

To distinguish between successful and unsuccessful termination, the definition of possible worlds semantics for BSP(A) is changed to the following [2].

Definition 3.5.24 (Possible Worlds Semantics (PW) - Termination)**.** The set D of deterministic processes is the largest set such that for each $p \in D$ and $a \in I(p)$ there is exactly one process q such that $p \xrightarrow{a} q$. Moreover, whenever we have that $p \xrightarrow{a} q$, then $q \in D$. A process p is a **possible world** of a process q if p is deterministic and pRq for some ready simulation R as defined in Definition 3.4.2. Let $PW(q)$ denote the class of possible worlds of q. Two processes q and r are **possible worlds equivalent**, notation $q =_{PW} r$ if $PW(q) = PW(r)$.

For possible worlds semantics (PW) we have axiom Pw (Table 2.2). The axiomatisation of BSP(A) modulo possible worlds semantics consists of axioms A1-A3, A6 and Pw (Table 2.2).

3.6 Axiomatisations for TSP

We consider TSP, which contains successful termination as well as sequential composition. A theory that only adds sequential composition, such as TSP without termination or BPA [5] is not considered. The reason for this is that when using TSP without termination then the constant 0 is still present and terms such as $a.0 \cdot b.0$ would not be logical. BPA is not considered, because it deviates too much from BCCSP. For example, it introduces $\surd$ to denote successful termination [4].
In this section we show that we can reuse the results of BSP(A) by proving that we can eliminate sequential composition for closed terms when we have the axioms of TSP(A). We prove that TSP(A) is sound and ground-complete modulo bisimilarity. Furthermore, we give an overview of sound and ground-complete axiomatisations for TSP.

3.6.1 Axioms A1-A10

The axiomatisation of TSP(A) modulo bisimilarity consists of axioms A1-A10 in Table 2.2 and Table 2.3. We show that for every closed TSP(A)-term there exists a closed BSP(A)-term, since the axioms for bisimilarity, which is the finest equivalence in the spectrum, admit an elimination theorem [2]. This can be directly exploited to extend the results for BSP from the previous subsections with sequential composition.

Lemma 3.6.1. *For every closed TSP(A)-term p, there is a closed BSP(A)-term p' such that $TSP(A) \vdash p = p'$.*

Proof. This is proved in [2]. □

We prove that TSP(A), i.e. axioms A1-A10 in Table 2.2 and Table 2.3, is sound and ground-complete modulo bisimilarity.

Theorem 3.6.2. *TSP(A) is sound for the algebra of closed TSP(A)-terms modulo bisimilarity.*

Proof. This is proved in [2]. □

Corollary 3.6.3 follows from Theorem 3.6.2 and from the fact that $\forall_{o \in \mathcal{O}} : B \subseteq o$.

Corollary 3.6.3. *For all $o \in \mathcal{O}$ TSP(A) is sound for the algebra of closed TSP(A)-terms modulo o-semantics.*

Theorem 3.6.4. *TSP(A) is ground-complete for the algebra of closed TSP(A)-terms modulo bisimilarity.*

Proof. This is proved in [2]. □

3.6.2 Overview ground-complete axiomatisations for TSP(A)

For all $o \in \mathcal{O}$ sound, ground-complete axiomatisations for TSP(A) modulo o-semantics are given in Table 3.2. Note that for sound, ground-complete axiomatisations for BSP(A) modulo o-semantics we only have to remove axioms A4, A5, and A7-A10.

Theorem 3.6.5. *For each of the semantics o, two closed TSP(A)-terms p, q are o-equivalent iff they can be proved equal from the axioms marked with "+" in the column for o in Table 3.2. The axioms marked with "v" or "ω" are valid in o-semantics but not needed for the proof.*

Proof. The soundness and ground-completeness proofs of TSP modulo bisimulation semantics are given in Sections 3.6.1 and 3.6.1. The soundness and ground-completeness proofs of BSP modulo trace semantics are given in Sections 3.3.2 and 3.3.3. The soundness and ground-completeness proofs of BSP modulo ready simulation semantics are given in Sections 3.4.2 and 3.4.3. The soundness and ground-completeness proofs of BSP modulo the remaining semantics are not explicitly given, but starting points are given in Section 3.5. □

Axiom name	Axiom	B	RS	PW	RT	FT	R	F	CS	CT	S	T
A1	$x + y = y + x$	+	+	+	+	+	+	+	+	+	+	+
A2	$(x + y) + z = x + (y + z)$	+	+	+	+	+	+	+	+	+	+	+
A3	$x + x = x$	+	+	+	+	+	+	+	+	+	+	+
A4	$(x + y) \cdot z = x \cdot z + y \cdot z$	+	+	+	+	+	+	+	+	+	+	+
A5	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	+	+	+	+	+	+	+	+	+	+	+
A6	$x + 0 = x$	+	+	+	+	+	+	+	+	+	+	+
A7	$0 \cdot x = 0$	+	+	+	+	+	+	+	+	+	+	+
A8	$x \cdot 1 = x$	+	+	+	+	+	+	+	+	+	+	+
A9	$1 \cdot x = x$	+	+	+	+	+	+	+	+	+	+	+
A10	$a.x \cdot y = a.(x \cdot y)$	+	+	+	+	+	+	+	+	+	+	+
Rs’	$a.(b.x + b.y + z) = a.(b.x + b.y + z) + a.(b.x + z)$		+	v	v	v	v				v	v
Pw	$a(bx + by + z) = a(bx + z) + a(by + z)$			+	v	v	v	v		v	v	v
Ft$_2$, Rt	$menu(x) = menu(y) \Rightarrow a.x + a.y = a.(x + y)$				+	+	v	v		v		v
Ft$_1$	$a.x + a.y = a.x + a.y + a.(x + y)$					+		v		v		v
Fa$_2$, Re	$a.(b.x + u) + a.(b.y + v) = a.(b.x + b.y + u) + a.(b.y + v)$						+	+		v		v
Fa$_1$	$a.x + a.(y + z) = a.x + a.(x + y) + a.(y + z)$							+		ω		v
Cs$_2$	$a.(x + 1) = a.(x + 1) + a.1$								+	v	v	v
Cs$_1$	$a.(x + b.y + z) = a.(x + b.y + z) + a.(b.y + z)$								+	v	v	v
Ct$_2$	$a.(b.x + u) + a.1 = a.(b.x + u + 1)$									+		v
Ct$_1$	$a.(b.x + u) + a.(c.y + v) = a.(b.x + c.y + u + v)$									+		v
Si	$a.(x + y) = a.(x + y) + a.y$										+	v
Tr	$a.x + a.y = a.(x + y)$											+
	$menu(0) = 0$	+	+	+	+	+	+	+	+	+	+	+
	$menu(1) = 1$	+	+	+	+	+	+	+	+	+	+	+
	$menu(a.x) = a.0$	+	+	+	+	+	+	+	+	+	+	+
	$menu(x + y) = menu(x) + menu(y)$	+	+	+	+	+	+	+	+	+	+	+

Table 3.2: Ground-complete axiomatisations for TSP(A) modulo the semantics

Chapter 4

ω-Completeness

For any process theory X such that X $\in \{BCCSP(A), BSP(A), TSP(A)\}$, an X-term can contain variables. We fix an enumeration of the variables: $x_1, x_2, \ldots$. A substitution σ maps variables x_i, where $i \geq 1$, to X-terms. The application of a substitution σ on a term t is obtained by replacing all occurrences of variables x_i in t with $\sigma(x_i)$. We denote the set of open X-terms by $\mathbb{T}$(X).

For open terms, the same operational rules for BSP and TSP apply as given in Figure 2.4 and Figure 2.6. For open terms t and u and for all $o \in \mathcal{O}$, we write $t \sqsubseteq_o u$ if for all ground substitutions σ it holds that $\sigma(t) \sqsubseteq_o \sigma(u)$.

Definition 4.0.1. [11] For any process theory X and $o \in \mathcal{O}$, the process theory X is **ω-complete** modulo o-semantics iff for all open X-terms p and q:

for all closed substitutions σ: $\quad$ X $\vdash \sigma(p) = \sigma(q) \Leftrightarrow$ X $\vdash p = q$.

Definition 4.0.1 indicates that a process theory is ω-complete if and only if an equation of arbitrary open terms is derivable when all closed-terms equations obtainable from it through substitutions are derivable [2]. For all $o \in \mathcal{O}$, an ω-complete process theory that is ground-complete modulo o-semantics is a **finite basis** for o-semantics.

In [7] it is shown for BCCSP modulo the semantics in van Glabbeek's spectrum whether they are ω-complete, either by giving a finite sound and ground-complete axiomatisation that is ω-complete or by proving that such a finite basis does not exist. Table 4.1 presents an overview for o-semantics, where $o \in \mathcal{O}$, with a + indicating that a finite basis exists and a – indicating that a finite basis does not exist. This overview distinguishes three categories according to the cardinality of A, namely $|A| = 1$, $1 < |A| < \infty$ and $|A| = \infty$.

	$\lvert A\rvert = 1$	$1 < \lvert A\rvert < \infty$	$\lvert A\rvert = \infty$
Bisimulation	+	+	+
Ready simulation	+	-	+
Possible worlds	+	-	+
Ready traces	+	-	-
Failure traces	+	-	+
Readiness	+	-	+
Failures	+	+	+
Completed simulation	+	-	-
Completed traces	+	+	+
Simulation	+	-	+
Traces	+	+	+

Table 4.1: The existence of finite bases for BCCSP modulo o-semantics [7]

In this section we consider ω-completeness of BSP and TSP. We give finite bases for BSP and TSP when there are no actions. We prove that for BSP modulo ready simulation semantics with infinitely many actions a finite basis exists, while there does not exist a finite basis when the number of actions is finite. Furthermore, we prove that TSP modulo bisimilarity with at least one action is ω-complete and we observe that for TSP modulo ready simulation semantics, we cannot reuse any of the previous results, i.e. the proof for BSP modulo ready simulation semantics or the proof for TSP modulo bisimilarity. Finally, we show that the infinite families of equations given in [7] for BCCSP modulo completed simulation and failure trace semantics can be derived from a sound and finite collection of equations over TSP(A).

4.1 BSP

If A is empty we add axiom B1 and prove that for all $o \in \mathcal{O}$ BSP_{B1} is ω-complete modulo o-semantics. After this, we prove that if A is finite, then BSP(A) modulo ready simulation semantics does not have a finite basis. Finally, we prove that BSP_{Rs}(A) is ω-complete modulo ready simulation semantics if A is infinite using Groote's [11] technique of inverted substitutions.

4.1.1 All semantics with $|A| = 0$

We add axiom B1, presented in Table 4.2 below, and we prove that for all $o \in \mathcal{O}$ BSP_{B1} is ω-complete modulo o-semantics if A is empty.

$x + 1$	=	1	B1

Table 4.2: Axiom B1

Lemma 4.1.1. *Axiom B1 is sound for BSP modulo bisimulation semantics when* $|A| = 0$.

Proof. The soundness proof can be found in Appendix A.1. □

Corollary 4.1.2 follows from Lemma 4.1.1 and from the fact that $\forall_{o \in \mathcal{O}} : B \subseteq o$.

Corollary 4.1.2. *For all* $o \in \mathcal{O}$ *axiom B1 is sound for BSP modulo o-semantics when* $|A| = 0$.

We define 0-1-BSP normal forms and prove that for every term there is an equivalent term in 0-1-BSP normal form. This allows simpler reasoning in the ω-completeness proof.

Definition 4.1.3. A term $t \in \mathbb{T}(\mathrm{BSP})$ is in *0-1-BSP normal form* if modulo A1-A3 $t = 0$, $t = 1$ or $t = \sum_{i \in I} x_i$, where I is a subset of the set of natural numbers ≥ 1. Intuitively, t is a term to which axiom A3 cannot be applied from left to right.

Note that by the fixed enumeration of variables $x_1, x_2, \ldots$, we have that all x_i in Definition 4.1.3 are distinct.

Lemma 4.1.4. *If* $|A| = 0$ *then for any term* t *there exists a term* t' *in 0-1-BSP normal form such that BSP(A)* $\vdash t = t'$.

Proof. The proof is by structural induction on t.

- $t \equiv 0$ is in 0-1-BSP normal form by definition.
- $t \equiv 1$ is in 0-1-BSP normal form by definition.
- $t \equiv x_i$ for some $i \geq 1$ is in 0-1-BSP normal form by definition.
- $t \equiv t_1 + t_2$ and suppose that t_1 and t_2 are in 0-1-BSP normal form (IH). There are three cases:

- If $t_1 \equiv 0$, then by A6 $t = t_2$.
- If $t_1 \equiv 1$, then by B1 $t = 1$.
- If $t_1 \equiv \sum_{i \in I} x_i$ (modulo A1, A2) then there are three cases:
  - $t_2 \equiv 0$, then by A6 $t = t_1$.
  - $t_2 \equiv 1$, then by B1 $t = 1$.
  - $t_2 \equiv \sum_{j \in J} x_j$ (modulo A1, A2), then we take $K = I \cup J$ and we remove duplicates from the set K by A3 (and A1, A2). We obtain $t = \sum_{k \in K} x_k$, which is in 0-1-BSP normal form.

□

The following lemma is useful for the ω-completeness proof.

Lemma 4.1.5. *Let $t_1, t_2 \in \mathbb{T}(BSP)$. If for each summand $u \in \mathbb{T}(BSP)$ we have $u \trianglelefteq t_1 \Leftrightarrow u \trianglelefteq t_2$ then $BSP \vdash t_1 = t_2$.*

Proof. Straightforward. □

Theorem 4.1.6. *For all $o \in \mathcal{O}$, if $|A| = 0$ then BSP_{B1} is ω-complete modulo o-semantics.*

Proof. We need to prove that for all terms t, u if $\mathrm{BSP}_{B1} \vdash t^\sigma = u^\sigma$ for all ground substitutions σ, then $\mathrm{BSP}_{B1} \vdash t = u$. We assume, by Lemma 4.1.4, that t, u are in 0-1-BSP normal form and we assume that $t^\sigma = u^\sigma$ for all ground substitutions σ.

