Exploiting Algebraic Laws to Improve Mechanized Axiomatizations*

Luca Aceto[1], Eugen-Ioan Goriac[1], Anna Ingolfsdottir[1], Mohammad Reza Mousavi[2], and Michel A. Reniers[3]

[1] ICE-TCS, School of Computer Science, Reykjavik University, Menntavegur 1, IS-101, Reykjavik, Iceland
[2] Center for Research on Embedded Systems (CERES), Halmstad University, Sweden
[3] Department of Mechanical Engineering, Eindhoven University of Technology, P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands

Abstract. In the field of structural operational semantics (SOS), there have been several proposals both for syntactic rule formats guaranteeing the validity of algebraic laws, and for algorithms for automatically generating ground-complete axiomatizations. However, there has been no synergy between these two types of results. This paper takes the first steps in marrying these two areas of research in the meta-theory of SOS and shows that taking algebraic laws into account in the mechanical generation of axiomatizations results in simpler axiomatizations. The proposed theory is applied to a paradigmatic example from the literature, showing that, in this case, the generated axiomatization coincides with a classic hand-crafted one.

1 Introduction

Algebraic properties, such as commutativity, associativity and idempotence of binary operators, specify some natural properties of programming and specification constructs. These properties can either be validated using the semantics of the language with respect to a suitable notion of program equivalence, or they can be guaranteed a priori 'by design'. In particular, for languages equipped with a Structural Operational Semantics (SOS) [19], there are two closely related lines of work to achieve this goal: firstly, there is a rich body of syntactic rule formats that can guarantee the validity of certain algebraic properties; see [5, 17] for recent surveys. Secondly, there are numerous results regarding the mechanical generation of ground-complete axiomatizations of various behavioral equivalences and preorders for SOS language specifications in certain formats—see, e.g., [1, 7, 20]. However, these two lines of research have evolved separately and no link has been established between the two types of results so far. In this paper, we take the first steps in marrying these two research areas and in using rule formats for algebraic properties (specifically, for commutativity) to enhance the

* The first three authors have been partially supported by the project 'Meta-theory of Algebraic Process Theories' (nr. 100014021) of the Icelandic Research Fund. Eugen-Ioan Goriac is also funded by the project 'Extending and Axiomatizing Structural Operational Semantics: Theory and Tools' (nr. 1102940061) of the Icelandic Research Fund.

process of automatic generation of axiomatizations for strong bisimilarity from GSOS language specifications [10]. In particular, we show that linking these two areas results in axiomatizations that look like hand-crafted ones.

Contribution and Related Work. Many ground-completeness results have been presented in the literature on process calculi. (See, for instance, the survey paper [3] for pointers to the literature.) A common proof strategy for establishing such ground-completeness results is to reduce the problem of axiomatizing the notion of behavioural equivalence under consideration over arbitrary closed terms to that of axiomatizing it over 'synchronization-tree terms'. This approach is also at the heart of the algorithm proposed in [1] for the automatic generation of finite, equational, ground-complete axiomatizations for bisimilarity over language specifications in the GSOS format. A variation on that algorithm for GSOS language specifications with termination has been presented in [7]. In [20], Ulidowski has instead offered algorithms for the automatic generation of finite axiom systems for the testing preorder over de Simone process languages. In Section 4 of this paper, we present a refinement of the algorithm from [1] that uses a rule format guaranteeing commutativity of certain operators to obtain ground-complete axiomatizations of bisimilarity that are closer to the hand-crafted ones than those produced by existing algorithms. (See Section 5, where we apply the algorithm to axiomatize the classic parallel composition operator and compare the generated axiomatization to earlier ones.)

Our rule format for commutativity (presented in Section 3) is a generalization of the rule format for commutativity from [16], which allows operators to have various sets of commutative arguments. Apart from being natural, such a generalization is useful in the automatic generation of ground-complete axiomatizations, as the developments in this study show.

2 Preliminaries

In this section we review, for the sake of completeness, some standard definitions from process theory and the meta-theory of SOS that will be used in the remainder of the paper. We refer the interested reader to [4, 17] for further details.

Transition System Specifications in GSOS Format. We let V denote an infinite set of variables with typical members $x, x', x_i, y, y', y_i, \ldots$. A *signature* Σ is a set of function symbols, each with a fixed arity. We call these symbols *operators* and usually represent them by $f, g, \ldots$. An operator with arity zero is called a *constant*. We define the set $\mathbb{T}(\Sigma)$ of *terms* over Σ (sometimes referred to as Σ-terms) as the smallest set satisfying the following constraints: (1) A variable $x \in V$ is a term. (2) If $f \in \Sigma$ has arity n and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term. We use $s, t, t', t_i, u, \ldots$ to range over terms. We write $t_1 \equiv t_2$ if t_1 and t_2 are syntactically equal. The function $vars : \mathbb{T}(\Sigma) \to 2^V$ gives the set of variables appearing in a term. The set $\mathbb{C}(\Sigma)$ is the set of *closed terms*, i.e., the set of all terms t such that $vars(t) = \emptyset$. We use $p, p', p_i, q, r \ldots$ to range over closed terms. A *substitution* σ is a function of type $V \to \mathbb{T}(\Sigma)$. We extend the domain of substitutions to terms homomorphically. If the range of a substitution lies in $\mathbb{C}(\Sigma)$, we say that it is a *closed substitution*.

The GSOS format is a widely studied format of deduction rules in transition system specifications proposed by Bloom, Istrail and Meyer [10]. Transition system specifications whose rules are in the GSOS format enjoy many desirable properties, and several studies in the literature on the meta-theory of SOS have focused on them—see, e.g., the survey [4]. Following [1], in this study we shall also focus on transition system specifications in the GSOS format, which we now proceed to define.

Definition 1 (GSOS Format [10]). *A deduction rule for an operator f of arity n is in the GSOS format if and only if it has the following form:*

$$\frac{\{x_i \xrightarrow{l_{ij}} y_{ij} \mid 1 \leq i \leq n, 1 \leq j \leq m_i\} \cup \{x_i \stackrel{l_{ik}}{\nrightarrow} \mid 1 \leq i \leq n, 1 \leq k \leq n_i\}}{f(\overrightarrow{x}) \xrightarrow{l} C[\overrightarrow{x}, \overrightarrow{y}]}$$

*where the x_i's and the y_{ij}'s ($1 \leq i \leq n$ and $1 \leq j \leq m_i$) are all distinct variables, m_i and n_i are natural numbers, $C[\overrightarrow{x}, \overrightarrow{y}]$ is a Σ-term with variables including at most the x_i's and the y_{ij}'s, and the l_{ij}'s and l are labels. If $m_i > 0$, for some i, then we say that the rule tests its i-th argument positively. The above rule is said to be f-*defining *and* l-emitting.

