Axiomatizing GSOS with termination

TU/e technische universiteit eindhoven

CS-Report 01-06

Aximatizing GSOS with Termination

J.C.M. Baeten
E.P. de Vink

/ department of mathematics and computing science

Technische Universiteit Eindhoven
Department of Mathematics and Computer Science

Aximatizing GSOS with termination

by

J.C.M. Baeten and E.P. de Vink

01/06

ISSN 0926-4515

Reports are available at:
http://www.win.tue.nl/win/cs

Computer Science Reports 01/06
Eindhoven, October 2001

AXIOMATIZING GSOS WITH TERMINATION

J.C.M. Baeten[1] & E.P. de Vink[1,2]

Abstract We discuss a combination of *GSOS*-type structural operational semantics with explicit termination, that we call the *tagh*-format (*tagh* being short for termination and *GSOS* hybrid). The *tagh*-format distinguishes between transition and termination rules, but allows besides active and negative premises as in *GSOS*, also for, what is called terminating and passive arguments. We extend the result of Aceto, Bloom and Vaandrager on the automatic generation of sound and complete axiomatizations for *GSOS* to the setting of *tagh*-transition systems. The construction of the equational theory is based upon the notion of a smooth and distinctive operation, which have been generalized from *GSOS* to *tagh*. We prove the soundness of the synthesized laws and show their completeness modulo bisimulation. The examples provided indicate a significant, though yet not ideal, improvement over the axiomatization techniques known so far.

Keywords Structured operational semantics, GSOS format, equational theories

1 Introduction

It has become very popular in the concurrency community to define various process operators by means of Plotkin-style operational rules (see e.g. [AFV01]). These are usually pretty intuitive, and they can be used to derive a transition system for each process expression. Properties of such a transition system can then be checked using a model checker.

But it is also well-known that this approach has its restrictions. Often, transition systems become too large to be handled by model checkers, or, due to the presence of parameters, transition systems have infinitely many states. In these cases, an approach using theorem provers or deploying equational reasoning can be very helpful.

In the face of these alternative approaches, it is often profitable to generate a set of laws or equations for an operator that is given by a set of operational rules. Moreover, we want two characterizations that match: the axiomatization should be sound and complete for the model of transition systems modulo (strong) bisimulation. The paper [ABV94] points the way in such an endeavour: in some cases an axiomatization can be derived by just following a recipe. Some other papers in this area are [Uli95, Uli00] (where other equivalence relations besides bisimulation equivalence are considered). However, in the years since the appearance of these papers, we have seen no application of the theory. The reader may wonder why this is so.

In our opinion, this is due to the limited process algebraic basis employed in [ABV94]; in particular, termination and deadlock are identified. Any language, both programming and specification languages, involving some form of parallel composition will know the situation when no further action is possible, but components are not finished, e.g. when two components are waiting for different communications. This situation is usually called deadlock or

[1]Department of Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

[2]LIACS, Universiteit Leiden, P.O. Box 9512, 2300 RA Leiden, The Netherlands

unsuccessful termination. Now if the language also involves some form of sequential composition, we have to know when the first component in a sequential composition is finished, i.e. successfully terminated, *in order for the second component to continue*. In such a case, deadlock must be distinguished from successful termination, and, subsequently, the axiomatization method of [ABV94] does not apply.

There are three ways to handle this combination of parallel composition and sequential composition. First, we can do away with sequential composition as a basic operator, only have prefixing as a rudimentary form of sequential composition, and use tricks like a special communication to mimic some form of sequential composition. This is the solution of CCS [Mil80, Mil89], in our opinion an unsatisfactory solution. Second, we can use *implicit termination* as in ACP [BK84, BW90], where successful termination is implicitly "tacked onto" the last action. Finally, in the majority of cases, we find *explicit termination*, usually implemented by having two separate constants, one denoting deadlock, inaction or unsuccessful termination, the other one denoting skip or successful termination. Operationally, deadlock has no rules, and termination is denoted by a predicate on states. Examples are LOTOS [Bri89], SDL [EHS97], CSP [BHR84], χ [BR97], and DiCons [JBM01].