To prove that $t = u$ it suffices to show that $u' \trianglelefteq u \Rightarrow u' \trianglelefteq t$, since then by a symmetric argument it follows that $t' \trianglelefteq t \Rightarrow t' \trianglelefteq u$ and hence by Lemma 4.1.5 $t = u$. Since u is in 0-1-BSP normal form it can have three different kinds of summands. So, there are three cases:

- Suppose $u' \equiv 0$. Since u is in 0-1-BSP normal form, this can only be the case if u does not contain any other summands, so $u = 0$. We define $\sigma_1(x_i) = 1$ for all $x_i \in V$, where $i \geq 1$. By assumption we have $t^{\sigma_1} = u^{\sigma_1}$ and by soundness of BSP_{B1} modulo o-semantics we also have $t^{\sigma_1} =_o u^{\sigma_1}$. Then $u^{\sigma_1} = 0$, and by $u^{\sigma_1} =_o t^{\sigma_1}$ we must also have $t^{\sigma_1} =_o 0$. We show that $t^{\sigma_1} =_o 0$ is only the case when $t = 0$. Since t is in 0-1-BSP normal form it can have three different forms.
  - If $t = 0$, then $t^{\sigma_1} = 0 =_o 0$.
  - If $t = 1$, then $t^{\sigma_1} = 1 \neq_o 0$.
  - If $t = \sum_{i \in I} x_i$, then $t^{\sigma_1} = \sum_{i \in I} 1 \overset{\mathrm{A3}}{=} 1 \neq_o 0$.

  Hence $t = 0$ and $0 \trianglelefteq t$.
- Suppose $u' \equiv 1$. Since u is in 0-1-BSP normal form, this can only be the case if u does not contain any other summands, so $u = 1$. We define $\sigma_0(x_i) = 0$ for all $x_i \in V$, where $i \geq 1$. By assumption we have $t^{\sigma_0} = u^{\sigma_0}$ and by soundness of BSP_{B1} modulo o-semantics we also have $t^{\sigma_0} =_o u^{\sigma_0}$. Then $u^{\sigma_0} = 1$, and by $u^{\sigma_0} =_o t^{\sigma_0}$ we must also have $t^{\sigma_0} =_o 1$. We show that $t^{\sigma_0} =_o 1$ is only the case when $t = 1$. Since t is in 0-1-BSP normal form it can have three different forms.
  - If $t = 0$, then $t^{\sigma_0} = 0 \neq_o 1$.
  - If $t = 1$, then $t^{\sigma_0} = 1 =_o 1$.
  - If $t = \sum_{i \in I} x_i$, then $t^{\sigma_0} = \sum_{i \in I} 0 \overset{\mathrm{A3}}{=} 0 \neq_o 1$.

  Hence $t = 1$ and $1 \trianglelefteq t$.

- Suppose $u' \equiv x_i$. We define $\sigma_2(x_i) = 1$ and we define $\sigma_2(x_j) = 0$ for $j \neq i$. By assumption we have $t^{\sigma_2} = u^{\sigma_2}$ and by soundness of BSP_{B1} modulo o-semantics we also have $t^{\sigma_2} =_o u^{\sigma_2}$.
  We have $(u')^{\sigma_2} = 1$ and hence $1 \preccurlyeq u^{\sigma_2}$. Since $t^{\sigma_2} =_o u^{\sigma_2}$ and $1 \preccurlyeq u^{\sigma_2}$, which means $1 \in menu(u^{\sigma_2})$ and $u^{\sigma_2} \downarrow$, we must also have $1 \in menu(t^{\sigma_2})$ or $t^{\sigma_2} \downarrow$ (depending on the semantics) and hence $1 \preccurlyeq t^{\sigma_2}$.
  This summand 1 of t^{σ_2} can originate from two types of summands of t:
  - The summand 1 of t. We prove that this is not possible for this situation. We assume, towards a contradiction, that t contains the summand 1. Since t is in 0-1-BSP normal form this means that $t = 1$.
    We now show that this results in a contradiction, by reversing the argument. We have $1 \preccurlyeq t$ and define $\sigma_3(x_i) = 0$ for all $x_i \in V$, where $i \geq 1$. By assumption we have $t^{\sigma_3} = u^{\sigma_3}$ and by soundness of BSP_{B1} modulo o-semantics we also have $t^{\sigma_3} =_o u^{\sigma_3}$. Using the same reasoning as before, we have $1 \preccurlyeq t^{\sigma_3}$, so $1 \in menu(t^{\sigma_3})$ and $t^{\sigma_3} \downarrow$ and by $t^{\sigma_3} =_o u^{\sigma_3}$ we must also have $1 \in menu(u^{\sigma_3})$ or $u^{\sigma_3} \downarrow$ (depending on the semantics).
    Since $x_i \preccurlyeq u$ and u is in 0-1-BSP normal form, we know that u is of the form $\sum_{i \in I} x_i$. We have $u^{\sigma_3} = \sum_{i \in I} 0 \overset{\mathrm{A3}}{=} 0$. Hence, $1 \notin menu(u^{\sigma_3})$ and $u^{\sigma_3} \not\downarrow$ and we have reached a contradiction, which means that the assumption that t contains the summand 1 is incorrect.
  - The summand x_i of t. Note that no other variables result in summand 1 of t^{σ_2}, since their substitution equals 0. Hence $x_i \preccurlyeq t$.

□

4.1.2 Ready simulation semantics with $1 \leq |A| < \infty$

In [7] it is stated that ready simulation semantics coincides with completed trace semantics for BCCSP when $|A| = 1$. However, this does not hold for BSP. This follows from the example below.

Example 4.1.7. Consider the processes $p = a.(a.1) + a.(a.a.0 + 1)$ and $q = a.(a.1 + a.a.0 + 1)$. It can easily be seen that these processes are completed trace equivalent by A6 and Ct. However p and q are not ready simulation equivalent, since we have $p \xrightarrow{a} a.1$ and this can only be simulated by $q \xrightarrow{a} a.1 + a.a.0 + 1$. So there is a ready simulation relation R such that $a.1 \; R \; (a.1 + a.a.0 + 1)$, but $menu(a.1) \neq menu(a.1 + a.a.0 + 1)$, which contradicts the definition of ready simulation semantics.

Using similar arguments we can conclude that ready simulation semantics does not coincide with any other semantics for BSP when $|A| = 1$.

We consider $1 \leq |A| < \infty$, and we show that there does not exist a finite basis for BSP modulo ready simulation semantics. In [8] it is proved that BCCSP modulo ready simulation semantics does not have a finite basis when $|A| < \infty$. They provide the infinite family of inequations:

$$a^n.x \sqsubseteq a^n.0 + \sum_{b \in A} a^n.(x + b.0) \tag{4.1}$$

which are sound for BCCSP modulo $\sqsubseteq_{RS}$. The idea is that either x cannot perform any action, in which case $a^n.x$ is ready simulated by $a^n.0$, or x can perform some action b, in which case $a^n.x$ is ready simulated by $a^n.(x + b.0)$ [8].
They prove that Equation 4.1 cannot be derived from a finite collection E of inequations over BCCSP(A) that is sound modulo $\sqsubseteq_{RS}$ if n is larger than the depth of any term in E. After this they provide the infinite family of equations:

$$a^n.x + a^n.0 + \sum_{b \in A} a^n.(x + b.0) = a^n.0 + \sum_{b \in A} a^n.(x + b.0) \tag{4.2}$$

They prove that Equation 4.2 cannot be derived from a finite collection E of equations over BCCSP(A) that is sound modulo $=_{RS}$ if n is larger than the depth of any term in E.

To prove that BSP modulo ready simulation semantics does not have a finite basis, we use the same approach. Since in BSP processes have a termination option, we change Equation 4.1 to the following:

$$a^n.x \sqsubseteq a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0) \tag{4.3}$$

In case $|A| = 1$, we remove the summations and substitute b by a. The idea of Equation 4.3 is that either x cannot perform an action and cannot terminate, in which case $a^n.x$ is ready simulated by $a^n.0$, or x cannot perform an action and can terminate, in which case $a^n.x$ is ready simulated by $a^n.1$ or x can perform some action b, in which case $a^n.x$ is ready simulated by $a^n.(x + b.0)$.

We prove that Equation 4.3 cannot be derived from a finite collection E of inequations over BSP(A) that is sound modulo $\sqsubseteq_{RS}$ if n is larger than the depth of any term in E. The proof for this is an extension of the proof given in [8] of Equation 4.1 for BCCSP when $|A| < \infty$.

Definition 4.1.8. Let $t \xrightarrow{a_1 \dots a_k} t'$ (with $k \geq 0$) denote the trace $t = t_0 \xrightarrow{a_1} t_1 \xrightarrow{a_2} \dots \xrightarrow{a_k} t_k = t'$. If t' cannot perform any transitions (meaning that each summand of t' is a variable, 0 or 1), then $t \xrightarrow{a_1 \dots a_k} t'$ is called a **termination trace** of t.

Lemma 4.1.9. *Let $t \sqsubseteq_{RS} u$. If $t \xrightarrow{a_1 \dots a_k} x + t'$, then $u \xrightarrow{a_1 \dots a_k} x + u'$.*

Proof. This is proved in Lemma 1 in [8]. □

The following lemma is based on Lemma 5 in [8]. We present additional cases, namely we prove that $\rho(t) \not\sqsubseteq_{RS} a^{n-1}.1$ and we consider $t_1 =_{RS} 1$ and $t_2 =_{RS} 1$, to take the constant 1 into account.

Lemma 4.1.10. *Let $1 \leq |A| < \infty$. If $a.t \sqsubseteq_{RS} a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0)$ then $a.t$ is ready similar to $a^n.0$, $a^n.1$, $a^n.x$ or $a^n.(x + b.0)$ for some $b \in A$.*

Proof. By assumption $a.t \sqsubseteq_{RS} a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0)$. Then every termination trace of t is of the form $t \xrightarrow{a^{n-1}} t'$ or $t \xrightarrow{a^{n-1}.b} t'$. Moreover, by Lemma 4.1.9, t can only contain the variable x and x cannot occur at depth n in t. It follows that for every trace of t such that $t \xrightarrow{a^{n-1}} t'$, t' is ready similar to either 0, 1, x or $x + b_0.0$ for some $b_0 \in A$. Suppose, towards a contradiction, that $t \xrightarrow{a^{n-1}} t_1$ and $t \xrightarrow{a^{n-1}} t_2$ with $t_1 \not\sqsubseteq_{RS} t_2$. In each of the possible cases (modulo symmetry) we give a closed substitution ρ with $\rho(a.t) \not\sqsubseteq_{RS} \rho(a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0))$.

- $t_1 =_{RS} 0$ and $t_2 =_{RS} x$ or $t_2 = 1$ or $t_2 =_{RS} x + b_0.0$ for some $b_0 \in A$. Let $\rho(x) \not=_{RS} 0$.
  - $\rho(t) \not\sqsubseteq_{RS} a^{n-1}.0$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_2) \not=_{RS} 0$.
  - $\rho(t) \not\sqsubseteq_{RS} a^{n-1}.1$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_1) \not=_{RS} 1$.
  - $\rho(t) \not\sqsubseteq_{RS} a^{n-1}.\rho(x + b.0)$ for each $b \in A$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_1) =_{RS} 0$ and $\rho(x + b.0) \not=_{RS} 0$.
- $t_1 =_{RS} x$ and $t_2 = 1$ or $t_2 =_{RS} x + b_0.0$ for some $b_0 \in A$. Let $\rho(x) = 0$.
  - $\rho(t) \not\sqsubseteq_{RS} a^{n-1}.0$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_2) \not=_{RS} 0$.
  - $\rho(t) \not\sqsubseteq_{RS} a^{n-1}.1$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_1) \not=_{RS} 1$.
  - $\rho(t) \not\sqsubseteq_{RS} a^{n-1}.\rho(x + b.0)$ for each $b \in A$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_1) =_{RS} 0$ and $\rho(x + b.0) \not=_{RS} 0$.
- $t_1 =_{RS} x + b_0.0$ and $t_2 = 1$ for some $b_0 \in A$. Let $\rho(x) = 0$.

- $\rho(t) \not\in_{RS} a^{n-1}.0$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_2) \not=_{RS} 0$.
- $\rho(t) \not\in_{RS} a^{n-1}.1$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_1) \not=_{RS} 1$.
- $\rho(t) \not\in_{RS} a^{n-1}.\rho(x+b.0)$ for each $b \in A$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_2) =_{RS} 1$ and $\rho(x+b.0) \not=_{RS} 1$.

- $t_1 =_{RS} x + b_1.0$ and $t_2 =_{RS} x + b_2.0$ for some $b_1, b_2 \in A$ with $b_1 \neq b_2$. Let $\rho(x) = 0$.
  - $\rho(t) \not\in_{RS} a^{n-1}.0$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_1) \not=_{RS} 0$.
  - $\rho(t) \not\in_{RS} a^{n-1}.1$ because $\rho(t) \xrightarrow{a^{n-1}} \rho(t_1) \not=_{RS} 1$.
  - $\rho(t) \not\in_{RS} a^{n-1}.\rho(x + b.0)$ for each $b \in A$ because $b \neq b_i$ for $i = 1$ or $i = 2$, so that $\rho(t) \xrightarrow{a^{n-1}} \rho(t_i) =_{RS} b_i.0$ and $\rho(x + b.0) \not=_{RS} b_i.0$.

We conclude that the cases above all contradict $a.t \sqsubseteq a^n.0 + a^n.1 + \sum_{b\in A} a^n.(x+b.0)$. Hence, it must be the case that for each pair of traces $t \xrightarrow{a^{n-1}} t_1$ and $t \xrightarrow{a^{n-1}} t_2$, $t_1 =_{RS} t_2$. Moreover, by Lemma 4.1.9, t does not contain variables at depths smaller than $n-1$. This implies the lemma. □

Lemma 4.1.11. *Let $1 \leq |A| < \infty$ and let E be a finite collection of inequations over BSP(A) that is sound modulo $\sqsubseteq_{RS}$. Let n be larger than the depth of any term in E. Assume that:*

- $E \vdash t \sqsubseteq u$.
- $u \sqsubseteq_{RS} a^n.0 + a^n.1 + \sum_{b\in A} a^n.(x + b.0)$.
- *t has a summand ready similar to $a^n.x$.*

Then u has a summand ready similar to $a^n.x$.

Proof. Lemma 6 of [8] is similar to this one and is proved. Only minor changes, such as adding the summand $a^n.1$ in case we have $a^n.0 + \sum_{b\in A} a^n.(x + b.0)$, to the proof in [8] are required and therefore this is omitted here. Note that another change compared to Lemma 6 of [8] is that we require Lemma 4.1.10 to prove this lemma (instead of Lemma 5 of [8]). □

Lemma 4.1.12. *Let $1 \leq |A| < \infty$ and let E be a finite collection of equations over BSP(A) that is sound modulo $\sqsubseteq_{RS}$. Let n be larger than the depth of any term in E. Then from E we cannot derive the inequation*

$$a^n.x \sqsubseteq a^n.0 + a^n.1 + \sum_{b\in A} a^n.(x + b.0) \tag{4.4}$$

Proof. $a^n.0 + a^n.1 + \sum_{b\in A} a^n.(x + b.0)$ does not contain a summand ready similar to $a^n.x$. So according to Lemma 4.1.11, the inequation $a^n.x \sqsubseteq a^n.0 + a^n.1 + \sum_{b\in A} a^n.(x + b.0)$, which is sound modulo $\sqsubseteq_{RS}$, cannot be derived form E. □

Theorem 4.1.13. *Let $1 \leq |A| < \infty$. Then $\sqsubseteq_{RS}$ is not finitely based over BSP(A).*

Proof. By Lemma 4.1.12, no finite collection of inequations over BSP(A) that is sound modulo $\sqsubseteq_{RS}$ proves all inequations that are sound modulo $\sqsubseteq_{RS}$. □

We now prove that if $1 \leq |A| < \infty$, then BSP(A) modulo $=_{RS}$ does not have a finite basis.