A transition system specification (TSS) in the GSOS format $\mathcal{T}$ *is a triple* (Σ, L, D) *where Σ is a finite signature, L is a finite set of labels, and D is a finite set of deduction rules in the GSOS format. We shall sometimes refer to a TSS in the GSOS format as a* GSOS system*.*

In addition to the syntactic restrictions on deduction rules, the GSOS format, as presented in [10], requires the signature to include a constant **0**, a collection of unary operators $a.\_$ ($a \in L$) and a binary operator $\_+\_$. Intuitively, **0** represents a process that does not exhibit any behaviour, $s + t$ is the nondeterministic choice between the behaviours of s and t, while $a.t$ is a process that first performs action a and behaves like t afterwards. The standard deduction rules for these operations are given below:

$$\frac{}{a.x_1 \xrightarrow{a} x_1} \qquad \frac{x_1 \xrightarrow{a} x_1'}{x_1 + x_2 \xrightarrow{a} x_1'} \qquad \frac{x_2 \xrightarrow{a} x_2'}{x_1 + x_2 \xrightarrow{a} x_2'}.$$

In the remainder of this paper, following [10], we shall tacitly assume that each TSS in the GSOS format contains these operators with the rules given above. The import of this assumption is that, as is well known, within each TSS in the GSOS format it is possible to express each finite synchronization tree over L. Following [12], the TSS containing the operators **0**, $a.\_$ ($a \in L$) and $\_+\_$, with the above-given rules, is denoted by BCCSP.

The transition relation associated with a TSS in the GSOS format is the one defined by structural induction over closed terms using the rules. We refer the interested reader to [10] for the precise definition and much more information on GSOS languages.

Definition 2 ([1]). *A GSOS system $\mathcal{T}'$ is a* disjoint extension *of a GSOS system $\mathcal{T}$, denoted by $\mathcal{T} \sqsubseteq \mathcal{T}'$, if the signature and rules of $\mathcal{T}'$ include those of $\mathcal{T}$, and $\mathcal{T}'$ introduces no new rules for operators in the signature of $\mathcal{T}$.*

Bisimilarity and Axiom Systems. The notion of behavioural equivalence that we will use in this paper is the following, classic notion of bisimilarity [15, 18].

Definition 3. *Let $\mathcal{T}$ be a GSOS system with signature Σ. A relation $R \subseteq \mathbb{C}(\Sigma) \times \mathbb{C}(\Sigma)$ is a* bisimulation *if and only if R is symmetric and, for all $p_0, p_1, p'_0 \in \mathbb{C}(\Sigma)$ and $l \in L$,*

$$(p_0 \; R \; p_1 \wedge p_0 \xrightarrow{l} p'_0) \Rightarrow \exists p'_1 \in \mathbb{C}(\Sigma).\ (p_1 \xrightarrow{l} p'_1 \wedge p'_0 \; R \; p'_1).$$

Two terms $p_0, p_1 \in \mathbb{C}(\Sigma)$ are called bisimilar*, denoted by $\mathcal{T} \vdash p_0 \leftrightarroweq p_1$ (or simply by $p_0 \leftrightarroweq p_1$ when $\mathcal{T}$ is clear from the context), when there exists a bisimulation R such that $p_0 \; R \; p_1$.*

It is well known that $\leftrightarroweq$ is an equivalence relation over $\mathbb{C}(\Sigma)$. Any equivalence relation $\sim$ over closed terms in a TSS $\mathcal{T}$ is extended to open terms in the standard fashion, i.e., for all $t, u \in \mathbb{T}(\Sigma)$, the equation $t = u$ holds over $\mathcal{T}$ modulo $\sim$ (sometimes abbreviated to $t \sim u$) if, and only if, $\mathcal{T} \vdash \sigma(t) \sim \sigma(u)$ for each closed substitution σ.

Remark 1. If $\mathcal{T}'$ is a disjoint extension of $\mathcal{T}$, then two closed terms over the signature of $\mathcal{T}$ are bisimilar in $\mathcal{T}$ if and only if they are bisimilar in $\mathcal{T}'$.

Proposition 1 ([10]). *$\leftrightarroweq$ is a congruence for any TSS in GSOS format—that is, for all $f \in \Sigma$ and terms $t_1, \ldots, t_n, u_1, \ldots, u_n$, where n is the arity of f, if $t_i \leftrightarroweq u_i$ for each $i \in \{1, \ldots, n\}$ then $f(t_1, \ldots, t_n) \leftrightarroweq f(u_1, \ldots, u_n)$.*

Definition 4 (Axiom System). *An* axiom system *E over a signature Σ is a set of equalities of the form $t = t'$, where $t, t' \in \mathbb{T}(\Sigma)$. An equality $t = t'$, for some $t, t' \in \mathbb{T}(\Sigma)$, is derivable from E, denoted by $E \vdash t = t'$, if and only if it is in the smallest congruence relation over Σ-terms induced by the equalities in E.*

In the context of a fixed TSS $\mathcal{T}$, an axiom system E (over the same signature) is sound *with respect to a congruence relation $\sim$ if and only if for all $t, t' \in \mathbb{T}(\Sigma)$, if $E \vdash t = t'$, then it holds that $\mathcal{T} \vdash t \sim t'$. The axiom system E is* ground complete *if the implication holds in the opposite direction whenever t and t' are closed terms.*

3 Commutativity Format

Commutativity is an essential property specifying that the order of arguments of an operator is immaterial. In the setting of process algebras, commutativity is defined with respect to a notion of behavioural equivalence over terms. In this section, we first present a generalized notion of commutativity that allows n-ary operators to have various sets of commutative arguments and then slightly adapt the commutativity rule format proposed in [16] to the extended setting. Moreover, we give some auxiliary definitions that will be used in the axiomatization procedure proposed in the next section.