In this paper, we adapt the theory of [ABV94] for the case of explicit termination. We think that the theory presented can be extended in order to deal also with implicit termination, but leave this as future research. Starting from the *GSOS*-format (cf. [BIM95]), we extend it with termination to obtain the *tagh*-format (termination and *GSOS* hybrid). We also employ some additional generalizations so that auxiliary operators are needed in fewer cases: for instance, the definition of sequential composition does not require auxiliary operators as in [ABV94]. This does make the theory a lot more complicated, but we gain that the generated axiomatizations are almost optimal, intuitively understandable, and are sound and complete for the model of transition systems modulo bisimulation.

The outcome is a recipe that can be applied in a straightforward manner. It is presented in Section 3. We also provide a few examples (sequential composition, leftmerge, disrupt and the priority operator) to illustrate the technique. Section 2 provides the necessary preliminaries, while section 4 and Section 5 are devoted to the soundness and completeness of the generated theory. Some concluding remarks are collected in Section 6. We hope that our generalizations will lead to actual applications.

2 Preliminaries

We assume the reader to be familiar with the standard notions and examples of process algebra (cf. [BW90, Fok00, Mil89]). Below we present the transition system for the basic process language with explicit termination ε, deadlock δ (which has no rules), a prefixing operation '$a.$ ' for every a taken from the finite alphabet of actions Act, nondeterministic choice '$+$' and unary one-step restriction operations ∂^1_B for every subset $B \subseteq Act$. The expression $\partial^1_B(t)$ indicates that the term t is not permitted to perform any action from B as a first step. However, this restriction is dropped after t has done a step outside of the action set B. For the termination predicate '$\downarrow$', we use the postfix notation $t\downarrow$ meaning that the term t has an option to terminate immediately. (See [Bae00] for a further discussion on the advantage of having explicit termination as first class citizen in a transition system.)

Definition 1

(a) The transition system TS_∂^1 consists of the following transition and termination rules:

$$a.x \xrightarrow{a} x \qquad \frac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x+y \xrightarrow{a} y'} \qquad \frac{x \xrightarrow{a} x'}{\partial_B^1(x) \xrightarrow{a} x'}\ (a \notin B)$$

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

(b) The equational theory ET_∂^1 consists of the following equations:

$$\begin{array}{rcl@{\qquad}rcl} x+y &=& y+x & \partial_B^1(x+y) &=& \partial_B^1(x)+\partial_B^1(y) \\ (x+y)+z &=& x+(y+x) & \partial_B^1(a.x) &=& a.x \ \text{ if } a \notin B \\ x+x &=& x & \partial_B^1(a.x) &=& \delta \ \text{ if } a \in B \\ x+\delta &=& x & \partial_B^1(\delta) &=& \delta \\ & & & \partial_B^1(\varepsilon) &=& \varepsilon \end{array}$$

The operation '∂_B^1' is necessary to deal with negative premises. However, as no negative premises are involved in the transition for '∂_B^1', it will turn out that the axiomatization above for this operation can be obtained from the algorithm presented below, which implies that this axiomatization is sound and complete.

We have the standard notion of strong bisimulation with predicates, in our set-up in the form of a termination condition (cf., e.g., [BW90, BV95]).

Definition 2 A bisimulation relation R for a transition system TS is a binary relation for closed terms over TS such that whenever $t_1 R t_2$ it holds that (i) $t_1 \xrightarrow{a} t_1' \implies \exists t_2'\colon t_2 \xrightarrow{a} t_2' \wedge t_1' R t_2'$, (ii) $t_2 \xrightarrow{a} t_2' \implies \exists t_1'\colon t_1 \xrightarrow{a} t_1' \wedge t_1' R t_2'$, (iii) $t_1\downarrow \iff t_2\downarrow$. Two terms t_1, t_2 are bisimilar with respect to TS if there exists a bisimulation relation R for TS with $t_1 R t_2$, notation: $t_1 \sim_{TS} t_2$ or just $t_1 \sim t_2$.

When proving soundness of the various laws that will be introduced in the sequel, the following property comes in handy.