Lemma 4.1.14. *Let $1 \leq |A| < \infty$ and let E be a finite collection of equations over BSP(A) that is sound modulo $=_{RS}$. Let n be larger than the depth of any term in E. Assume that:*

- $E \vdash t = u$.
- $u =_{RS} a^n.0 + a^n.1 + \sum_{b\in A} a^n.(x + b.0)$.

- *t has a summand ready similar to $a^n.x$.*

Then u has a summand ready similar to $a^n.x$.

Proof. The proof is similar to the proof of Lemma 4.1.11 and is therefore omitted here. □

Lemma 4.1.15. *Let $1 \leq |A| < \infty$ and let E be a finite collection of equations over BSP(A) that is sound modulo $=_{RS}$. Let n be larger than the depth of any term in E. Then from E we cannot derive the equation*

$$a^n.x + a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0) = a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0) \tag{4.5}$$

Proof. $a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0)$ does not contain a summand ready similar to $a^n.x$. So according to Lemma 4.1.14, the equation $a^n.x + a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0) = a^n.0 + a^n.1 + \sum_{b \in A} a^n.(x + b.0)$, which is sound modulo $=_{RS}$, cannot be derived form E. □

Theorem 4.1.16. *Let $1 \leq |A| < \infty$. Then $=_{RS}$ is not finitely based over BSP(A).*

Proof. By Lemma 4.1.15, no finite collection of equations over BSP(A) that is sound modulo $=_{RS}$ proves all equations that are sound modulo $=_{RS}$. □

4.1.3 Ready simulation semantics with $|A| = \infty$

In [11], Groote introduces a technique to prove ω-completeness. The proof is based on inverted substitutions and works as described below.

Theorem 4.1.17. Consider an axiomatisation E. For each equation $t = u$ of which all closed instances can be derived from E, one must define a closed substitution ρ and a mapping $R : \mathrm{T(BSP)} \to \mathbb{T}\mathrm{(BSP)}$ such that:

1. $E \vdash R(\rho(t)) = t$ and $E \vdash R(\rho(u)) = u$.
2. For each function symbol f (with arity n), $E \cup \{p_i = q_i, R(p_i) = R(q_i) | i = 1, \ldots, n\} \vdash R(f(p_1, \ldots, p_n)) = R(f(q_1, \ldots, q_n))$ for all closed terms $p_1, \ldots, p_n$, $q_1, \ldots, q_n$.
3. For each $v = w \in E$ and closed substitution σ, $E \vdash R(\sigma(v)) = R(\sigma(w))$.

If these conditions are satisfied, then E is ω-complete [11].

We prove that $\mathrm{BSP}_{Rs'}$ is ω-complete modulo ready simulation semantics if A is infinite using Groote's technique of inverted substitutions. The proof below is based on the proof given in [8] for BCCSP modulo ready simulation semantics. It can be adapted for BSP modulo bisimulation by removing the check of axiom Rs'.

Theorem 4.1.18. *If $|A| = \infty$ then $BSP_{Rs'}$ is ω-complete modulo ready simulation semantics.*

Proof. Consider two terms $t, t' \in \mathbb{T}\mathrm{(BSP)}$. Since $|A| = \infty$, we define $\rho : V \to T\mathrm{(BSP)}$ by:

$$\rho(x) = a_x.1$$

where a_x is a unique action for $x \in V$ that occurs in neither t nor t'. Define $R : T\mathrm{(BSP)} \to \mathbb{T}\mathrm{(BSP)}$ as follows:

$R(0) = 0$
$R(1) = 1$
$R(t + u) = R(t) + R(u)$
$R(a.t) = a.R(t)$ if $a \not\equiv a_x$ for all $x \in V$
$R(a_x.t) = x$

Condition 1
Since t, t' do not contain actions a_x, it follows that $R(\rho(t)) = t$ and $R(\rho(t')) = t'$.

Condition 2
Assume BSP(A) $\vdash u_i = u_i'$ and BSP(A) $\vdash R(u_i) = R(u_i')$ for $u_i, u_i' \in T(\text{BSP(A)})$ and $i = 1, 2$.
Consider the operator $\_ + \_$. BSP(A) $\vdash R(u_1 + u_2) = R(u_1) + R(u_2) = R(u_1') + R(u_2') = R(u_1' + u_2')$
Consider the operator $a.\_$. We distinguish two cases:

- Suppose $a \neq a_y$ for all $y \in V$. Then BSP(A) $\vdash R(a.u_1) = a.R(u_1) = a.R(u_1') = R(a.u_1')$
- Suppose $a = a_y$ for some $y \in V$. Then $R(a_y.u_1) = y = R(a_y.u_1')$

Condition 3
For axioms A1-A3 and A6 (Table 3.2) the proof is trivial, because they do not contain actions. To illustrate this, we check A1. Axioms A2, A3 and A6 can be dealt with similarly. Let $\sigma : V \to T(\text{BSP})$ be a substitution, then:

$$\text{BSP(A)} \vdash R(\sigma(x + y)) = R(\sigma(x)) + R(\sigma(y)) = R(\sigma(y)) + R(\sigma(x)) = R(\sigma(y + x))$$

We check axiom Rs' (Table 3.2). We distinguish three cases.

- Suppose $a = a_y$ for some $y \in V$. Then $R(a_y.(b.\sigma(x_1) + b.\sigma(x_2) + \sigma(x_3))) = y = y + y = R(a_y.(b.\sigma(x_1) + b.\sigma(x_2) + \sigma(x_3)) + a_y.(b.\sigma(x_1) + \sigma(x_3)))$.
- Suppose $a \neq a_y$ for all $y \in V$ and $b = b_z$ for some $z \in V$. Then $R(a.(b_z.\sigma(x_1) + b_z.\sigma(x_2) + \sigma(x_3))) = a.(z + z + R(\sigma(x_3))) = a.(z + z + R(\sigma(x_3))) + a.(z + R(\sigma(x_3))) = R(a.(b_z.\sigma(x_1) + b_z.\sigma(x_2) + \sigma(x_3)) + a.(b_x.\sigma(x_1) + \sigma(x_3)))$.
- Suppose $a \neq a_y$ for all $y \in V$ and $b \neq b_z$ for all $z \in V$. Then $R(a.(b.\sigma(x_1) + b.\sigma(x_2) + \sigma(x_3))) = a.(b.R(\sigma(x_1))+b.R(\sigma(x_2))+R(\sigma(x_3))) = a.(b.R(\sigma(x_1))+b.R(\sigma(x_2))+R(\sigma(x_3)))+ a.(b.R(\sigma(x_1)) + R(\sigma(x_3))) = R(a.(b.\sigma(x_1) + b.\sigma(x_2) + \sigma(x_3)) + a.(b.\sigma(x_1) + \sigma(x_3)))$

□

4.2 TSP

We add axioms and prove that, for all $o \in \mathcal{O}$, TSP together with these axioms is ω-complete modulo o-semantics if A is empty. After this, we prove that TSP is ω-complete modulo bisimilarity if A contains at least one element. Furthermore, we show that if $|A| \geq 1$, then for TSP modulo ready simulation semantics we cannot use the proof methods of BSP modulo ready simulation semantics. Finally, we show that the infinite families of equations given in [7], which are sound modulo completed simulation and failure trace semantics but which cannot be derived from a sound and finite collection of equations over BCCSP(A), can be derived from a sound and finite collection of equations over TSP(A).

4.2.1 All semantics with $|A| = 0$

We add axioms B1 (Table 4.2) and B2-B6 (Table 4.3) and we prove that, for all $o \in \mathcal{O}$, TSP_{B1-B6} is ω-complete modulo o-semantics if A is empty.

$x \cdot 0$	=	0	B2
$x \cdot y$	=	$y \cdot x$	B3
$x \cdot x$	=	x	B4
$x + (x \cdot y)$	=	x	B5
$x \cdot (y + z)$	=	$x \cdot y + x \cdot z$	B6

Table 4.3: Axioms B2-B6

Lemma 4.2.1. *Axioms B1-B6 are sound for TSP modulo bisimilarity when* $|A| = 0$.

Proof. Their soundness proofs can be found in Appendix A.1 and Appendix A.2. □

Corollary 4.2.2 follows from Lemma 4.2.1 and from the fact that $\forall_{o \in \mathcal{O}} : B \subseteq o$.

Corollary 4.2.2. *For all* $o \in \mathcal{O}$*, axioms B1-B6 are sound for TSP modulo o-semantics when* $|A| = 0$.

We define 0-1 normal forms and prove that for every term there is an equivalent term in 0-1 normal form. This allows simpler reasoning in the ω-completeness proof. This proof resembles the ω-completeness proof for BSP modulo o-semantics when $|A| = 0$, but it is slightly more complex due to the addition of sequential composition.

Definition 4.2.3. A term $t \in \mathbb{T}(\text{TSP})$ is in *0-1 normal form* if modulo A1-A3 $t = 0$, $t = 1$ or

$$t = \sum_{i \in I} x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i}$$

where I is a subset of the set of natural numbers ≥ 1. Moreover, for all $j, k \in I$ it holds that $\{x_{j,0}, x_{j,1}, \ldots, x_{j,n_j}\} \neq \{x_{k,0}, x_{k,1}, \ldots, x_{k,n_k}\}$. Furthermore, for all $j, k \in I$ it holds that $\{x_{j,0}, x_{j,1}, \ldots, x_{j,n_j}\} \notin \{x_{k,0}, x_{k,1}, \ldots, x_{k,n_k}\}$. Intuitively, t is a term to which axioms A3, B4 and B5 cannot be applied from left to right.

Note that by the fixed enumeration of variables $x_1, x_2, \ldots$, we have that all $x_{i,r}$ in Definition 4.2.3 are distinct $(0 \leq r \leq n_i)$.

Lemma 4.2.4. *If* $A = \varnothing$ *then for any term* t *there exists a term* t' *in 0-1 normal form such that* $\text{TSP}(A) \vdash t = t'$.

Proof. The proof is by structural induction on t.

- $t \equiv 0$ is in 0-1 normal form by definition.
- $t \equiv 1$ is in 0-1 normal form by definition.
- $t \equiv x$ is in 0-1 normal form by definition.
- $t \equiv t_1 + t_2$ and suppose that t_1 and t_2 are in 0-1 normal form (IH). There are three cases:
  - If $t_1 \equiv 0$, then by A6 $t = t_2$.
  - If $t_1 \equiv 1$, then by B1 $t = 1$.
  - If $t_1 \equiv \sum_{i \in I} x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i}$ (modulo A1, A2) then there are three cases:
    - $t_2 \equiv 0$, then by A6 $t = t_1$.
    - $t_2 \equiv 1$, then by B1 $t = 1$.
    - $t_2 \equiv \sum_{j \in J} x_{j,0} \cdot x_{j,1} \cdot \ \ldots \ \cdot x_{j,n_j}$ (modulo A1, A2), then we take $K = I \cup J$ and we obtain $t = \sum_{k \in K} x_{k,0} \cdot x_{k,1} \cdot \ \ldots \ \cdot x_{k,n_k}$. If there exist $k, l \in K$ such that $\{x_{k,0}, \ldots, x_{k,n_k}\} = \{x_{l,0}, \ldots, x_{l,n_l}\}$, then we eliminate the summand $\{x_{k,0}, \ldots, x_{k,n_k}\}$ by A3 (and A1, A2 and B3). If there exist $k, l \in K$ such that $\{x_{k,0}, \ldots, x_{k,n_k}\} \subset \{x_{l,0}, \ldots, x_{l,n_l}\}$, then we eliminate the summand $\{x_{k,0}, \ldots, x_{k,n_k}\}$ by B5 (and A1, A2 and B3). The resulting t is in 0-1 normal form.
- $t \equiv t_1 \cdot t_2$ and suppose that t_1 and t_2 are in ω-B normal form (IH). There are three cases:
  - If $t_1 \equiv 0$, then by A7 $t = 0$.
  - If $t_1 \equiv 1$, then by A9 $t = t_2$.
  - If $t_1 \equiv \sum_{i \in I} x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i}$ (modulo A1, A2), then there are three cases:

* $t_2 \equiv 0$, then by A4 we have $t = \sum_{i\in I} x_{i,0} \cdot x_{i,1} \cdot \ \dots \ \cdot x_{i,n_i} \cdot 0$ and by B2 we have $t = \sum_{i\in I} 0$ and by A6 we have $t = 0$.
* $t_2 \equiv 1$, then by A8 we have $t = t_1$.
* $t_2 \equiv \sum_{j\in J} x_{j,0} \cdot x_{j,1} \cdot \ \dots \ \cdot x_{j,n_j}$ (modulo A1, A2), then by A4 we have $t = \sum_{i\in I} x_{i,0} \cdot x_{i,1} \cdot \ \dots \ \cdot x_{i,n_i} \cdot (\sum_{j\in J} x_{j,0} \cdot x_{j,1} \cdot \ \dots \ \cdot x_{j,n_j})$. We can then apply B6 to obtain $t = \sum_{i\in I} \sum_{j\in J} x_{i,0} \cdot x_{i,1} \cdot \ \dots \ \cdot x_{i,n_i} \cdot x_{j,0} \cdot x_{j,1} \cdot \ \dots \ \cdot x_{j,n_j}$. We can write t as $t = \sum_{k\in K} x_{k,0} \cdot x_{k,1} \cdot \ \dots \ \cdot x_{k,n_k}$. If there exist $0 \leq g, h \leq k_n$ such that $x_{k,g} \equiv x_{k,h}$, then we eliminate $x_{k,h}$ by B4. If there exist $k, l \in K$ such that $\{x_{k,0}, \dots, x_{k,n_k}\} \subset \{x_{l,0}, \dots, x_{l,n_l}\}$, then we eliminate the summand $\{x_{k,0}, \dots, x_{k,n_k}\}$ by B5 (and A1, A2 and B3). The resulting t is in 0-1 normal form.

□

The following lemma is useful for the ω-completeness proof.