In order to motivate the generalized notion of commutativity we present below, consider, by way of example, the ternary operator f defined by the rules below, where a ranges over the collection of action labels L.

$$\frac{x \xrightarrow{a} x'}{f(x,y,z) \xrightarrow{a} f(x',y,z)} \quad \frac{y \xrightarrow{a} y'}{f(x,y,z) \xrightarrow{a} f(x,y',z)}$$
$$\frac{x \xrightarrow{a} x' \quad z \xrightarrow{a} z'}{f(x,y,z) \xrightarrow{a} f(x',y,z')} \quad \frac{y \xrightarrow{a} y' \quad z \xrightarrow{a} z'}{f(x,y,z) \xrightarrow{a} f(x,y',z')}.$$

It is not hard to show that the operator f is commutative in its first two arguments modulo bisimilarity, irrespective of the other operators in the TSS under consideration—that is, $f(p, q, r) \leftrightarrow f(q, p, r)$, for all closed terms p, q, r. On the other hand, the third argument does not commute with respect to the other two. For example, we have that $f(a.\mathbf{0}, \mathbf{0}, \mathbf{0}) \not\leftrightarrow f(\mathbf{0}, \mathbf{0}, a.\mathbf{0})$ because $f(a.\mathbf{0}, \mathbf{0}, \mathbf{0}) \xrightarrow{a} f(\mathbf{0}, \mathbf{0}, \mathbf{0})$, but $f(\mathbf{0}, \mathbf{0}, a.\mathbf{0})$ has no outgoing transitions.

The commutativity format presented in [16] can only deal with operators that are commutative for each pair of arguments and, unlike the format that we present below, is therefore unable to detect that f is commutative in its first two arguments.

In what follows, we shall often use $[n]$, $n \geq 0$, to stand for the set $\{1, \ldots, n\}$. Note that $[0]$ is just the empty set.

Definition 5 (Generalized Commutativity). *Given a set I, a family $\prod_I$ of non-empty, pairwise disjoint subsets of I is called a* partition *of I when $\bigcup \prod_I = I$.*

*Let Σ be a signature. Assume that $f \in \Sigma$ is an n-ary operator, $\prod_{[n]}$ is a partition of $[n]$ and $\sim$ is an equivalence relation over $\mathbb{C}(\Sigma)$. The operator f is called $\prod_{[n]}$-*commutative with respect to $\sim$ *when, for each $K \in \prod_{[n]}$ and each two $j, k \in K$ such that $j < k$, the following equation is sound with respect to $\sim$:*

$$f(x_1, \ldots, x_n) = f(x_1, \ldots, x_{j-1}, x_k, x_{j+1}, \ldots, x_{k-1}, x_j, x_{k+1}, \ldots, x_n).$$

Note that the traditional notion of commutativity for binary operators can be recovered using Definition 5 in terms of $\{\{1, 2\}\}$-commutativity. Moreover, the notion of commutativity for n-ary operators from [16] corresponds to $\{[n]\}$-commutativity. Any n-ary operator is $1_{[n]}$-commutative with respect to any equivalence relation $\sim$, where $1_{[n]} = \{\{1\}, \ldots, \{n\}\}$ is the discrete partition of $[n]$.

From this point onward, whenever a signature Σ is provided, we also assume that every function symbol $f \in \Sigma$ of arity n has an associated fixed partition of its set of arguments $[n]$ denoted by $\prod_f$. We denote the indexed set of all these partitions by $\prod^{\Sigma} = \{\prod_f\}_{f \in \Sigma}$.

Definition 6. *Assume $\Sigma_1 \subseteq \Sigma_2$. Let $\prod^{\Sigma_1}$ be a family of partitions. The* extension of $\prod^{\Sigma_1}$ to Σ_2 *is obtained by taking $\prod_f$ to be the discrete partition over $[n]$ for each $f \in \Sigma_2 \setminus \Sigma_1$, where n is the arity of f.*

Our aim is to define a restriction of the GSOS rule format that guarantees the notion of generalized commutativity defined above for any behavioural equivalence that is coarser than bisimilarity. To this end, we begin by extending the notion of commutative congruence introduced in [16] to the context of this generalized notion of commutativity.

Definition 7 (Commutative Congruence). *Consider a signature Σ and a set of partitions $\prod^{\Sigma}$. The commutative congruence relation $\sim_{cc}$ (with respect to $\prod^{\Sigma}$) is the least congruence relation over $\mathbb{T}(\Sigma)$ satisfying the following requirement: for all $f \in \Sigma$ (of arity n), $K \in \prod_f$, $j, k \in K$ with $j < k$, and $t_1, \ldots, t_n \in \mathbb{T}(\Sigma)$, it holds that*

$$f(t_1, \ldots, t_n) \sim_{cc} f(t_1, \ldots, t_{j-1}, t_k, t_{j+1}, \ldots, t_{k-1}, t_j, t_{k+1}, \ldots, t_n).$$

We are now ready to present a syntactic restriction on the GSOS format that guarantees commutativity with respect to a set of partitions $\prod^{\Sigma}$ modulo any notion of behavioural equivalence that includes strong bisimilarity. Unlike the format for $\{[n]\}$-commutativity given in [16], the format offered below applies to generalized commutativity, in the sense of Definition 5, and is defined for TSSs whose rules can have negative premises. On the other hand, unlike ours, the format introduced in [16] applies to rules whose positive premises need not have variables as their sources and targets. Extending our format in order to accommodate this kind of premises in deduction rules is straightforward, but is not relevant for the purpose of this paper.

Definition 8 (Comm-GSOS). *A transition system specification over signature Σ is in the comm-GSOS format with respect to a set of partitions $\prod^{\Sigma}$ if it is in the GSOS format and for each f-defining deduction rule $d = \dfrac{H}{f(x_1,\ldots,x_n)\xrightarrow{l} t}$, each $K \in \prod_f$ and for all $j,k \in K$ with $j < k$, there exist a deduction rule $d' = \dfrac{H'}{f(x'_1,\ldots,x'_n)\xrightarrow{l} t'}$ and a bijective mapping $\hbar$ over variables such that*

- $\hbar(x'_i) = x_i$ *for each* $i \in [n]$ *such that* $i \neq j$ *and* $i \neq k$,
- $\hbar(x'_j) = x_k$ *and* $\hbar(x'_k) = x_j$,
- $\hbar(t') \sim_{cc} t$, *and*
- $\hbar(H') = H$.

Deduction rule d' is called a commutative mirror *of d (with respect to j, k and $\prod^{\Sigma}$).*

The above format requires that, when $f \in \Sigma$, for each f-defining rule and for each pair (j,k) of arguments for which f is supposed to be commutative, as specified by $\prod_f$, there exists a commutative mirror that enables the 'same transitions up to the commutative congruence $\sim_{cc}$ associated with $\prod^{\Sigma}$' when the jth and kth arguments of f are swapped. This is the essence of the proof of the following theorem, which states the correctness of the syntactic comm-GSOS format.