Lemma 3 Let t_1, t_2 be two closed terms such that $t_1 \xrightarrow{a} t \iff t_2 \xrightarrow{a} t$ for all actions a and closed terms t, and $t_1\downarrow \iff t_2\downarrow$. Then it holds that $t_1 \sim t_2$. □

The next basic soundness and completeness result can be shown with standard techniques. See, e.g., [Mil89, BV95].

Theorem 4 *The equational theory ET_∂^1 as given in Definition 1b is sound and complete for TS_∂^1 modulo bisimulation.* □

The following property is straightforward.

Lemma 5 Suppose t is a term of the form $\sum_{i \in I} a_i.t_i'$ or $(\sum_{i \in I} a_i.t_i') + \varepsilon$ with, for some set of actions $B \subseteq Act$, $a_i \notin B$ for all $i \in I$. Then it holds that $ET_\partial^1 \vdash t = \partial_B^1(t)$. □

In Section 5, on completeness, we make use of the concept of head normalization. In the context of process algebra with explicit termination its definition is as follows.

Definition 6 A term t of the form ε, δ, $\sum_{i\in I} a_i.t'_i$ or $(\sum_{i\in I} a_i.t'_i) + \varepsilon$ with I a finite non-empty index set, is in *head normal form.* An equational theory ET is *head normalizing* if for all terms t there exists a term t' in head normal form such that $ET \vdash t = t'$.

Below we will use $t_1 \equiv t_2$ to denote syntactic equality of the terms t_1 and t_2. We also use expressions like $C[x_k, y_\ell, z_m]$ to indicate that only variables from the set

$$\{ x_k \mid k \in K \} \cup \{ y_\ell \mid \ell \in L \} \cup \{ z_m \mid m \in M \}$$

occur in the context $C[\;]$ with respect to some given index sets K, L and M.

3 Generating equations for the *tagh*-format

In this section we introduce the *tagh*-format for transition systems. The acronym *tagh* stands for *termination and GSOS hybrid.* It extends the *GSOS*-format as introduced in [BIM95] with a notion of explicit termination. We provide, at the end of this section, a general procedure to obtain, for each transition system in *tagh*-format, a disjoint extension TS' and an equational theory ET'. In later sections we investigate the soundness and completeness of ET' for TS'-bisimulation. As the transition system TS' is a disjoint extension of the transition system TS this amounts for terms t_1, t_2 over TS to coincidence of bisimulation with respect to TS and equality based on ET'. Thus, ET' is a sound and complete axiomatization of TS-bisimulation.

Definition 7

(a) A *tagh*-transition rule ρ for an n-ary operation f is a deduction rule of the format

$$\frac{\{ x_i \xrightarrow{a_{ip}} y_{ip} \mid i \in I, p \in P_i \} \quad \{ x_j \stackrel{b}{\not\to} \mid j \in J, b \in B_j \} \quad \{ x_k{\downarrow} \mid k \in K \}}{f(x_1, \ldots, x_n) \xrightarrow{a} C[x_m, y_{ip}]} \tag{1}$$

with $I, J, K \subseteq \{1, \ldots, n\}$, for $i \in I$, P_i a nonempty finite index set, for $j \in J$, B_j a finite (possibly empty) set of actions from Act, and, x_m, y_{ip}, for $m \in \{1, \ldots, n\}, i \in I, p \in P_i$, pairwise distinct variables, that are the only variables that may occur in the context $C[x_m, y_{ip}]$.

(b) A *tagh*-termination rule θ for an n-ary operation f is a deduction rule of the format

$$\frac{\{ x_k{\downarrow} \mid k \in K \}}{f(x_1, \ldots, x_n){\downarrow}} \tag{2}$$

with $x_1, \ldots, x_n$ pairwise distinct variables and the index set $K \subseteq \{1, \ldots, n\}$.

(c) A *tagh-transition system* is a transition system where any operation f different from 'ε', 'δ', '$a.$ ', '$+$' and '∂^1_B' has transition rules and termination rules of the *tagh*-format only.