Lemma 4.2.5. *Let $t_1, t_2 \in \mathbb{T}(TSP)$. If for each summand $u \in \mathbb{T}(TSP)$ we have $u \preccurlyeq t_1 \Leftrightarrow u \preccurlyeq t_2$ then $TSP \vdash t_1 = t_2$.*

Proof. Straightforward. □

Theorem 4.2.6. *If $|A| = 0$ then for all $o \in \mathcal{O}$ TSP_{B1-B6} is ω-complete modulo o-semantics.*

Proof. We need to prove that for all terms t, u if $\mathrm{TSP}_{B1-B6} \vdash t^\sigma = u^\sigma$ for all ground substitutions σ, then $\mathrm{TSP}_{B1-B6} \vdash t = u$. We assume, by Lemma 4.2.4, that t, u are in 0-1 normal form and we assume that $t^\sigma = u^\sigma$ for all ground substitutions σ.

To prove that $t = u$ it suffices to show that $u' \preccurlyeq u \Rightarrow u' \preccurlyeq t$, since then by a symmetric argument it follows that $t' \preccurlyeq t \Rightarrow t' \preccurlyeq u$ and hence by Lemma 4.2.5 $t = u$. Since u is in 0-1 normal form it can have three different kinds of summands. So, there are three cases:

- Suppose $u' \equiv 0$. Since u is in 0-1 normal form, this can only be the case if u does not contain any other summands, so $u = 0$. We define $\sigma_1(x) = 1$ for all $x \in V$. By assumption we have $t^{\sigma_1} = u^{\sigma_1}$ and by soundness of TSP_{B1-B6} modulo o-semantics we also have $t^{\sigma_1} =_o u^{\sigma_1}$. Then $u^{\sigma_1} = 0$, and by $u^{\sigma_1} =_o t^{\sigma_1}$ we must also have $t^{\sigma_1} =_o 0$. We show that $t^{\sigma_1} =_o 0$ is only the case when $t = 0$. Since t is in 0-1 normal form it can have three different forms.
  - If $t = 0$, then $t^{\sigma_1} = 0 =_o 0$.
  - If $t = 1$, then $t^{\sigma_1} = 1 \neq_o 0$.
  - If $t = \sum_{i\in I} x_{i,0} \cdot x_{i,1} \cdot \ \dots \ \cdot x_{i,n_i}$, then $t^{\sigma_1} = \sum_{i\in I} 1 \cdot 1 \cdot \ \dots \ \cdot 1 \overset{A8}{=} \sum_{i\in I} 1 \overset{A3}{=} 1 \neq_o 0$.

  Hence $t = 0$ and $0 \preccurlyeq t$.
- Suppose $u' \equiv 1$. Since u is in 0-1 normal form, this can only be the case if u does not contain any other summands, so $u = 1$. We define $\sigma_0(x) = 0$ for all $x \in V$. By assumption we have $t^{\sigma_0} = u^{\sigma_0}$ and by soundness of TSP_{B1-B6} modulo o-semantics we also have $t^{\sigma_0} =_o u^{\sigma_0}$. Then $u^{\sigma_0} = 1$, and by $u^{\sigma_0} =_o t^{\sigma_0}$ we must also have $t^{\sigma_0} =_o 1$. We show that $t^{\sigma_0} =_o 1$ is only the case when $t = 1$. Since t is in 0-1 normal form it can have three different forms.
  - If $t = 0$, then $t^{\sigma_0} = 0 \neq_o 1$.
  - If $t = 1$, then $t^{\sigma_0} =_o 1$.
  - If $t = \sum_{i\in I} x_{i,0} \cdot x_{i,1} \cdot \ \dots \ \cdot x_{i,n_i}$, then $t^{\sigma_0} = \sum_{i\in I} 0 \cdot 0 \cdot \ \dots \ \cdot 0 \overset{A7}{=} \sum_{i\in I} 0 \overset{A3}{=} 0 \neq_o 1$.

  Hence $t = 1$ and $1 \preccurlyeq t$.

- Suppose $x_{i,r}$ where $0 \leq r \leq n_i$ are variables such that $u' \equiv x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i}$. We define $\sigma_2(x_{i,r}) = 1$ for $0 \leq r \leq n_i$ and we define $\sigma_2(x_{j,s}) = 0$ for all variables $x_{j,s} \notin \{x_{i,0}, \ldots, x_{i,n_i}\}$. By assumption we have $t^{\sigma_2} = u^{\sigma_2}$ and by soundness of TSP_{B1-B6} modulo o-semantics we also have $t^{\sigma_2} =_o u^{\sigma_2}$.
  We have $(u')^{\sigma_2} = 1 \cdot 1 \cdots 1 \overset{A8}{=} 1$ and hence $1 \preccurlyeq u^{\sigma_2}$. Since $t^{\sigma_2} =_o u^{\sigma_2}$ and $1 \preccurlyeq u^{\sigma_2}$, which means $1 \in menu(u^{\sigma_2})$ and $u^{\sigma_2} \downarrow$, we must also have $1 \in menu(t^{\sigma_2})$ or $t^{\sigma_2} \downarrow$ (depending on the semantics) and hence $1 \preccurlyeq t^{\sigma_2}$.
  This summand 1 of t^{σ_2} can originate from two types of summands of t:

  - The summand 1 of t.
  - A summand consisting of a combination of the variables $x_{i,r}$ in t. It is possible for this combination to consist of a single variable. This summand cannot contain any other variables, since the substitution of all other variables equals 0 and hence the summand would be equal to 0 by A7, B2.

  We split the remainder of the proof according to this case distinction.

  Summand 1 in t
  We want to prove that it is not possible that t contains the summand 1. We assume, towards a contradiction, that t contains the summand 1. Since t is in 0-1 normal form this means that $t = 1$.
  We now show that this results in a contradiction, by reversing the argument. We have $1 \preccurlyeq t$ and define $\sigma_3(x) = 0$ for all $x \in V$. By assumption we have $t^{\sigma_3} = u^{\sigma_3}$ and by soundness of TSP_{B1-B6} modulo o-semantics we also have $t^{\sigma_3} =_o u^{\sigma_3}$.
  Using the same reasoning as before, we have $1 \preccurlyeq t^{\sigma_3}$, so $1 \in menu(t^{\sigma_3})$ and $t^{\sigma_3} \downarrow$ and by $t^{\sigma_3} =_o u^{\sigma_3}$ we must also have $1 \in menu(u^{\sigma_3})$ or $u^{\sigma_3} \downarrow$ (depending on the semantics).
  Since $x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i} \preccurlyeq u$ and u is in 0-1 normal form, we know that u is of the form $\sum_{i \in I} x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i}$. We have $u^{\sigma_3} = \sum_{i \in I} 0 \cdot 0 \cdot \ \ldots \ \cdot 0 \overset{A7}{=} \sum_{i \in I} 0 \overset{A3}{=} 0$. Hence, $1 \notin menu(u^{\sigma_3})$ and $u^{\sigma_3} \not\downarrow$ and we have reached a contradiction, which means that the assumption that t contains the summand 1 is incorrect.

  Summand with combination of variables $x_{i,r}$ in t
  We want to prove that it is not possible that t contains a strict subset of the variables $x_{i,r}$, but that it must contain all variables $x_{i,r}$ and hence that $x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i} \preccurlyeq t$. We assume, towards a contradiction, that t contains a summand that sequentially composes a strict subset of the variables $x_{i,r}$. So t contains a summand s' such that $s' = x_{i,s_0} \cdot x_{i,s_1} \cdot \ \ldots \ \cdot x_{i,s_n}$. Note that since this is a strict subset of the variables $x_{i,r}$ and since the variables $x_{i,r}$ do not contain duplicates, that there must be at least one variable $x_{i,r*}$ where $0 \leq r* \leq n_i$ that does not occur in summand s'.
  We now show that this results in a contradiction, by reversing the argument. We have $s' \preccurlyeq t$ and define $\sigma_4(x_{i,s}) = 1$ where $s \in \{s_0, \ldots s_n\}$ and $\sigma_4(x_{j,v}) = 0$ for all variables $x_{j,v} \notin \{x_{i,s_0}, \ldots, x_{i,s_n}\}$. By assumption we have $t^{\sigma_4} = u^{\sigma_4}$ and by soundness of TSP_{B1-B6} modulo o-semantics we also have $t^{\sigma_4} =_o u^{\sigma_4}$.
  Using the same reasoning as before, we have $1 \preccurlyeq t^{\sigma_4}$, so $1 \in menu(t^{\sigma_4})$ and $t^{\sigma_4} \downarrow$ and by $t^{\sigma_4} =_o u^{\sigma_4}$ we must also have $1 \in menu(u^{\sigma_4})$ or $u^{\sigma_4} \downarrow$ (depending on the semantics) and hence $1 \preccurlyeq u^{\sigma_4}$.
  Note that it is not possible for u to contain the summand 1, since we already know that u is of the form $\sum_{i \in I} x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i}$ and u is in 0-1 normal form. So, the only possibility for u^{σ_4} to terminate is that u contains a summand d' that sequentially composes a (not necessarily strict) subset of the variables $x_{i,s}$. However, this d' cannot be present in u, since otherwise we would have $u' \preccurlyeq u$ and $d' \preccurlyeq u$ where d' is a summand that sequentially composes a strict subset of the variables in u'. If this were the case, then axiom B5 could be applied to u, which would contradict u being in 0-1 normal form.
  Hence, since $1 \notin menu(u^{\sigma_5})$ and $u^{\sigma_5} \not\downarrow$ we have reached a contradiction, which means that the assumption that t contains a summand that sequentially composes a strict subset of the

variables $x_{i,r}$ is incorrect. It follows that $x_{i,0} \cdot x_{i,1} \cdot \ \ldots \ \cdot x_{i,n_i} \preccurlyeq t$.

□

4.2.2 Bisimulation semantics with $|A| \geq 1$

We prove that TSP is ω-complete modulo bisimilarity if A contains at least one element. The proof below is based on the proof given in [9] for PA, which is a process theory that includes parallel composition.

We define ω-B normal forms and prove that for every term there is an equivalent term in ω-B normal form. This allows simpler reasoning in the ω-completeness proof.

Definition 4.2.7. A term $t \in \mathbb{T}(\text{TSP})$ is in *ω-B normal form* if modulo A1, A2

$$t = \sum_{i \in I} a_i.t_i + \sum_{j \in J} x_j \cdot u_j \ (+1)$$

where $a_i \in A$, t_i and u_j are arbitrary terms, and x_j are variables. Term t can have a termination option, denoted by $(+1)$ (duplicate constants 1 are removed using A3). If $I, J = \varnothing$ and t does not contain the summand 1, then $t = 0$.

Lemma 4.2.8. *For any term t there exists a term t' in ω-B normal form such that TSP(A) $\vdash t = t'$*

Proof. The proof is by structural induction on t.

- $t \equiv 0$ is in ω-B normal form by definition.
- $t \equiv 1$ is in ω-B normal form by definition.
- $t \equiv x$, then by A8 $t = x \cdot 1$, which is in ω-B normal form.
- $t \equiv a.t''$ and suppose that t'' is in ω-B normal form (IH). Then $a.t''$ is in ω-B normal form.
- $t \equiv t_1 + t_2$ and suppose that t_1 and t_2 are in ω-B normal form (IH). There are two cases:
  - If $t_1 = 0$ then by A6 $t = t_2$ which is in ω-B normal form.
  - If $t_1 = \sum_{i \in I} a_i.t_i + \sum_{j \in J} x_j \cdot u_j (+1)$, then there are two cases for t_2:
    - If $t_2 = 0$, then by A6 $t = t_1$, which is in ω-B normal form.
    - If $t_2 = \sum_{g \in G} a_g.t_g + \sum_{h \in H} x_h \cdot u_h (+1)$, then by A1-A3

      $$t = \sum_{k \in I \cup G} a_k.t_k + \sum_{l \in J \cup H} x_l \cdot u_l (+1)$$

      which is in ω-B normal form.
- $t \equiv t_1 \cdot t_2$ and suppose that t_1 and t_2 are in ω-B normal form (IH). There are two cases:
  - If $t_1 = 0$, then by A7 $t = 0$ which is in ω-B normal form.
  - If $t_1 = \sum_{i \in I} a_i.t_i + \sum_{j \in J} x_j \cdot u_j (+1)$, then by A4 we have

    $$t = \sum_{i \in I} (a_i.t_i) \cdot t_2 + \sum_{j \in J} (x_j \cdot u_j) \cdot t_2 \ (+1 \cdot t_2)$$

    We can apply A10 to $(a_i.t_i) \cdot t_2$, A5 to $(x_j \cdot u_j) \cdot t_2$ and A9 to $1 \cdot t_2$ (if $t_2 = 0$, then we can further apply A6 to remove this last summand). The resulting t is in ω-B normal form.

□

To prove that TSP is ω-complete modulo bisimilarity when $|A| \geq 1$, we define a closed substitution σ_m for which it holds that if $t^{\sigma_m} = u^{\sigma_m}$ then $t = u$. We define the *depth* and *norm* of terms, which are the longest and shortest path respectively. Substitution σ_m distinguishes variables by giving them the possibility to increase *norm* significantly (by more than *depth*) or to decrease *norm* by 1.

Definition 4.2.9. We define the *depth* of t as follows:

$$\begin{array}{rcll} depth(0) & = & 0 & \\ depth(1) & = & 0 & \\ depth(a.t) & = & depth(t)+1 & \text{where } a \in A \\ depth(x) & = & 4 & \text{where } x \text{ is a variable} \\ depth(t \cdot u) & = & depth(t)+depth(u) & \\ depth(t+u) & = & max\{depth(t), depth(u)\} & \end{array}$$

Definition 4.2.10. We define the *norm* of t as follows:

$$\begin{array}{rcll} norm(0) & = & 0 & \\ norm(1) & = & 0 & \\ norm(a.t) & = & norm(t)+1 & \text{where } a \in A \\ norm(x) & = & 3 & \text{where } x \text{ is a variable} \\ norm(t \cdot u) & = & norm(t)+norm(u) & \\ norm(t+u) & = & min\{norm(t), norm(u)\} & \end{array}$$

We have fixed an enumeration of the variables: $x_1, x_2, \ldots$. Let x_i be a variable and let m be a natural number. We define the substitution σ_m by

$$\sigma_m(x_i) = a.(a.a^{i+m}.1 + a) \cdot a.1$$

where $a^n.1$ denotes n applications of the a-prefix to 1. We want to choose m large compared to the depths already occurring in t and u. With every term t we associate a natural number $depth(t)$ that denotes the maximal depth that occurs in t.
We set $depth(x_i) = 4$ for variable x_i, because after applying substitution σ_m to variable x_i we have $depth(x_i) \geq 4$. This is because the longest path is $aaa^{i+m}a$ and hence $depth(x_i) = 3 + i + m$ where $i \geq 1$.
Similarly, we set $norm(x) = 3$ for variable x because after applying substitution σ_m to variable x, we have $norm(x) = 3$, since the shortest path is aaa.