Theorem 1. *If a transition system specification is in the comm-GSOS format with respect to a set of partitions $\prod^{\Sigma}$, then each operator $f \in \Sigma$ is $\prod_f$-commutative with respect to any notion of behavioural equivalence that includes bisimilarity.*

Example 1. Consider the ternary operator f we used earlier to motivate the notion of generalized commutativity. Any transition system specification including the operator f is in the comm-GSOS format with respect to any set of partitions $\prod^{\Sigma}$ such that $\prod_f = \{\{1,2\},\{3\}\}$. Indeed, the a-emitting rules in the first row are one the commutative mirror of the other with respect to $\prod^{\Sigma}$, and so are those in the second row. The constraints in Definition 8 are vacuously satisfied when we take $K = \{3\}$. Therefore, by Theorem 1, we recover the fact that f is commutative in its first two arguments.

Example 2 (Parallel Composition). A frequently occurring commutative operator is parallel composition. It appears in, amongst others, ACP [9], CCS [15], and CSP [14]. Here we discuss parallel composition with communication in the style of ACP [9], of which the others are special cases. The rules for this operator are listed below. In those rules, a, b, c range over L and $\gamma : L \times L \hookrightarrow L$ is a partial communication function.

$$(\mathrm{p}_1)\ \frac{x \xrightarrow{a} x'}{x \,||\, y \xrightarrow{a} x' \,||\, y} \qquad (\mathrm{p}_2)\ \frac{y \xrightarrow{a} y'}{x \,||\, y \xrightarrow{a} x \,||\, y'} \qquad (\mathrm{p}_3)\ \frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y'}{x \,||\, y \xrightarrow{c} x' \,||\, y'} \quad \gamma(a,b) = c$$

If the partial communication function γ is commutative, then any GSOS system including the operator $||$ given by the above rules is in the comm-GSOS format with respect to any set of partitions $\prod^{\Sigma}$ such that $\prod_{||} = \{\{1,2\}\}$. Hence it follows from Theorem 1 that $||$ is $\{\{1,2\}\}$-commutative.

4 Mechanized Axiomatization

In this section, we present a technique for the automatic generation of ground-complete axiomatizations of bisimilarity over TSSs in the comm-GSOS format, which is derived from the one introduced in [1]. Our approach improves upon the one in [1] by making use of the rule format for generalized commutativity we introduced in the previous section. Our goal is to generate a disjoint extension of the original TSS and a finite axiom system that is sound and ground complete for bisimilarity over it. This finite axiom system may then also be used for equationally establishing bisimilarity over closed terms from the original TSS. We start by axiomatizing a rather restrictive subset of 'good' operators in Section 4.1. Then we turn 'bad' operators into good ones by means of auxiliary operators. In both of these steps, we exploit commutativity information, where possible, in order to reduce the number of generated axioms, as well as the number of generated auxiliary operators.

4.1 Axiomatizing Good Operators

The approach offered in [1] relies on the fact that the signature includes the operators from BCCSP. (Recall that, in keeping with [10], we assume that these operators are present in any TSS in the GSOS format.) The aim of the axiomatization procedure is then to generate an axiom system that can rewrite any closed term p into a term p' in head normal form such that $p \leftrightarrow p'$. (We call an axiom system with this property *head normalizing*.) Recall that a term t is in *head normal form* if it has the form $a_1.t_1 + \cdots + a_n.t_n$ for some $n \geq 0$, some set of actions $\{a_i \mid i \in [n]\}$ and set of terms $\{t_i \mid i \in [n]\}$. If $n = 0$ then $a_1.t_1 + \cdots + a_n.t_n$ stands for $\mathbf{0}$.

For 'semantically well founded' terms (see [1, Definition 5.1 on page 28]), rewriting into head normal form can be used to prove that each closed term is equal to a closed term over the signature for BCCSP. This leads to a ground-complete axiomatization of bisimilarity, since BCCSP is finitely axiomatized modulo bisimilarity by the axiom system E_{BCCSP} from [13] consisting of the axioms stating that '$+$' is associative, commutative, idempotent and has $\mathbf{0}$ as unit element.

To start with, we focus on the case of closed terms built using only *good* operators, which we now proceed to define.

Definition 9 (Smooth and distinctive operator). *Consider an n-ary operator f.*

1. *A* smooth GSOS deduction rule *is of the form*

$$\frac{\{x_i \xrightarrow{a_i} y_i \mid i \in I\} \cup \{x_i \stackrel{b_{ij}}{\nrightarrow} \mid i \in J, 1 \leq j \leq n_i\}}{f(x_1, \ldots, x_n) \xrightarrow{c} C[\overrightarrow{x}, \overrightarrow{y}]}$$

where I and J are disjoint subsets of $[n]$ such that $I \cup J = [n]$, and $C[\overrightarrow{x}, \overrightarrow{y}]$ can only include the variables x_i $(i \in [n] \setminus I)$ and y_i $(i \in I)$.
An operator f of a TSS in the GSOS format is smooth *if all its rules are smooth.*

2. *An n-ary operator f of a TSS in the GSOS format is* distinctive *if it is smooth, each f-defining rule tests the same set of arguments I positively, and for every two distinct f-defining rules there is some argument tested positively by both rules, but with a different action.*

We refer the interested reader to [1, Section 4.1] for an in-depth discussion of the constraints for smooth and distinctive operators.

Remark 2. The ternary operator f from Example 1 and the parallel composition operator from Example 2 are smooth but not distinctive. On the other hand, the classic communication merge operator [6, 8], given by the rules $\frac{x \xrightarrow{a} x' \quad y \xrightarrow{b} y'}{x \mid y \xrightarrow{c} x' \,||\, y'}$ $(\gamma(a, b) = c)$, is smooth and distinctive. Moreover, assuming that γ is commutative, any TSS whose signature Σ includes $||$ and $\mid$ with the previously given rules is in the comm-GSOS format with respect to any set of partitions $\prod^{\Sigma}$ such that $\prod_{\mid} = \prod_{||} = \{\{1, 2\}\}$.

Definition 10 (Discarding and Good Operators). *A smooth GSOS rule of the form given in Definition 9 is* discarding *if none of the variables x_i with $i \in J$ and $n_i > 0$ occurs in $C[\overrightarrow{x}, \overrightarrow{y}]$. A smooth operator is* discarding *if so are all the rules for it. A smooth operator is* good *[11] if it is both distinctive and discarding.*

In the remainder of this subsection, we assume that the GSOS system $\mathcal{T}$ has signature Σ and is in the comm-GSOS format with respect to a set of partitions $\prod^{\Sigma}$. Let $f \in \Sigma$ be a good operator that is not in the signature for BCCSP, and let n be its arity. Our goal is to generate an axiom system that can be used to turn any term of the form $f(t_1, \ldots, t_n)$, where the t_i's are in head normal form, into a head normal form. In the generation of the axiom system, we will exploit the commutativity information that is provided by the partition $\prod_f$ and therefore we assume that $n \geq 2$. (If f is either a constant or a unary operator, then it will be axiomatized exactly as in [1], since commutativity information is immaterial.) Let $I_f \subseteq [n]$ be the set of arguments that are tested positively by f and let J_f be the complement of I_f. Assume that $\prod_f = \{K_1, \ldots, K_\ell\}$. Since $\mathcal{T}$ is in the comm-GSOS format with respect to $\prod^{\Sigma}$, and f is smooth and distinctive, it is not hard to see that $K_h \subseteq I_f$ or $K_h \subseteq J_f$, for each $h \in [\ell]$. Indeed exactly one of the above inclusions holds. Let $\prod_f^+ = \{K \mid K \in \prod_f \text{ and } K \subseteq I_f\}$ and $\prod_f^- = \{K \mid K \in \prod_f \text{ and } K \subseteq J_f\}$. We use K_f^+ (respectively, K_f^-) to denote a subset of I_f (respectively, J_f) that results by choosing exactly one representative element for each $K \in \prod_f^+$ (respectively, $K \in \prod_f^-$).

Example 3. Consider the communication merge operator $|$ from Remark 2. We already remarked that, when the communication function γ is commutative, the rules for $|$ are in the comm-GSOS format with respect to any set of partitions $\prod^{\Sigma}$ such that $\prod_{|} = \prod_{||} = \{\{1,2\}\}$. For the operator $|$, we may take $K_{|}^{+} = \{1\}$. Since the rules for $|$ have no negative premises, $K_{|}^{-}$ is empty.

Definition 11. *Let f be a good n-ary operator, and let K_f^+ and K_f^- be defined as above. We associate with f the finite axiom system E_f consisting of the following equations.*

1. Distributivity laws*: For each $i \in K_f^+$, we have the equation:*

$$f(x_1,\ldots,x_i+x_i',\ldots,x_n) = f(x_1,\ldots,x_i,\ldots,x_n) + f(x_1,\ldots,x_i',\ldots,x_n).$$

2. Peeling laws*: For each rule for f of the form given in Definition 9, each $k \in K_f^-$ with $n_k > 0$ and each $a \notin \{b_{kj} \mid 1 \le j \le n_k\}$, we have the equation:*

$$f(P_1,\ldots,P_n) = f(Q_1,\ldots,Q_n),$$

where $P_i \equiv \begin{cases} a_i.y_i & i \in I \\ a.x_k' + x_k'' & i = k \\ x_i & \textit{otherwise} \end{cases}$ *and* $Q_i \equiv \begin{cases} a_i.y_i & i \in I \\ x_k'' & i = k \\ x_i & \textit{otherwise}. \end{cases}$

3. Action laws*: For each rule for f of the form given in Definition 9, we have the equation: $f(P_1,\ldots,P_n) = c.C[\overrightarrow{P},\overrightarrow{y}]$, where* $P_i \equiv \begin{cases} a_i.y_i & i \in I \\ \mathbf{0} & i \in J \textit{ and } n_i > 0 \\ x_i & \textit{otherwise}. \end{cases}$

4. Inaction laws*: For each $i \in K_f^+$, we have the equation*

$$f(x_1,\ldots,x_{i-1},\mathbf{0},x_{i+1},\ldots,x_n) = \mathbf{0}.$$

Suppose that, for each $i \in [n]$, term P_i is of the form $a.z_i$ when $i \in I_f$, and of the form $a.z_i + z_i'$ or z_i when $i \in J_f$. Suppose further that, for each rule for f of the form given in Definition 9, there exists some $i \in [n]$ such that one of the following holds:
   - *$i \in I_f$ and ($P_i \equiv a.z_i$, for some $a \neq a_i$),*
   - *$i \in J_f$ and ($P_i \equiv b_{ij}.z_i + z_i'$, for some $1 \le j \le n_i$).*

   Then we have the equation $f(P_1,\ldots,P_n) = \mathbf{0}$.

5. Commutativity laws*: For each equivalence class $K \in \prod_f$ and each two $i,j \in K$ such that $i < j$, we have the equation:*

$$f(x_1,\ldots,x_i,\ldots,x_j,\ldots,x_n) = f(x_1,\ldots,x_j,\ldots,x_i,\ldots,x_n).$$

Theorem 2. *Consider a TSS $\mathcal{T}$ in GSOS format. Let Σ_g be a collection of good operators of $\mathcal{T}$. Let E_{Σ_g} be the finite axiom system that consists of the axioms in E_{BCCSP} and the axioms in E_f, for each $f \in \Sigma_g$. Then, for any GSOS system $\mathcal{T}'$ such that $\mathcal{T} \sqsubseteq \mathcal{T}'$, the axiom system E_{Σ_g} is sound and is ground complete for terms built using the operations in the signature Σ_g and those in the signature of BCCSP.*

Example 4. For the communication merge operator $|$, taking $K_{|}^{+} = \{1\}$ as in Example 3, Definition 11 yields the following axiom system $E_{|}$:

$$\begin{array}{ll}
\text{distributivity:} & (x+y) \mid z = (x \mid z) + (y \mid z), \\
\text{action:} & a.x \mid b.y = c.(x \,||\, y) \quad \text{if } \gamma(a,b) = c, \\
\text{inaction:} & \mathbf{0} \mid y = \mathbf{0}, \\
\text{inaction:} & a.x \mid b.y = \mathbf{0} \quad \text{if } \gamma(a,b) \text{ is undefined}, \\
\text{commutativity:} & x \mid y = y \mid x.
\end{array}$$

These are exactly the equations describing the interplay between the operator $|$ and the BCCSP operators given in Table 7.1 on page 204 of [6].

4.2 Turning Bad into Good

In order to handle arbitrary GSOS operators, one needs two additional procedures: one for transforming non-smooth operators into smooth and discarding (but not necessarily distinctive) operators, and one for expressing smooth, discarding and non-distinctive operators in terms of good operators. We adopt the same approach for the first procedure as the one presented in Lemma 4.13 in [1]. On the other hand, for the second procedure, we improve on the algorithm derived from Lemma 4.10 in [1].

The step from smooth, discarding and non-distinctive operators to good ones involves the synthesis of several new operators. We now show how to improve this transformation, as presented in the aforementioned reference, by reducing the number of the generated auxiliary operators, making use of the ideas underlying the generalized commutativity format presented in Section 3.