We prove that *norm* of some term t remains the same after applying substitution σ_m.

Lemma 4.2.11. *For any term t in ω-B normal form, and for σ_m as defined above, it holds that $norm(t) = norm(t^{\sigma_m})$.*

Proof. The proof is by structural induction on t.

- $t \equiv 0$, then $t^{\sigma_m} = 0$ and hence $norm(t) = 0 = norm(t^{\sigma_m})$.
- $t \equiv 1$, then $t^{\sigma_m} = 1$ and hence $norm(t) = 0 = norm(t^{\sigma_m})$.
- $t \equiv x$, then $t^{\sigma_m} = a.(a.a^{i+m}.1 + a) \cdot a.1$ and hence $norm(t) = 3 = norm(t^{\sigma_m})$.
- $t \equiv a.t'$ and suppose that we have $norm(t') = norm((t')^{\sigma_m})$ (IH). Then $t^{\sigma_m} = a.(t')^{\sigma_m}$ and hence $norm(t) = norm(a.t') = norm(t') + 1 \overset{\text{IH}}{=} norm((t')^{\sigma_m}) + 1 = norm(a.(t')^{\sigma_m}) = norm(t^{\sigma_m})$.
- $t \equiv t_1 + t_2$ and suppose that we have $norm(t_1) = norm((t_1)^{\sigma_m})$ and $norm(t_2) = norm((t_2)^{\sigma_m})$ (IH). Then $t^{\sigma_m} = (t_1)^{\sigma_m} + (t_2)^{\sigma_m}$ and hence $norm(t) = norm(t_1 + t_2) = min\{norm(t_1), norm(t_2)\} \overset{\text{IH}}{=} min\{norm((t_1)^{\sigma_m}), norm((t_2)^{\sigma_m})\} =$
$norm((t_1)^{\sigma_m} + (t_2)^{\sigma_m}) = norm(t^{\sigma_m})$.

- $t \equiv t_1 \cdot t_2$ and suppose that we have $norm(t_1) = norm((t_1)^{\sigma_m})$ and $norm(t_2) = norm((t_2)^{\sigma_m})$ (IH). Then $t^{\sigma_m} = (t_1)^{\sigma_m} \cdot (t_2)^{\sigma_m}$ and hence $norm(t) = norm(t_1 \cdot t_2) = norm(t_1) + norm(t_2) \overset{\text{IH}}{=} norm((t_1)^{\sigma_m}) + norm((t_2)^{\sigma_m}) = norm((t_1)^{\sigma_m} \cdot (t_2)^{\sigma_m}) = norm(t^{\sigma_m})$.

□

Lemma 4.2.12. *Let t be a term in ω-B normal form and let σ_m be as defined above where $m \geq depth(t)$ and $i \geq 1$. Term t^{σ_m} cannot do an a-step such that $norm(t^{\sigma_m})$ increases by $i+m-1$.*

Proof. Note that if $t^{\sigma_m} \xrightarrow{a} p$, then t must have a summand t' such that $(t')^{\sigma_m} \xrightarrow{a} p$. Since t is in ω-B normal form, it can have four different kinds of summands t'. The proof is by a case distinction on these summands.

- $t' \equiv 0$, then $(t')^{\sigma_m} = 0 \not\xrightarrow{}$.
- $t' \equiv 1$, then $(t')^{\sigma_m} = 1 \not\xrightarrow{}$.
- $t' \equiv a.t''$, then $(a.t'')^{\sigma_m} = a.(t'')^{\sigma_m} \xrightarrow{a} (t'')^{\sigma_m}$. Since t' is in ω-B normal form, we have by Lemma 4.2.11 $norm(t') = norm((t')^{\sigma_m})$. Hence, $norm((t')^{\sigma_m}) = norm(t') \leq depth(t) \leq m$. Because of this, $norm((t')^{\sigma_m}) = norm(a.(t'')^{\sigma_m}) = 1 + norm((t'')^{\sigma_m})$ can be at most m larger than $norm(t^{\sigma_m})$ and $norm((t'')^{\sigma_m}$ can be at most $m-1$ larger than $norm(t^{\sigma_m})$. So, after one a-step of $(t')^{\sigma_m}$, $norm(t^{\sigma_m})$ can increase by at most $m-1$. Since $i \geq 1$, we have $m-1 < i+m-1$ and we conclude that $t' \equiv a.t''$ cannot do an a-step such that $norm(t^{\sigma_m})$ increases by $i+m-1$.
- $t' \equiv x_j \cdot t'_j$, then $(x_j \cdot t'_j)^{\sigma_m} = a.(a.a^{j+m}.1 + a) \cdot a.1 \cdot (t'_j)^{\sigma_m} \xrightarrow{a} (a.a^{j+m}.1 + a) \cdot a.1 \cdot (t'_j)^{\sigma_m}$. We derive that $t' \equiv x_j \cdot t'_j$ cannot do an a-step such that $norm(t^{\sigma_m})$ increases by $i+m-1$ using similar reasoning as in the previous case.

□

We show, by means of an example, that a specific situation occurs when we use σ_m to substitute a summand starting with a variable and we prove that this situation can only occur for a summand that starts with a variable.

Example 4.2.13. Suppose t is in ω-B normal form, $t = x_i \cdot t^* + t'$ and σ_m is defined as above with $m \geq depth(t)$. Then we have $t^{\sigma_m} = a.(a.a^{i+m}.1 + a) \cdot a.1 \cdot (t^*)^{\sigma_m} + (t')^{\sigma_m}$. So, $t^{\sigma_m} \xrightarrow{a} (a.a^{i+m}.1 + a) \cdot a.1 \cdot (t^*)^{\sigma_m}$ and $norm(t^{\sigma_m})$ becomes $norm(a \cdot a.1 \cdot (t^*)^{\sigma_m}) = 2 + norm((t^*)^{\sigma_m})$ and we have two possibilities:

- $(a.a^{i+m}.1 + a) \cdot a.1 \cdot (t^*)^{\sigma_m} \xrightarrow{a} a^{i+m}.1 \cdot a.1 \cdot (t^*)^{\sigma_m}$, and $norm(t^{\sigma_m})$ becomes $norm(a^{i+m}.1 \cdot a.1 \cdot (t^*)^{\sigma_m}) = i + m + 1 + norm((t^*)^{\sigma_m})$ and hence $norm(t^{\sigma_m})$ increases by $i+m-1$.
- $(a.a^{j+m}.1 + a) \cdot a.1 \cdot (t^*)^{\sigma_m} \xrightarrow{a} a.1 \cdot (t^*)^{\sigma_m}$, and $norm(t^{\sigma_m})$ becomes $norm(a.1 \cdot (t^*)^{\sigma_m}) = 1 + norm((t^*)^{\sigma_m})$ and hence $norm(t^{\sigma_m})$ decreases by 1.

Lemma 4.2.14. *Let σ_m be as defined above and let $m \geq depth(t)$ for term t in ω-B normal form. If it holds that*

> *$t^{\sigma_m} \xrightarrow{a} p$ for some term p such that there are two possibilities for p:*
>
> - *$p \xrightarrow{a} p'$ such that $norm(t^{\sigma_m})$ increases by $i+m-1$.*
> - *$p \xrightarrow{a} p''$ such that $norm(t^{\sigma_m})$ decreases by 1.*

then term t has a summand of the form $x_i \cdot t^*$.

Proof. In Example 4.2.13 it is shown that for term t with summand $x_i \cdot t^*$ the situation above occurs. Since t is in ω-B normal form, t can have four different kinds of summands. We prove that the situation above cannot occur for a summand different from $x_i \cdot t^*$ by a case distinction on the summands t' of t.

- $t' \equiv 0$, then $(t')^{\sigma_m} = 0 \not\rightarrow$.
- $t' \equiv 1$, then $(t')^{\sigma_m} = 1 \not\rightarrow$.
- $t' \equiv a.t''$, then $(a.t'')^{\sigma_m} = a.(t'')^{\sigma_m} \xrightarrow{a} (t'')^{\sigma_m}$. However, by Lemma 4.2.12 we know that $(t'')^{\sigma_m}$ cannot do an a-step such that $norm(t^{\sigma_m})$ increases by $i+m-1$.
- $t' \equiv x_j \cdot t'_j$ where $j \neq i$ then $(x_j \cdot t'_j)^{\sigma_m} = a.(a.a^{j+m}.1+a) \cdot a.1 \cdot (t'_j)^{\sigma_m} \xrightarrow{a} (a.a^{j+m}.1+a) \cdot a.1 \cdot (t'_j)^{\sigma_m}$. However, $(a.a^{j+m}.1 + a) \cdot a.1 \cdot (t'_j)^{\sigma_m}$ cannot do an a-step such that $norm(t^{\sigma_m})$ increases by $i+m-1$. This is because $norm(a.a^{j+m}.1+a) \cdot a.1 \cdot (t'_j)^{\sigma_m} = 2 + norm(t'_j)^{\sigma_m}$ and after an a-step we have $norm(a^{j+m}.1) \cdot a.1 \cdot (t'_j)^{\sigma_m} = j + m + 1 + norm(t'_j)^{\sigma_m}$ or we have $norm(a.1 \cdot (t'_j)^{\sigma_m} = 1 + norm(t'_j)^{\sigma_m}$. Hence, after an a-step $norm(t^{\sigma_m})$ increases by $j + m - 1 \neq i + m - 1$ (since $i \neq j$) or $norm(t^{\sigma_m})$ decreases by 1.

□

The following two lemmas are useful for the ω-completeness proof.

Lemma 4.2.15. *Let σ_m be as defined above. For any term t in ω-B normal form, $t = 0 \Leftrightarrow t^{\sigma_m} = 0$.*

Proof. ($\Rightarrow$)
Suppose $t = 0$. Since $0^{\sigma_m} = 0$ we know that $t^{\sigma_m} = 0$.
($\Leftarrow$)
Suppose $t^{\sigma_m} = 0$. We want to prove that $t = 0$. Since t is in ω-B normal it can have four different kinds of summands t'. We prove by a case distinction on the structure of t' that we can only have $t^{\sigma_m} = 0$ if $t' = 0$ and hence $t = 0$.

- If $t' \equiv 0$, then since t is in ω-B normal form, we have $t = 0$. Then $t^{\sigma_m} = 0$.
- If $t' \equiv a_i.t_i$, then $(t')^{\sigma_m} = a_i.(t_i)^{\sigma_m} \preccurlyeq t^{\sigma_m}$ and hence $t^{\sigma_m} \neq 0$.
- If $t' \equiv x_j \cdot u_j$, then $(t')^{\sigma_m} = a.(a.a^{j+m} + a) \cdot a.1 \cdot (u_j)^{\sigma_m} \preccurlyeq t^{\sigma_m}$ and hence $t^{\sigma_m} \neq 0$.
- If $t' \equiv 1$, then $(t')^{\sigma_m} = 1 \preccurlyeq t^{\sigma_m}$ and hence $t^{\sigma_m} \neq 0$.

Hence, we only have $t^{\sigma_m} = 0$ in case $t = 0$. □

Lemma 4.2.16. *Let σ_m be as defined above. For any term t in ω-B normal form, $1 \preccurlyeq t \Leftrightarrow t^{\sigma_m} \downarrow$.*

Proof. ($\Rightarrow$)
Suppose $1 \preccurlyeq t$. Since $1^{\sigma_m} = 1$ we know that $1 \preccurlyeq t^{\sigma_m}$ and hence $t^{\sigma_m} \downarrow$.
($\Leftarrow$)
Suppose $t^{\sigma_m} \downarrow$. We want to prove that $1 \preccurlyeq t$. Since t is in ω-B normal form, it can have two different forms:

- If $t = 0$, then $t^{\sigma_m} = 0 \not\downarrow$.
- If $t = \sum_{i \in I} a_i.t_i + \sum_{j \in J} x_j \cdot u_j$ $(+1)$, then t can have three different kinds of summands t'. We prove by a case distinction on the structure of t' that we can only have $t^{\sigma_m} \downarrow$ if $t' = 1$ and hence $1 \preccurlyeq t$.
  - If $t' \equiv a_i.t_i$, then $(t')^{\sigma_m} = a_i.(t_i)^{\sigma_m} \not\downarrow$.
  - If $t' \equiv x_j \cdot u_j$, then $(t')^{\sigma_m} = a.(a.a^{j+m} + a) \cdot a.1 \cdot (u_j)^{\sigma_m} \not\downarrow$.
  - If $t' \equiv 1$, then $(t')^{\sigma_m} = 1 \downarrow$.

From this case distinction it follows that t^{σ_m} can only terminate if $1 \preccurlyeq t$. □

Theorem 4.2.17. *If $|A| \geq 1$ then TSP(A) is ω-complete modulo bisimilarity.*

Proof. We need to prove that for all terms t, u if $\mathrm{TSP(A)} \vdash t^\sigma = u^\sigma$ for all ground substitutions σ, then $\mathrm{TSP(A)} \vdash t = u$.
Let $m \geq max\{depth(t), depth(u)\}$. We prove that if $t^{\sigma_m} = u^{\sigma_m}$ then $t = u$. We assume, by Lemma 4.2.8, that t, u are in ω-B normal form and in particular that $t = 0$ or

$$t = \sum_{i \in I} a_i.t_i + \sum_{j \in J} x_j \cdot v_j \ (+1)$$

Assume that for all u' with $depth(u') < depth(u)$ it holds for all t' with $depth(t') < depth(t)$, if $(t')^{\sigma_m} = (u')^{\sigma_m}$ then $t' = u'$. (IH)

Suppose $t^{\sigma_m} = u^{\sigma_m}$. By soundness of TSP(A) modulo bisimilarity we also have $t^{\sigma_m} \leftrightarroweq u^{\sigma_m}$. To prove that $t = u$ it suffices to show that $u' \preccurlyeq u \Rightarrow u' \preccurlyeq t$, since then by a symmetric argument it follows that $t' \preccurlyeq t \Rightarrow t' \preccurlyeq u$ and hence by Lemma 4.2.5 $t = u$. Since u is in ω-B normal form it can have four different kinds of summands. So, there are four cases:

- Suppose $u' \equiv 0$. Since u is in ω-B normal form, this can only be the case if u does not contain any other summands, so $u \equiv 0$. Then $u^{\sigma_m} \equiv 0$ and by $u^{\sigma_m} = t^{\sigma_m}$ also $t^{\sigma_m} = 0$. By Lemma 4.2.15, it follows that $t = 0$ and so $0 \preccurlyeq t$.

- Suppose $u' \equiv 1$. Since $1 \preccurlyeq u^{\sigma_m}$, we have $u^{\sigma_m} \downarrow$ and by $t^{\sigma_m} \leftrightarroweq u^{\sigma_m}$ we must have $t^{\sigma_m} \downarrow$. By Lemma 4.2.16, it follows that $1 \preccurlyeq t$.

- Suppose $a_i \in A$, and u'' is a term such that $u' \equiv a_i.u''$. We know that σ_m does not change the action a_i, so $a_i.(u'')^{\sigma_m} \preccurlyeq u^{\sigma_m}$. By $t^{\sigma_m} \leftrightarroweq u^{\sigma_m}$ we know that $t^{\sigma_m} \xrightarrow{a_i} p$ for some term $p \leftrightarroweq (u'')^{\sigma_m}$. Since t is in ω-B normal form, there are two cases for p:

  - The action a_i can originate from a summand of the form $a_i.t_i$, which means that $p = (t_i)^{\sigma_m}$. So $a_i.t_i \preccurlyeq t$ and it follows by the induction hypothesis and ground-completeness that $t_i = u''$ and hence $a_i.u'' \preccurlyeq t$.
  - If $a_i = a$, the action a can originate from a summand of the form $x_j \cdot v_j$, which means that $p = (a.a^{j+m} + a) \cdot a.1 \cdot (v_j)^{\sigma_m}$. So $x_j \cdot v_j \preccurlyeq t$ and we have the situation described in Lemma 4.2.14. Namely $t^{\sigma_m} \xrightarrow{a} p$ for some p such that:
    - $p \xrightarrow{a} p'$ such that $norm(t^{\sigma_m})$ increases by $j + m - 1$.
    - $p \xrightarrow{a} p''$ such that $norm(t^{\sigma_m})$ decreases by 1.

    We know that $p \leftrightarroweq (u'')^{\sigma_m}$. Hence, we should have $(u'')^{\sigma_m} \xrightarrow{a} q$ such that $norm(u^{\sigma_m})$ increases by $j + m - 1$. However, by Lemma 4.2.12 this cannot be the case and we have reached a contradiction. This means that the assumption that $p = (a.a^{j+m} + a) \cdot a.1 \cdot (v_j)^{\sigma_m}$ is incorrect.

- Suppose x_j is a variable and u'' is a term such that $u' \equiv x_j \cdot u''$. We know that $x_j^{\sigma_m} = a.(a.a^{j+m} + a) \cdot a.1$ and so $a.(a.a^{j+m} + a) \cdot a.1 \cdot (u'')^{\sigma_m} \preccurlyeq u^{\sigma_m}$. By $t^{\sigma_m} \leftrightarroweq u^{\sigma_m}$ we must have $t^{\sigma_m} \xrightarrow{a} p$ such that $p \leftrightarroweq (a.a^{j+m} + a) \cdot a.1 \cdot (u'')^{\sigma_m}$. This corresponds to the situation of Lemma 4.2.14:

  - $p \xrightarrow{a} p'$ such that $norm(t^{\sigma_m})$ increases by $j + m - 1$.
  - $p \xrightarrow{a} p'$ such that $norm(t^{\sigma_m})$ decreases by 1.

  By Lemma 4.2.14 this can only be the case if $x_j \cdot v_j \preccurlyeq t$. By the induction hypothesis we have $v_j = u''$ and hence $x_j \cdot u'' \preccurlyeq t$.

□

4.2.3 Ready simulation semantics with $|A| \geq 1$

We observe that we cannot reuse the proof for TSP modulo bisimilarity with $|A| \geq 1$ for TSP modulo ready simulation semantics. Specifically, issues arise when we try to reuse the second part of the third case in the proof of Theorem 4.2.17. There we use the fact that $p \leftrightarrows (u'')^{\sigma_m}$ to conclude that $(u'')^{\sigma_m} \xrightarrow{a} q$ such that $norm(u^{\sigma_m})$ increases by $j+m+i$. However, if we were to use this for ready simulation semantics, then we would have that there is a ready simulation relation R such that $(u'')^{\sigma_m}\ R\ p$ and hence we cannot conclude that $(u'')^{\sigma_m} \xrightarrow{a} q$ such that $norm(u^{\sigma_m})$ increases by $j+m+i$. Hence, for this case we cannot reason towards a contradiction. So, when reusing this proof, we are not able to prove that if $t^{\sigma_m} = u^{\sigma_m}$ and $a_i.u'' \preccurlyeq u$, then also $a_i.u'' \preccurlyeq t$. Since this is necessary for the ω-completeness proof, we conclude that we cannot reuse the proof for TSP modulo bisimilarity with $|A| \geq 1$ for TSP modulo ready simulation semantics.

We check if we can reuse the proofs for BSP modulo ready simulation semantics with $1 \leq |A| < \infty$ and $|A| = \infty$ for TSP modulo ready simulation semantics. We split the remainder of this section into two parts, $1 \leq |A| < \infty$ and $|A| = \infty$, and show that in both cases we cannot use the same techniques as for BSP modulo ready simulation semantics.

Ready simulation semantics with $1 \leq |A| < \infty$

Using similar reasoning as in Section 4.1.2 we find that ready simulation semantics does not coincide with any other semantics for TSP when $|A| = 1$.
Recall from Section 4.1.2 that we cannot derive Equation 4.5 from a sound and finite collection E of equations over BSP(A) if n is larger than the depth of any term in E.
We prove that we cannot use the same reasoning for TSP modulo ready simulation semantics. Since TSP has the sequential composition operator, we can rewrite Equation 4.5 to:

$$x \cdot y + x \cdot 0 + x \cdot 1 + \sum_{b \in A} x \cdot (y + b.0) = x \cdot 0 + x \cdot 1 + \sum_{b \in A} x \cdot (y + b.0) \tag{4.6}$$

Note that Equation 4.5 is an instance of Equation 4.6. More specifically, we substitute x of Equation 4.6 by $a^n.1$ and y of Equation 4.6 by x and we apply axioms A9 and A10.
We prove that Equation 4.6 is sound in the following lemma. By doing this we find that Equation 4.5 is not an infinite family of equations which is sound for TSP modulo $=_{RS}$ but which cannot be derived, since we know that it can be derived from the axiom in Equation 4.6.

Lemma 4.2.18. *$p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) =_{RS} p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)$ for all closed TSP(A)-terms p, q.*

Proof. To show that $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)$ is ready simulation equivalent to $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)$, we give two ready simulations R and S, with $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)\ R\ p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)$ and $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)\ S\ p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)$. We define:

$$\begin{aligned} R = \{&(p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0),\ p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)) \mid \\ &p, q \in T(TSP(A))\} \cup \{(p,\ p) \mid p \in T(TSP(A))\} \cup \\ &\{(p \cdot q, p \cdot 0) | p, q \in T(TSP(A)) \wedge q \not\downarrow \wedge q \not\xrightarrow{}\} \cup \\ &\{(p \cdot q, p \cdot 1) | p, q \in T(TSP(A)) \wedge q \downarrow \wedge q \not\xrightarrow{}\} \cup \\ &\{(p \cdot q,\ p \cdot (q + b.0)) \mid p, q \in T(TSP(A)) \wedge b \in menu(q)\} \end{aligned}$$

It is verified that R is a ready simulation relation. Since ready simulation equivalence on pairs of the shape (p, p) is straightforward, this part is omitted.

Condition 1
Pair $(p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0),\ p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0))$
Suppose $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} s$.

If $p \not\downarrow$, there are four different kinds of possibilities for $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)$ to do a step.

- If $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot q$, $s = p' \cdot q$, then there are three cases.
  - If $q \not\downarrow \wedge q \not\rightarrow$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot 0$ and clearly $(p' \cdot q, p' \cdot 0) \in R$.
  - If $q \downarrow \wedge q \not\rightarrow$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot 1$ and clearly $(p' \cdot q, p' \cdot 1) \in R$.
  - If $q \rightarrow$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot (q + b.0)$ where $b \in menu(q)$ and clearly $(p' \cdot q, p' \cdot (q + b.0)) \in R$.
- If $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot 0$, $s = p' \cdot 0$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot 0$ and clearly $(p' \cdot 0, p' \cdot 0) \in R$.
- If $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot 1$, $s = p' \cdot 1$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot 1$ and clearly $(p' \cdot 1, p' \cdot 1) \in R$.
- If $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot (q + b.0)$, for any $b \in A$, $s = p' \cdot (q + b.0)$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{a} p' \cdot (q + b.0)$ and clearly $(p' \cdot (q + b.0), p' \cdot (q + b.0) \in R$.

If $p \downarrow$, there are two different kinds of possibilities for $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)$ to do a step.

- If $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{b} q'$ (b originates from the first or last summand), $s = q'$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{b} q'$ and clearly $(q', q') \in R$.
- If $p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{b} 0$ (b originates from the last summand), for any $b \in A$, $s = 0$, then $p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0) \xrightarrow{b} 0$ and clearly $(0, 0) \in R$.

Pair $(p \cdot q,\ p \cdot 0)$ *where* $q \not\downarrow \wedge q \not\rightarrow$
Suppose $p \cdot q \xrightarrow{a} s$. Then there are two cases:

- If $p \not\downarrow$, then $p \cdot q \xrightarrow{a} p' \cdot q$, $s = p' \cdot q$. Then also $p \cdot 0 \xrightarrow{a} p' \cdot 0$ and clearly $(p' \cdot q, p' \cdot 0) \in R$.
- If $p \downarrow$, then $q \not\rightarrow$.

Pair $(p \cdot q,\ p \cdot 1)$ *where* $q \downarrow \wedge q \not\rightarrow$
Suppose $p \cdot q \xrightarrow{a} s$. Then there are two cases:

- If $p \not\downarrow$, then $p \cdot q \xrightarrow{a} p' \cdot q$, $s = p' \cdot q$. Then also $p \cdot 1 \xrightarrow{a} p' \cdot 1$ and clearly $(p' \cdot q, p' \cdot 1) \in R$.
- If $p \downarrow$, then $q \not\rightarrow$.

Pair $(p \cdot q,\ p \cdot (q + b.0))$ *where* $b \in menu(q)$
Suppose $p \cdot q \xrightarrow{a} s$. Then there are two cases:

- If $p \cdot q \xrightarrow{a} p' \cdot q$, $s = p' \cdot q$, then also $p \cdot (q + b.0) \xrightarrow{a} p' \cdot (q + b.0)$ and clearly $(p' \cdot q, p' \cdot (q + b.0)) \in R$.
- If $p \downarrow$ and $q \xrightarrow{b} q'$, $s = q'$, then also since $p \downarrow$ we have $p \cdot (q + b.0) \xrightarrow{b} q'$ and clearly $(q', q') \in R$.

Condition 2
Pair $(p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0),\ p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0))$
There are two cases:

- If $p \not\downarrow$, we have $menu(p \cdot q + p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0)) = menu(p) = menu(p \cdot 0 + p \cdot 1 + \sum_{b \in A} p \cdot (q + b.0))$.

- If $p\downarrow$, we have $menu(p\cdot q+p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0))=menu(q)+menu(0)+menu(1)+\sum_{b\in A}menu(q+b.0)=menu(q)+0+1+\sum_{b\in A}(menu(q)+menu(b.0))=menu(q)+1+menu(q)+\sum_{b\in A}menu(b.0)=1+menu(q)+\sum_{b\in A}menu(b.0)=1+\sum_{b\in A}(menu(q)+menu(b.0))$. Since $p\downarrow$ this equals $menu(p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0))$.

Pair $(p\cdot q,\ p\cdot 0)$ *where* $q\not\downarrow\wedge q\not\rightarrow$
There are two cases:

- If $p\not\downarrow$, then $menu(p\cdot q)=menu(p)=menu(p\cdot 0)$.
- If $p\downarrow$, then $menu(p\cdot q)=menu(q)$. Since by assumption we have $q\not\downarrow\wedge q\not\rightarrow$, we know that $menu(q)=\varnothing$ and since $p\downarrow$, we also have $menu(p\cdot 0)=\varnothing$. Hence, $menu(p\cdot q)=\varnothing=menu(p\cdot 0)$.

Pair $(p\cdot q,\ p\cdot 1)$ *where* $q\downarrow\wedge q\not\rightarrow$
There are two cases:

- If $p\not\downarrow$, then $menu(p\cdot q)=menu(p)=menu(p\cdot 1)$.
- If $p\downarrow$, then $menu(p\cdot q)=menu(q)$. Since by assumption we have $q\downarrow\wedge q\not\rightarrow$, we know that $menu(q)=1$ and since $p\downarrow$, we also have $menu(p\cdot 1)=1$. Hence, $menu(p\cdot q)=1=menu(p\cdot 1)$.

Pair $(p\cdot q,\ p\cdot(q+b.0))$ *where* $b\in menu(q)$
There are two cases:

- If $p\not\downarrow$, then $menu(p\cdot q)=menu(p)=menu(p\cdot(q+b.0))$.
- If $p\downarrow$, then $menu(p\cdot q)=menu(q)$. Since by assumption we have $b\in menu(q)$, we know that $menu(q)=menu(q+b.0)$ and since $p\downarrow$, this equals $menu(p\cdot(q+b.0))$. Hence, $menu(p\cdot q)=menu(p\cdot(q+b.0))$.

This proves that there exists a ready simulation R with $p\cdot q+p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)\ R\ p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)$.
We define:

$$S=\{(p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0),\ p\cdot q+p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0))\mid p,q\in T(TSP(A))\}\cup\{(p,p)\mid p\in T(TSP(A))\}$$

Verifying that S is a ready simulation relation is similar to the verification above and is therefore omitted. Hence, since $p\cdot q+p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)\ R\ p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)$ and $p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)\ S\ p\cdot q+p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)$, it is concluded that $p\cdot q+p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)=_{RS}p\cdot 0+p\cdot 1+\sum_{b\in A}p\cdot(q+b.0)$. □

So, we have proved that we can derive the infinite family of equations (Equation 4.5) of BSP(A) modulo ready simulation semantics from a sound and finite collection of equations over TSP(A) modulo ready simulation semantics. Note that this only proves that we cannot reuse the result of BSP(A) modulo ready simulation semantics for TSP(A) and that this does not prove whether TSP(A) modulo ready simulation semantics affords a finite basis.

Ready simulation semantics with $|A|=\infty$

We observe that we cannot use the proof of Theorem 4.1.18 in Section 4.1.3. If we use the substitution of Theorem 4.1.18 with $R(t\cdot u)=R(t)\cdot R(u)$ added to the definition of R, then condition 3 of Theorem 4.1.17 does not hold for A10. More specifically, if $a=a_z$ we have that the left-hand side of A10 equals $R((a_z.\sigma(x))\cdot\sigma(y))=R(a_z.\sigma(x))\cdot R(\sigma(y))=z\cdot R(\sigma(y))$, whereas the right-hand side of A10 equals $R(a_z.\sigma(x\cdot y))=z$. Clearly $z\cdot R(\sigma(y))\neq z$ and hence we cannot use the proof of Theorem 4.1.18.