Making Smooth and Discarding Operators Distinctive. Consider a TSS $\mathcal{T}$ with signature Σ in the comm-GSOS format with respect to a set of partitions $\prod^{\Sigma}$. Let $f \in \Sigma$ be a smooth and discarding, but not distinctive operator, and let n be its arity. We will now show how to express f in terms of good operators. We start with partitioning the set of f-defining rules into sets $R_1, \ldots, R_m$, $m > 1$, such that f is distinctive when its rules are restricted to those in R_i for each $i \in [m]$. Note that all the rules in each R_i test the same arguments positively. If $\prod_f$ is the discrete partition over $[n]$ then one proceeds by axiomatizing f as in the version of the original algorithm based on the so-called peeling laws presented in [1]. Indeed, in that case, f has no pair of commutative arguments. Suppose therefore that $\prod_f$ is not the discrete partition, and take some $K \in \prod_f$ of maximum cardinality. (Any non-singleton K would do in what follows. However, picking a set K of maximum cardinality will reduce the number of auxiliary operators that is generated by the procedure outlined below.) Our aim now is to define when two sets of rules for f are 'essentially the same up to the commutative arguments in K' and to use this information in order to synthesize enough good operators for expressing f up to bisimilarity.

Definition 12.

– *Let d and d' be two f-defining and l-emitting rules. We say that d' is a* commutative mirror *of d with respect to K and $\prod^{\Sigma}$ if the constraints in Definition 8 are met for* *some $j, k \in K$ with $j < k$. We use $\overset{K}{\smile}$ to denote the reflexive and transitive closure of the relation 'is a commutative mirror with respect to K'.*

- *Let R and R' be two sets of f-defining rules. We write $R \overset{K}{\smile} R'$ if, and only if, (1) for each $d \in R$ there is some $d' \in R'$ such that $d \overset{K}{\smile} d'$, and (2) for each $d' \in R'$ there is some $d \in R$ such that $d \overset{K}{\smile} d'$.*

Example 5. Consider the ternary operator f defined by the rules on page 4. That operator is smooth and discarding, but not distinctive. Collecting all the rules that test the same arguments positively in the same set, we obtain the following four sets of rules:

- R_1 contains all the rules of the form $\dfrac{x \xrightarrow{a} x'}{f(x,y,z) \xrightarrow{a} f(x',y,z)}$ $(a \in L)$.
- R_2 contains all the rules of the form $\dfrac{y \xrightarrow{a} y'}{f(x,y,z) \xrightarrow{a} f(x,y',z)}$ $(a \in L)$.
- R_3 contains all the rules of the form $\dfrac{x \xrightarrow{a} x',\ z \xrightarrow{a} z'}{f(x,y,z) \xrightarrow{a} f(x',y,z')}$ $(a \in L)$.
- R_4 contains all the rules of the form $\dfrac{y \xrightarrow{a} y',\ z \xrightarrow{a} z'}{f(x,y,z) \xrightarrow{a} f(x,y',z')}$ $(a \in L)$.

We have already seen in Example 1 that any GSOS system including the operator f is in the comm-GSOS format with respect to any set of partitions $\prod^{\Sigma}$ such that $\prod_f = \{\{1,2\},\{3\}\}$. Take $K = \{1,2\}$. It is not hard to see that $R_1 \overset{K}{\smile} R_2$ and $R_3 \overset{K}{\smile} R_4$ hold. Indeed, as we observed in Example 1, each a-emitting rule in R_1 (respectively, R_3) is the commutative mirror of the a-emitting rule in R_2 (respectively, R_4) with respect to K, and vice versa.

Lemma 1. *$\overset{K}{\smile}$ is an equivalence relation over f-defining rules and over sets of f-defining rules.*

Recall that $\{R_1, \ldots, R_m\}$, $m > 1$, is a partition of the set of f-defining rules such that f is distinctive when its rules are restricted to those in R_i for each $i \in [m]$. Consider $\{R_1, \ldots, R_m\}/\overset{K}{\smile}$, the quotient of the set $\{R_1, \ldots, R_m\}$ with respect to the equivalence relation $\overset{K}{\smile}$. Let $\rho_1, \ldots, \rho_\ell$ be representatives of its equivalence classes. For example, in the case of the operator considered in Example 5 above, one could pick R_1 and R_4, say, as representatives of the two equivalence classes with respect to $\overset{\{1,2\}}{\smile}$. We proceed by adding to the signature Σ fresh n-ary operator symbols $f_1, \ldots, f_\ell$. The rules for the operator f_i are obtained by simply turning those in ρ_i into f_i-defining ones. Let $\mathcal{T}'$ be the resulting disjoint extension of $\mathcal{T}$. Following [1], we now need to generate an axiom that expresses f in terms of $f_1, \ldots, f_\ell$.

Definition 13. *Let $n > 0$ and let $K \subseteq [n]$. A bijection $\pi : [n] \to [n]$ is a K-*permutation *if it is the identity function over $[n] \setminus K$.*

The equation relating f to the f_i's, $i \in [\ell]$, can now be stated as follows:

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{\ell} \sum \{ f_i(x_{\pi(1)}, \ldots, x_{\pi(n)}) \mid \pi \text{ is a } K\text{-permutation} \}. \qquad (1)$$

For our running example, namely the ternary operator f defined by the rules on page 4 and considered in Examples 1 and 5, with the choice of representatives mentioned above, there are two auxiliary operators f_1 and f_2 with rules $\dfrac{x \xrightarrow{a} x'}{f_1(x,y,z) \xrightarrow{a} f(x',y,z)}$ $\dfrac{y \xrightarrow{a} y',\ z \xrightarrow{a} z'}{f_2(x,y,z) \xrightarrow{a} f(x,y',z')}$, where a ranges over L. Apart from the identity function over [3], the only $\{1,2\}$-permutation is the one that swaps 1 and 2. Therefore, instantiating equation (1), we obtain that

$$f(x_1,x_2,x_3) = f_1(x_1,x_2,x_3) + f_1(x_2,x_1,x_3) + f_2(x_1,x_2,x_3) + f_2(x_2,x_1,x_3).$$

Using Definition 6, the family of partitions $\prod^{\Sigma}$ can be extended to any signature that includes the signature $\Sigma \cup \{f_i \mid i \in [\ell]\}$. Note that any disjoint extension of $\mathcal{T}'$ is in the comm-GSOS format with respect to this extension of $\prod^{\Sigma}$.