4.2.4 Infinite families of equations with $1 \leq |A| < \infty$

In [7] infinite families of equations are given that are sound modulo the semantics but which cannot be derived from a sound and finite collection of equations over BCCSP(A). We consider the equations given for completed simulation semantics and failure trace semantics and check if these can be derived from a sound and finite collection of equations over TSP(A).

Completed simulation semantics

For BCCSP(A) modulo completed simulation semantics, the following infinite family of equations is provided in [7]:

$$a^n.x + a^n.0 + a^n.(x+y) = a^n.0 + a^n.(x+y) \tag{4.7}$$

Since we consider TSP, we need to take termination into account and we change Equation 4.7 to the following:

$$a^n.x + a^n.0 + a^n.1 + a^n.(x+y) = a^n.0 + a^n.1 + a^n.(x+y) \tag{4.8}$$

The idea of Equation 4.8 is that either x cannot perform an action and cannot terminate, in which case $a^n.x$ is completed simulated by $a^n.0$, or x cannot perform an action and can terminate, in which case $a^n.x$ is completed simulated by $a^n.1$ or x can perform some action b, in which case $a^n.x$ is completed simulated by $a^n.(x+y)$.
Since TSP has the sequential composition operator, we can rewrite Equation 4.8 to:

$$x \cdot y + x \cdot 0 + x \cdot 1 + x \cdot (y+z) = x \cdot 0 + x \cdot 1 + x \cdot (y+z) \tag{4.9}$$

Note that Equation 4.8 is an instance of Equation 4.9. More specifically, we substitute x of Equation 4.9 by $a^n.1$ and y of Equation 4.9 by x and we apply axioms A9 and A10. An elaborate soundness proof for this equation can be found in Appendix A.3.

Failure trace semantics

For BCCSP(A) modulo failure trace semantics, the following infinite family of equations is provided in [7]:

$$a^{n+1}.x + a.(a^n.x + x) + a.\sum_{b \in A \setminus \{a\}} a^n.(b.0 + x) = a.(a^n.x + x) + a.\sum_{b \in A \setminus \{a\}} a^n.(b.0 + x) \tag{4.10}$$

Since we consider TSP, we need to take termination into account and we change Equation 4.10 to the following:

$$\begin{aligned} a^{n+1}.x + a^{n+1}.0 + a^{n+1}.1 + a.(a^n.x + x) + a.\sum_{b \in A \setminus \{a\}} a^n.(b.0 + x) = \\ a^{n+1}.0 + a^{n+1}.1 + a.(a^n.x + x) + a.\sum_{b \in A \setminus \{a\}} a^n.(b.0 + x) \end{aligned} \tag{4.11}$$

The idea of Equation 4.11 is:

- If $menu(x) = \varnothing$, then the failure traces of $a^{n+1}.x$ are included in those of $a^{n+1}.0$.
- If $menu(x) = 1$, then the failure traces of $a^{n+1}.x$ are included in those of $a^{n+1}.1$.
- If $menu(x) \setminus \{1\} \subseteq \{a\}$, then the failure traces of $a^{n+1}.x$ are included in those of $a.(a^n.x + x)$.
- If $c \in menu(x) \setminus \{1\}$ and $c \neq a$, then the failure traces of $a^{n+1}.x$ are included in those of $a.\sum_{b \in A \setminus \{a\}} a^n.(b.0 + x)$

Since TSP has the sequential composition operator, we can rewrite Equation 4.11 to:

$$\begin{aligned} a.x \cdot y + a.x \cdot 0 + a.x \cdot 1 + a.(x \cdot y + y) + a. \sum_{b \in A \setminus \{I(x)\}} x \cdot (b.0 + y) = \\ a.x \cdot 0 + a.x \cdot 1 + a.(x \cdot y + y) + a. \sum_{b \in A \setminus \{I(x)\}} x \cdot (b.0 + y) \end{aligned} \tag{4.12}$$

Note that Equation 4.11 is an instance of Equation 4.12. More specifically, we substitute x of Equation 4.12 by $a^n.1$ and y of Equation 4.12 by x and we apply axioms A9 and A10.
The idea of Equation 4.12 is:

- If $menu(y) = \varnothing$, then the failure traces of $a.x \cdot y$ are included in those of $a.x \cdot 0$.
- If $menu(y) = 1$, then the failure traces of $a.x \cdot y$ are included in those of $a.x \cdot 1$.
- If $menu(y) \setminus \{1\} \subseteq menu(x) \setminus \{1\}$, then the failure traces of $a.x \cdot y$ are included in those of $a.(x \cdot y + y)$.
- If $c \in menu(y) \setminus \{1\}$ and $c \notin menu(x) \setminus \{1\}$, then the failure traces of $a.x \cdot y$ are included in those of $a. \sum_{b \in A \setminus \{I(x)\}} x.(b.0 + y)$

Chapter 5

Conclusions

We have added sequential composition and successful termination to van Glabbeek's linear time - branching time spectrum, by considering BSP and TSP instead of BCCSP. For this we used different definitions of the semantics, taking successful termination into account.

We gave a template explaining how to prove soundness and ground-completeness "efficiently", i.e. in such a way that we can reuse (part of) the proofs. Namely, for $o \in \mathcal{O}$ soundness of axioms for BSP(A) modulo o-semantics follows from soundness of these axioms for BSP(A) modulo o'-semantics where $o' \in \mathcal{O}$ and $o' \subseteq o$. Ground-completeness can be proved by defining a set of terms in o normal form and proving that if p, q are in o normal form, then $p =_o q \Leftrightarrow p =_{o'} q$ for $o' \subseteq o$ where BSP modulo o' is proven to be ground-complete.

Soundness and ground-completeness of BSP modulo trace semantics and soundness of BSP modulo ready simulation semantics are proved by following this template. Since following the template to prove ground-completeness of BSP modulo ready simulation semantics yields an unnecessarily complex proof, we used a technique from [10] instead. Finally, we showed that the results of BSP can be used for TSP and we provided an overview of sound and ground-complete axiomatisations for TSP modulo o-semantics, where $o \in \mathcal{O}$.

We also considered the work in [7], i.e. the existence of finite bases, for BSP and TSP instead of BCCSP. For BSP and TSP modulo o-semantics, where $o \in \mathcal{O}$, we introduced new axioms when $|A| = 0$ and we established that BSP and TSP together with their respective new axioms are ω-complete. We showed that for BSP modulo ready simulation semantics a finite basis does not exist by giving an infinite family of equations which are sound but cannot be derived from BSP_{Rs}. On the other hand, when $|A| = \infty$, BSP modulo ready simulation semantics affords a finite basis, which we proved using Groote's method of inverted substitutions [11].

Furthermore, we proved ω-completeness for TSP modulo bisimilarity when $|A| \geq 1$ by defining a substitution that allows us to distinguish variables by giving them the possibility to increase *norm* significantly or to decrease *norm* by 1. Finally, we observed that for TSP modulo ready simulation semantics we cannot reuse the previous results, i.e. the proof for BSP modulo ready simulation semantics or the proof for TSP modulo bisimilarity, to prove whether it is ω-complete and that the infinite families of equations given in [7] for BCCSP modulo completed simulation and failure trace semantics can be derived from a sound and finite collection of equations over TSP(A).

Future work may include proving the existence or lack of finite bases for BSP and TSP modulo the other semantics in van Glabbeek's spectrum. In addition to this one could consider extending the theory with one or multiple operators, such as parallel composition [2], the alternative version of sequential composition [3] and Kleene star [1] and give sound, ground-complete axiomatisations modulo the semantics and prove whether they are finitely based over these extended theories. Finally, one could consider how adding recursion [2] impacts the axiomatisations.

Bibliography

[1] Aceto, L., Fokkink, W., & Ingólfsdóttir, A. (1998). A Menagerie of Non-Finitely Based Process Semantics over BPA* – From Ready Simulation to Completed Traces. *Mathematical Structures in Computer Science, 8*(3), 193-230.
Cited on page: 49

[2] Baeten, J.C.M., Basten, T. & Reniers, M.A. (2010). Process Algebra: Equational Theories of Communicating Processes, volume 50. Cambridge university press.
Cited on pages: 1, 2, 5, 8, 9, 10, 11, 12, 13, 17, 20, 21, 22, 23, 24, 25, 26, 28, 49

[3] Baeten, J.C.M., Luttik, B., & Yang, F (2017). Sequential Composition in the Presence of Intermediate Termination. *arXiv preprint arXiv:1706.08401.*
Cited on page: 49

[4] Baeten, J.C.M., & Weijland, W.P. (1990). Process Algebra. Cambridge University Press.
Cited on page: 26

[5] Bergstra, J.A., & Klop, J.W. (1984). Process Algebra for Synchronous Communication. *Information and Control, 60*(1-3), 109-137.
Cited on page: 26

[6] Blom, S.C.C., Fokkink, W.J., & Nain, S. (2003). On the Axiomatizability of Ready Traces, Ready Simulation and Failure Traces. *Proceedings 30th Colloquium on Automata, Languages and Programming*, Eindhoven, LNCS 2719, pp. 109–118.
Cited on page: 18

[7] Chen, T., Fokkink, W., Luttik, B., & Nain, S. (2008). On finite alphabets and infinite bases. *Information and Computation, 206*(5), 492–519.
Cited on pages: 1, 2, 3, 5, 28, 29, 31, 35, 47, 49, 53

[8] Chen, T., Fokkink, W., & Nain, S. (2006). On Finite Alphabets and Infinite Base II: Completed and Ready Simulation. *Proceedings FOSSACS'06*, LNCS 3921, pp. 1-15.
Cited on pages: 31, 32, 33, 34

[9] Fokkink, W., & Luttik, B. (2000). An ω-complete Equational Specification of Interleaving. *Proceedings 27th Colloquium on Automata, Languages and Programming*, Geneva, LNCS 1853, pp. 729-743.
Cited on page: 39

[10] van Glabbeek, R. (2001). The Linear Time - Branching Time Spectrum I. The Semantics of Concrete, Sequential Processes. *Handbook of Process Algebra*, 3-99.
Cited on pages: 1, 2, 3, 4, 5, 6, 7, 10, 13, 16, 17, 20, 21, 22, 23, 24, 25, 49

[11] Groote, J.F. (1990). A New Strategy for Proving ω-Completeness applied to Process Algebra. *Proceedings 1st Conference on Concurrency Theory*, Amsterdam, LNCS 458, pp. 314-331.
Cited on pages: 2, 28, 29, 34, 49

Appendix A

Soundness B1-B6

In the following sections we prove soundness of B1-B6.

A.1 BSP

We prove that B1 is sound for BSP modulo bisimilarity if $|A| = 0$.

A.1.1 Soundness B1

Lemma A.1.1 (B1-Bisimulation-BSP-$|A| = 0$)**.** *$p + 1 \leftrightarrows 1$ for all closed BSP-terms p.*

Proof. To show that $p + 1$ is bisimilar to 1, we give a bisimulation relation R, with $p + 1\ R\ 1$. We define:

$$R = \{(p + 1,\ 1) \mid p \in T(BSP(A))\}$$

It is verified that R is a bisimulation relation. Since $|A| = 0$, conditions 1 and 2 of Definition 3.2.1 vacuously hold and we only have to check conditions 3 and 4.

Condition 3
We need to prove that if $p + 1 \downarrow$ then also $1 \downarrow$. Since $1 \downarrow$, this holds.

Condition 4
We need to prove that if $1 \downarrow$ then also $p + 1 \downarrow$. Since $1 \downarrow$, also $p + 1 \downarrow$.

Hence, since $p + 1\ R\ 1$ it is concluded that $p + 1 \leftrightarrows 1$. □

A.2 TSP

In the following subsections we prove that B2-B6 are sound for TSP modulo bisimilarity if $|A| = 0$.

A.2.1 Soundness B2

Lemma A.2.1 (B2-Bisimulation-TSP-$|A| = 0$)**.** *$p \cdot 0 \leftrightarrows 0$ for all closed TSP-terms p.*

Proof. To show that $p \cdot 0$ is bisimilar to 0, we give a bisimulation relation R, with $p \cdot 0\ R\ 0$. We define:

$$R = \{(p \cdot 0,\ 0) \mid p \in T(TSP(A))\}$$

It is verified that R is a bisimulation relation. Since $|A| = 0$, conditions 1 and 2 of Definition 3.2.1 vacuously hold and we only have to check conditions 3 and 4.

Condition 3
We need to prove that if $p \cdot 0 \downarrow$ then also $0 \downarrow$. According to the operational rules given in Figure

2.6 $p \cdot 0$ can terminate if and only if $p\downarrow$ and $0\downarrow$. Since $0 \not\downarrow$ it follows that $p \cdot 0 \not\downarrow$. Hence, this case vacuously holds.

Condition 4
We need to prove that if $0\downarrow$ then also $p \cdot 0\downarrow$. Since $0 \not\downarrow$, this case vacuously holds.

Hence, since $p \cdot 0\ R\ 0$ it is concluded that $p \cdot 0 \leftrightarrows 0$. □

A.2.2 Soundness B3

Lemma A.2.2 (B3-Bisimulation-TSP-$|A| = 0$)**.** *$p \cdot q \leftrightarrows q \cdot p$ for all closed TSP-terms p, q.*

Proof. To show that $p \cdot q$ is bisimilar to $q \cdot p$, we give a bisimulation relation R, with $p \cdot q\ R\ q \cdot p$. We define:

$$R = \{(p \cdot q,\ q \cdot p) \mid p, q \in T(TSP(A))\}$$

It is verified that R is a bisimulation relation. Since $|A| = 0$, conditions 1 and 2 of Definition 3.2.1 vacuously hold and we only have to check conditions 3 and 4.

Condition 3
We need to prove that if $p \cdot q\downarrow$ then also $q \cdot p\downarrow$. According to the operational rules given in Figure 2.6 $p \cdot q$ can terminate if and only if $p\downarrow$ and $q\downarrow$. Then by the operational rules $q \cdot p\downarrow$.

Condition 4
We need to prove that if $q \cdot p\downarrow$ then also $p \cdot q\downarrow$. The proof for this is similar to the proof of condition 3.

Hence, since $p \cdot q\ R\ q \cdot p$ it is concluded that $p \cdot q \leftrightarrows q \cdot p$. □

A.2.3 Soundness B4

Lemma A.2.3 (B4-Bisimulation-TSP-$|A| = 0$)**.** *$p \cdot p \leftrightarrows p$ for all closed TSP-terms p.*

Proof. To show that $p \cdot p$ is bisimilar to p, we give a bisimulation relation R, with $p \cdot p\ R\ p$. We define:

$$R = \{(p \cdot p,\ p) \mid p \in T(TSP(A))\}$$

It is verified that R is a bisimulation relation. Since $|A| = 0$, conditions 1 and 2 of Definition 3.2.1 vacuously hold and we only have to check conditions 3 and 4.