Proposition 2. *Equation (1) is sound in any disjoint extension of $\mathcal{T}'$.*

Equation (1) can be simplified in case any of the auxiliary operators $f_1, \ldots, f_\ell$ is commutative in the set of arguments K. Indeed, let $N \subseteq [\ell]$, and assume that $\mathcal{T}'$ is in the comm-GSOS format with respect to the family of partitions that associates with each operator g the partition $\prod_g^{\Sigma}$ when $g \in \Sigma$, the partition $\{K\} \cup 1_{[n]\setminus K}$ when $g \in \{f_i \mid i \in N\}$, and the partition $1_{[n]}$ otherwise. Then we have the following result.

Proposition 3. *The following equation is sound in any disjoint extension of $\mathcal{T}'$.*

$$\begin{aligned} f(x_1,\ldots,x_n) = &\sum_{i \in [\ell]\setminus N} \sum \{f_i(x_{\pi(1)},\ldots,x_{\pi(n)}) \mid \pi \text{ is a } K\text{-permutation}\} \ + \\ &\sum_{i \in N} f_i(x_1,\ldots,x_n) \end{aligned} \tag{2}$$

In the following section, we will see that the above simplification leads to an axiomatization of the classic parallel composition operator that is equal to an existing handcrafted one. Of course, if either N or $[\ell] \setminus N$ are empty, the corresponding $\mathbf{0}$ summand can be omitted in equation (2).

Turning Non-Smooth Operators into Smooth Ones. The methods we have presented so far yield an algorithm that, given a TSS $\mathcal{T}$ with signature Σ in the comm-GSOS format with respect to a set of partitions $\prod^{\Sigma}$, can be used to generate a disjoint extension $\mathcal{T}'$ of $\mathcal{T}$ over some signature Σ' that includes Σ and a finite axiom system *E* such that *E* is sound modulo bisimilarity over any disjoint extension of $\mathcal{T}'$ and is head normalizing for all closed Σ'-terms. Ground-completeness of *E* with respect to bisimilarity over $\mathcal{T}'$ (and therefore over $\mathcal{T}$) follows using standard reasoning, by possibly using the well-known Approximation Induction Principle [8] if $\mathcal{T}'$ is not semantically well founded. See [1] for details.

The algorithm has the following steps:

1. Start with the axiom system E_{BCCSP} and consider next the operators that are not in the signature for BCCSP.
2. For each non-smooth operator $f \in \Sigma$, generate a fresh smooth and discarding operator f', and add to the axiom system the equation expressing f in terms of f' as in Lemma 4.13 in [1].
3. For each smooth and discarding, but not distinctive, operator f in the resulting signature, generate a family of fresh good operators $f_1, \dots f_\ell$, as indicated in this section, and add to the axiom system the instance of equation (1) or of equation (2), as appropriate, expressing f in terms of $f_1, \dots f_\ell$.
4. For each good operator in the resulting signature, add to the axiom system the equations mentioned in the statement of Theorem 2.

5 Axiomatizing Parallel Composition

Let us concretely analyze the axiomatization derived using the procedure described above for the classic parallel composition operator $||$ from Example 2. We assume henceforth that the partial synchronization function γ is commutative, so that $||$ is $\{\{1,2\}\}$-commutative. As we observed in Remark 2, the parallel composition operator is smooth but not distinctive. When we partition the set of rules for $||$ into subsets of rules that test the same arguments positively, we obtain three sets R_1, R_2 and R_3, where each R_i consists of all the instances of rule (p_i) from Example 2. It is easy to see that $R_1 \overset{\{1,2\}}{\smile} R_2$. Therefore, following the procedure described in Section 4.2, we can generate two auxiliary binary operators, which are the classic left merge and communication operators, denoted by $\lfloor\!\lfloor$ and $|$, respectively. The rules for $|$ are those in Remark 2 and those for the left merge operator are $\dfrac{x \xrightarrow{a} x'}{x \lfloor\!\lfloor y \xrightarrow{a} x' \,||\, y}$ $(a \in L)$. Since we know that $|$ is $\{\{1,2\}\}$-commutative, the relationship between $||$ and the two auxiliary operators can be expressed using equation (2), whose relevant instance becomes

$$x \,||\, y = (x \lfloor\!\lfloor y) + (y \lfloor\!\lfloor x) + (x \mid y).$$

This is exactly equation M in Table 7.1 on page 204 of [6]. The axioms for $|$ produced by our methods are those given in Example 4. On the other hand, the left merge operator is axiomatized as in [1] since commutativity information is immaterial for it.

In Figure 1 we compare the axiomatization for the parallel composition operator $||$ derived using the algorithm from [1] and the 'optimized axiomatization' one obtains using the algorithm mentioned above. (We omit the four equations in the axiom system E_{BCCSP} recalled in Section 4.) The axioms generated by the algorithm from [1] do resemble the original axioms of [9] to a large extent. The auxiliary operator $\rfloor\!\rfloor$ is called right merge in the literature.

6 Conclusions and Future Work

In this paper, we have taken a first step towards marrying two lines of development within the field of the meta-theory of SOS, viz. the study of algorithms for the auto-

Standard	Optimized
$x \,\|\, y = (x \mathbin{\|\!\lfloor} y) + (x \mathbin{\rfloor\!\|} y) + (x \mid y)$	$x \,\|\, y = (x \mathbin{\|\!\lfloor} y) + (y \mathbin{\|\!\lfloor} x) + (x \mid y)$
$(a.x) \mathbin{\|\!\lfloor} y = a.(x \,\|\, y)$	$(a.x) \mathbin{\|\!\lfloor} y = a.(x \,\|\, y)$
$x \mathbin{\rfloor\!\|} (a.y) = a.(x \,\|\, y)$	$(a.x) \mid (b.y) = c.(x \,\|\, y) \quad \text{if } \gamma(a,b) = c$
$(a.x) \mid (b.y) = c.(x \,\|\, y) \quad \text{if } \gamma(a,b) = c$	$(x+y) \mathbin{\|\!\lfloor} z = (x \mathbin{\|\!\lfloor} z) + (y \mathbin{\|\!\lfloor} z)$
$(x+y) \mathbin{\|\!\lfloor} z = (x \mathbin{\|\!\lfloor} z) + (y \mathbin{\|\!\lfloor} z)$	$(x+y) \mid z = (x \mid z) + (y \mid z)$
$x \mathbin{\rfloor\!\|} (y+z) = (x \mathbin{\rfloor\!\|} y) + (x \mathbin{\rfloor\!\|} z)$	
$(x+y) \mid z = (x \mid z) + (y \mid z)$	
$x \mid (y+z) = (x \mid y) + (x \mid z)$	$\mathbf{0} \mathbin{\|\!\lfloor} x = \mathbf{0}$
$\mathbf{0} \mathbin{\|\!\lfloor} x = \mathbf{0} \quad x \mathbin{\rfloor\!\|} \mathbf{0} = \mathbf{0}$	$\mathbf{0} \mid x = \mathbf{0}$
$\mathbf{0} \mid x = \mathbf{0} \quad x \mid \mathbf{0} = \mathbf{0}$	$(a.x) \mid (b.y) = \mathbf{0} \quad \text{if } \gamma(a,b) \text{ is undefined}$
$(a.x) \mid (b.y) = \mathbf{0} \quad \text{if } \gamma(a,b) \text{ is undefined}$	$x \mid y = y \mid x$