Condition 3
We need to prove that if $p \cdot p\downarrow$ then also $p\downarrow$. According to the operational rules given in Figure 2.6 $p \cdot p$ can terminate if and only if both sides of the sequential composition can terminate. Hence $p \cdot p$ can only terminate if $p\downarrow$.

Condition 4
We need to prove that if $p\downarrow$ then also $p \cdot p\downarrow$. Since we know that $p\downarrow$, then we know that both sides of the sequential composition $p \cdot p$ can terminate. Hence, it follows by the operational rules that $p \cdot p\downarrow$.

Hence, since $p \cdot p\ R\ p$ it is concluded that $p \cdot p \leftrightarrows p$. □

A.2.4 Soundness B5

Lemma A.2.4 (B5-Bisimulation-TSP-$|A| = 0$)**.** *$p + p \cdot q \leftrightarrows p$ for all closed TSP-terms p, q.*

Proof. To show that $p + p \cdot q$ is bisimilar to p, we give a bisimulation relation R, with $p + p \cdot q\ R\ p$. We define:

$$R = \{(p + p \cdot q,\ p) \mid p, q \in T(TSP(A))\}$$

It is verified that R is a bisimulation relation. Since $|A| = 0$, conditions 1 and 2 of Definition 3.2.1 vacuously hold and we only have to check conditions 3 and 4.

Condition 3
We need to prove that if $p+p\cdot q\downarrow$ then also $p\downarrow$. According to the operational rules given in Figure 2.6 $p+p\cdot q$ can terminate if $p\downarrow$ or $p\cdot q\downarrow$. Furthermore, $p\cdot q\downarrow$ if and only if $p\downarrow$ and $q\downarrow$. Hence, $p+p\cdot q$ can only terminate if $p\downarrow$.

Condition 4
We need to prove that if $p\downarrow$ then also $p+p\cdot q\downarrow$. Since we know that $p\downarrow$, then we know by the operational rules that $p+p\cdot q\downarrow$.

Hence, since $p+p\cdot q\ R\ p$ it is concluded that $p+p\cdot q\leftrightarrows p$. □

A.2.5 Soundness B6

Lemma A.2.5 (B6-Bisimulation-TSP-$|A|=0$)**.** *$p\cdot(q+r)\leftrightarrows p\cdot q+p\cdot r$ for all closed TSP-terms p,q,r.*

Proof. To show that $p\cdot(q+r)$ is bisimilar to $p\cdot q+p\cdot r$, we give a bisimulation relation R, with $p\cdot(q+r)\ R\ p\cdot q+p\cdot r$. We define:

$$R=\{(p\cdot(q+r),\ p\cdot q+p\cdot r)\mid p,q,r\in T(TSP(A))\}$$

It is verified that R is a bisimulation relation. Since $|A|=0$, conditions 1 and 2 of Definition 3.2.1 vacuously hold and we only have to check conditions 3 and 4.

Condition 3
We need to prove that if $p\cdot(q+r)\downarrow$ then also $p\cdot q+p\cdot r\downarrow$. According to the operational rules given in Figure 2.6 $p\cdot(q+r)$ can terminate if and only if $p\downarrow$ and $q+r\downarrow$. Furthermore, $q+r\downarrow$ if $q\downarrow$ or $r\downarrow$. Hence, $p\cdot(q+r)\downarrow$ if and only if $p\downarrow$ and $q\downarrow$ or if $p\downarrow$ and $r\downarrow$. Suppose that $p\downarrow$ and $q\downarrow$. Then $p\cdot q\downarrow$ and hence $p\cdot q+p\cdot r\downarrow$. Suppose that $p\downarrow$ and $r\downarrow$. Then $p\cdot r\downarrow$ and hence $p\cdot q+p\cdot r\downarrow$.

Condition 4
We need to prove that if $p\cdot q+p\cdot r\downarrow$ then also $p\cdot(q+r)\downarrow$. According to the operational rules $p\cdot q+p\cdot r\downarrow$ if $p\cdot q\downarrow$, which is the case if $p\downarrow$ and $q\downarrow$, or if $p\cdot r\downarrow$, which is the case if $p\downarrow$ and $r\downarrow$. Suppose $p\downarrow$ and $q\downarrow$. Then $(q+r)\downarrow$ and also $p\cdot(q+r)\downarrow$. Suppose $p\downarrow$ and $r\downarrow$. Then $(q+r)\downarrow$ and also $p\cdot(q+r)\downarrow$.

Hence, since $p\cdot(q+r)\ R\ p\cdot q+p\cdot r$ it is concluded that $p\cdot(q+r)\leftrightarrows p\cdot q+p\cdot r$. □

A.3 Infinite family of equations

We prove that the infinite family of equations given in [7] for BCCSP can be derived from a single axiom in TSP.

A.3.1 Completed simulation semantics

Lemma A.3.1. *$p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)=_{CS}p\cdot 0+p\cdot 1+p\cdot(q+r)$ for all closed TSP(A)-terms p,q,r.*

Proof. To show that $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)$ is completed simulation equivalent to $p\cdot 0+p\cdot 1+p\cdot(q+r)$, we give two completed simulations R and S, with $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\ R\ p\cdot 0+p\cdot 1+p\cdot(q+r)$ and $p\cdot 0+p\cdot 1+p\cdot(q+r)\ S\ p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)$.
We define:

$$\begin{aligned}R=&\{(p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)),\ p\cdot 0+p\cdot 1+p\cdot(q+r))\mid\\ &p,q\in T(TSP(A))\}\cup\{(p,\ p)\mid p\in T(TSP(A))\}\cup\\ &\{(p\cdot q,\ p\cdot 0)\mid p,q\in T(TSP(A))\wedge q\not\downarrow\wedge q\not\to\}\cup\\ &\{(p\cdot q,\ p\cdot 1)\mid p,q\in T(TSP(A))\wedge q\downarrow\wedge q\not\to\}\cup\\ &\{(p\cdot q,\ p\cdot(q+r))\mid p,q,r\in T(TSP(A))\wedge q\to\}\end{aligned}$$

It is verified that R is a completed simulation relation. Since completed simulation equivalence on pairs of the shape (p,p) is straightforward, this part is omitted.

Condition 1
Pair $(p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r),\ p\cdot 0+p\cdot 1+p\cdot(q+r))$
Suppose $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} s$.
If $p\not\downarrow$, there are four different kinds of possibilities for $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)$ to do a step.

- If $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot q$, $s=p'\cdot q$, then there are three cases.
  - If $q\not\downarrow\wedge q\not\rightarrow$, then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot 0$ and clearly $(p'\cdot q,p'\cdot 0)\in R$.
  - If $q\downarrow\wedge q\not\rightarrow$, then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot 1$ and clearly $(p'\cdot q,p'\cdot 1)\in R$.
  - If $q\rightarrow$, then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot(q+r)$ and clearly $(p'\cdot q,p'\cdot(q+r))\in R$.
- If $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot 0$, $s=p'\cdot 0$, then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot 0$ and clearly $(p'\cdot 0,p'\cdot 0)\in R$.
- If $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot 1$, $s=p'\cdot 1$, then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot 1$ and clearly $(p'\cdot 1,p'\cdot 1)\in R$.
- If $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot(q+r)$, then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{a} p'\cdot(q+r)$ and clearly $(p'\cdot(q+r),p'\cdot(q+r)\in R$.

If $p\downarrow$, there are two different kinds of possibilities for $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)$ to do a step.

- If $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{b} q'$ (b originates from the first or last summand), $s=q'$, then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{b} q'$ and clearly $(q',q')\in R$.
- If $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{b} r'$ (b originates from the last summand), then $p\cdot 0+p\cdot 1+p\cdot(q+r)\xrightarrow{b} r'$ and clearly $(r',r')\in R$.

Pair $(p\cdot q,\ p\cdot 0)$ *where* $q\not\downarrow\wedge q\not\rightarrow$
Suppose $p\cdot q\xrightarrow{a} s$. Then there are two cases:

- If $p\not\downarrow$, then $p\cdot q\xrightarrow{a} p'\cdot q$, $s=p'\cdot q$. Then also $p\cdot 0\xrightarrow{a} p'\cdot 0$ and clearly $(p'\cdot q,p'\cdot 0)\in R$.
- If $p\downarrow$, then $q\not\rightarrow$.

Pair $(p\cdot q,\ p\cdot 1)$ *where* $q\downarrow\wedge q\not\rightarrow$
Suppose $p\cdot q\xrightarrow{a} s$. Then there are two cases:

- If $p\not\downarrow$, then $p\cdot q\xrightarrow{a} p'\cdot q$, $s=p'\cdot q$. Then also $p\cdot 1\xrightarrow{a} p'\cdot 1$ and clearly $(p'\cdot q,p'\cdot 1)\in R$.
- If $p\downarrow$, then $q\not\rightarrow$.

Pair $(p\cdot q,\ p\cdot(q+r))$ *where* $q\rightarrow$
Suppose $p\cdot q\xrightarrow{a} s$. Then there are two cases:

- If $p\not\downarrow$, then $p\cdot q\xrightarrow{a} p'\cdot q$, $s=p'\cdot q$. Then also $p\cdot(q+r)\xrightarrow{a} p'\cdot(q+r)$ and clearly $(p'\cdot q,p'\cdot(q+r))\in R$.
- If $p\downarrow$, then $p\cdot q\xrightarrow{b} q'$, $s=q'$. then also $p\cdot(q+r)\xrightarrow{b} q'$ and clearly $(q',q')\in R$.

Condition 2
Pair $(p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r),\ p\cdot 0+p\cdot 1+p\cdot(q+r))$
There are two cases:

- If $p\not\downarrow$, we have $menu(p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r))=menu(p)=menu(p\cdot 0+p\cdot 1+p\cdot(q+r))$.

- If $p\downarrow$, we have $menu(p{\cdot}q{+}p{\cdot}0{+}p{\cdot}1{+}p{\cdot}(q{+}r)) = menu(q){+}menu(0){+}menu(1){+}menu(q{+}r) = menu(q)+0+1+menu(q)+menu(r) = 1+menu(q)+menu(r) = 1+menu(q+r)$. Since $p\downarrow$ this equals $menu(p\cdot 0+p\cdot 1+p\cdot(q+r))$.

Pair $(p\cdot q,\ p\cdot 0)$ *where* $q\not\downarrow \wedge q\not\rightarrow$
There are two cases:

- If $p\not\downarrow$, then $menu(p\cdot q) = menu(p) = menu(p\cdot 0)$.
- If $p\downarrow$, then $menu(p\cdot q) = menu(q)$. Since by assumption we have $q\not\downarrow \wedge q\not\rightarrow$, we know that $menu(q) = \varnothing$ and since $p\downarrow$, we also have $menu(p{\cdot}0) = \varnothing$. Hence, $menu(p{\cdot}q) = \varnothing = menu(p{\cdot}0)$.

Pair $(p\cdot q,\ p\cdot 1)$ *where* $q\downarrow \wedge q\not\rightarrow$
There are two cases:

- If $p\not\downarrow$, then $menu(p\cdot q) = menu(p) = menu(p\cdot 1)$.
- If $p\downarrow$, then $menu(p\cdot q) = menu(q)$. Since by assumption we have $q\downarrow \wedge q\not\rightarrow$, we know that $menu(q) = 1$ and so $menu(p\cdot q)\setminus\{1\} = \varnothing$. Since $p\downarrow$, we also have $menu(p\cdot 1) = 1$ and so $menu(p\cdot 1)\setminus\{1\} = \varnothing$. Hence, $menu(p\cdot q)\setminus\{1\} = \varnothing = menu(p\cdot 1)\setminus\{1\}$.

Pair $(p\cdot q,\ p\cdot(q+r))$ *where* $q\rightarrow$
There are two cases:

- If $p\not\downarrow$, then $menu(p\cdot q) = menu(p) = menu(p\cdot(q+r))$.
- If $p\downarrow$, then $menu(p\cdot q) = menu(q)$. Since by assumption we have $q\rightarrow$, we know that $menu(q)\neq\varnothing$ and so $menu(p\cdot q)\setminus\{1\}\neq\varnothing$, which means that the condition vacuously holds.

Condition 3
Pair $(p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r),\ p\cdot 0+p\cdot 1+p\cdot(q+r))$
Suppose $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\downarrow$. This can only be the case if one of the summands can terminate, so we have four cases.

- If $p\cdot q\downarrow$, then it must be the case that $p\downarrow$ and $q\downarrow$. Then we also have $p\cdot(q+r)\downarrow$ and hence $p\cdot 0+p\cdot 1+p\cdot(q+r)\downarrow$.
- We have $p\cdot 0\not\downarrow$.
- If $p\cdot 1\downarrow$, then it must be the case that $p\downarrow$ and we also have $p\cdot 0+p\cdot 1+p\cdot(q+r)\downarrow$.
- If $p\cdot(q+r)\downarrow$, then it must be the case that either $p\downarrow$ and $q\downarrow$ or $p\downarrow$ and $r\downarrow$. Then we also have $p\cdot(q+r)\downarrow$ and hence $p\cdot 0+p\cdot 1+p\cdot(q+r)\downarrow$.

Pair $(p\cdot q,\ p\cdot 0)$ *where* $q\not\downarrow \wedge q\not\rightarrow$
By assumption we know that $q\not\downarrow$ and so $p\cdot q\not\downarrow$ and the condition vacuously holds.

Pair $(p\cdot q,\ p\cdot 1)$ *where* $q\downarrow \wedge q\not\rightarrow$
Suppose $p\cdot q\downarrow$. This can only be the case if $p\downarrow$ and $q\downarrow$. Then we also have that $p\cdot 1\downarrow$.

Pair $(p\cdot q,\ p\cdot(q+r))$ *where* $q\rightarrow$
Suppose $p\cdot q\downarrow$. This can only be the case if $p\downarrow$ and $q\downarrow$. Then we also have that $p\cdot(q+r)\downarrow$.

This proves that there exists a completed simulation R with $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\ R\ p\cdot 0+p\cdot 1+p\cdot(q+r)$.
We define:

$$S = \{(p\cdot 0+p\cdot 1+p\cdot(q+r),\ p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)) \mid p,q\in T(TSP(A))\}\cup\{(p,p)\mid p\in T(TSP(A))\}$$

Verifying that S is a completed simulation relation is similar to the verification above and is therefore omitted. Hence, since $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)\ R\ p\cdot 0+p\cdot 1+p\cdot(q+r)$ and $p\cdot 0+p\cdot 1+p\cdot(q+r)\ S\ p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r)$, it is concluded that $p\cdot q+p\cdot 0+p\cdot 1+p\cdot(q+r) =_{CS} p\cdot 0+p\cdot 1+p\cdot(q+r)$. □