Fig. 1. Axiomatizing $\|$

matic generation of ground-complete axiomatizations for bisimilarity from SOS specifications (see, for instance, [1, 7, 20]) and the development of rule formats guaranteeing the validity of algebraic laws, such as those surveyed in [5]. More specifically, we have presented a rule format for commutativity that refines the one offered in [16] in that it allows one to consider various sets of commutative arguments, and we have used the information provided by that rule format to refine the algorithm for the automatic generation of ground-complete axiomatizations for bisimilarity from [1]. The resulting procedure yields axiom systems that use fewer auxiliary operators to axiomatize commutative operators than the one from [1]. Moreover, in some important cases, the mechanically produced axiomatizations of some operators are identical to the hand-crafted ones from the literature.

The ideas we have presented in this paper have never been explored before, and they enrich the toolbox one can use when reasoning about bisimilarity by means of axiomatizations. Moreover, the combination of two closely related strands of research on the meta-theory of SOS we have begun in this paper is of theoretical interest and may lead to further improvements on algorithms for the automatic generation of axiomatic characterizations of bisimilarity. As future work, we will implement the axiomatization procedure presented in this paper in the PREG Axiomatizer tool [2]. We also intend to explore the use of other rule formats for algebraic properties in improving mechanized axiomatizations for bisimilarity. The ultimate goals of this research are to make automatically generated axiomatizations comparable to the known ones from the literature and to make the first steps towards the automatic generation of axiomatizations that are complete for open terms. The latter goal is a very ambitious one since obtaining complete axiomatizations of bisimilarity is a very hard research problem even for sufficiently rich fragments of specific process calculi; see, for instance, [3].

References

1. L. Aceto, B. Bloom, and F. W. Vaandrager. Turning SOS rules into equations. *Information and Computation*, 111:1–52, 1994.

2. L. Aceto, G. Caltais, E.-I. Goriac, and A. Ingólfsdóttir. PREG Axiomatizer - a ground bisimilarity checker for GSOS with predicates. In A. Corradini, B. Klin, and C. Cîrstea, eds.: *Algebra and Coalgebra in Computer Science - 4th International Conference, CALCO 2011, Winchester, UK, August 30-September 2, 2011. Proceedings*, volume 6859 of *Lecture Notes in Computer Science*, pages 378–385. Springer, 2011.
3. L. Aceto, W. Fokkink, A. Ingólfsdóttir, and B. Luttik. Finite equational bases in process algebra: Results and open questions. In A. Middeldorp, V. van Oostrom, F. van Raamsdonk, and R. C. de Vrijer, eds.: *Processes, Terms and Cycles: Steps on the Road to Infinity, Essays Dedicated to Jan Willem Klop, on the Occasion of His 60th Birthday*, volume 3838 of *Lecture Notes in Computer Science*, pages 338–367. Springer, 2005.
4. L. Aceto, W. J. W. Fokkink, and C. Verhoef. Structural operational semantics. In J. A. Bergstra, A. Ponse, and S. A. Smolka, eds.: *Handbook of Process Algebra, Chapter 3*, pages 197–292. Elsevier Science, Dordrecht, The Netherlands, 2001.
5. L. Aceto, A. Ingolfsdottir, M. Mousavi, and M. A. Reniers. Algebraic properties for free! *Bulletin of the European Association for Theoretical Computer Science*, 99:81–104, 2009.
6. J. Baeten, T. Basten, and M. Reniers. *Process Algebra: Equational Theories of Communicating Processes*, volume 50 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2009.
7. J. J. Baeten and E. P. de Vink. Axiomatizing GSOS with termination. *Journal of Logic and Algebraic Programming*, 60-61:323–351, 2004.
8. J. A. Bergstra and J. W. Klop. Fixedpoint semantics in process algebra. Technical Report IW 206/82, Center for Mathematics, Amsterdam, The Netherlands, 1982.
9. J. A. Bergstra and J. W. Klop. Process algebra for synchronous communication. *Information and Control*, 60(1-3):109–137, 1984.
10. B. Bloom, S. Istrail, and A. R. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268, Jan. 1995.
11. D. Bosscher. Term rewriting properties of SOS axiomatisations. In M. Hagiya and J. C. Mitchell, eds.: *Theoretical Aspects of Computer Software, International Conference TACS '94, Sendai, Japan, April 19–22, 1994, Proceedings*, volume 789 of *Lecture Notes in Computer Science*, pages 425–439. Springer, 1994.
12. R. J. van Glabbeek. The linear time - branching time spectrum I. In Bergstra, J.A., Ponse, A., Smolka, S.A., eds.: *Handbook of Process Algebra, Chapter 1*, pages 3–100. Elsevier Science, Dordrecht, The Netherlands, 2001.
13. M. Hennessy and A. R. Milner. Algebraic laws for non-determinism and concurrency. *Journal of the ACM*, 32(1):137–161, 1985.
14. C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.
15. A. R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.
16. M. Mousavi, M. Reniers, and J. F. Groote. A syntactic commutativity format for SOS. *Information Processing Letters*, 93:217–223, Mar. 2005.
17. M. Mousavi, M. A. Reniers, and J. F. Groote. SOS formats and meta-theory: 20 years after. *Theoretical Computer Science*, 373:238–272, 2007.
18. D. M. Park. Concurrency and automata on infinite sequences. In Duessen, P., ed.: Proceedings of the 5th GI Conference. Volume 104 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, Germany (2001) 167–183
19. G. D. Plotkin. A structural approach to operational semantics. *Journal of Logic and Algebraic Progamming*, 60:17–139, 2004.
20. I. Ulidowski. Finite axiom systems for testing preorder and De Simone process languages. *Theoretical Computer Science*, 239(1):97–139, 2000.