In the context of a transition rule ρ of the format (1) we use $act(\rho)$, $neg(\rho)$, $term(\rho)$, $pass(\rho)$ to denote the index sets I, J, K, L, respectively, where $L = \{1, \ldots, n\} \setminus (I \cup J \cup K)$. For a rule θ conforming to equation (2) we put $term(\theta) = K$. For a transition rule ρ like (1), we refer to $f(x_1, \ldots, x_n)$, or an instantiation of it, as the source of ρ, and to the term $C[x_m, y_{ip}]$ as the target. Occasionally we will write $t\not\downarrow$ if *not* $t\downarrow$, i.e., t cannot terminate immediately.

The *tagh*-format is an extension of the *GSOS*-format of [BIM95]. If we strip all aspects of termination from the definition we end up with the original format for *GSOS*. We have, as the *tagh*-format is subsumed by the *panth*-format of [Ver95], that bisimulation is a congruence, just as for *GSOS*. The syntactic format of general *tagh*-transition rules though, is much too liberal to allow for an automatic generation of axioms directly. We therefore introduce (cf. [ABV94]) a more restricted format, called smooth, where there are no clashes between active, negative, terminating and passive arguments. Also an active position is not permitted to have multiple transitions. Regarding an operation f it is profitable to further restrict the collection of rules. In essence we want that at any time at most one of the transition rules for f applies. If the rules for f have this additional property, the operation is called smooth and distinctive.

Definition 8 Let *TS* be a *tagh*-transition system.

(a) A transition rule ρ in *TS* for an n-ary operation $f \in Sig$ is *smooth* if it is of the format

$$\frac{\{\, x_i \xrightarrow{a_i} y_i \mid i \in I \,\} \quad \{\, x_j \stackrel{b}{\not\rightarrow} \mid j \in J, b \in B_j \,\} \quad \{\, x_k\downarrow \mid k \in K \,\}}{f(x_1, \ldots, x_n) \xrightarrow{a} C[y_i, x_j, x_\ell]} \tag{3}$$

where the index sets I, J, K, L form a partition of $\{\, 1, \ldots, n \,\}$, $I \neq \emptyset$, $B_j \subseteq Act$ a finite (possibly empty) subset of actions, and, where in the target $C[y_i, x_j, x_\ell]$ only variables amongst $\{\, y_i \mid i \in I \,\}$, $\{\, x_p \mid p \in J \cup L \,\}$ occur. We use $act(\rho)$, $neg(\rho)$, $term(\rho)$, $pass(\rho)$ to denote I, J, K, L, respectively. The operation f is *smooth* with respect to *TS* if all of its transition rules in *TS* are smooth, and, moreover,

- for each position p in $\{1, \ldots, n\}$ it holds that $p \notin pass(\rho)$ for some rule ρ for f in *TS*.

(b) The rank of a rule ρ is the 4-tuple $\langle pass(\rho), act(\rho), term(\rho), neg(\rho) \rangle$, notation $rank(\rho)$. For two rules ρ, ρ' for an n-ary operation f we say that $rank(\rho) \succcurlyeq rank(\rho')$ iff

- $neg(\rho) = neg(\rho')$, $pass(\rho) \supseteq pass(\rho')$ and $term(\rho) \subseteq term(\rho')$, and
- $pass(\rho) \neq pass(\rho') \implies act(\rho) \cap term(\rho') \neq \emptyset$.

(c) A smooth n-ary operation f is called *smooth* and *distinctive* with respect to *TS* if

- the set $\{\, rank(\rho) \mid \rho$ a transition rule for f in $TS \,\}$ is totally ordered by the ordering $\succcurlyeq$ introduced in part (b);
- for any two distinct rules ρ, ρ' of the form (3) with $rank(\rho) = rank(\rho')$ there exists an index $i \in act(\rho) = act(\rho')$ such that $a_i \neq a'_i$;
- for each termination rule θ and each transition rule ρ for f in *TS* it holds that $term(\theta) \cap act(\rho) \neq \emptyset$.

For such an operation f it holds that $neg(\rho) = neg(\rho')$ for any two transition rules ρ, ρ'. We define $neg(f) = neg(\rho)$ and $nonneg(f) = \{1, \ldots, n\} \setminus neg(\rho)$ where ρ is an arbitrary transition rule for f in TS.

The intuition for the ordering on the transition rules for a smooth and distinctive n-ary operation f is the following: Suppose ρ and ρ' are two transition rules for f with $\rho \succcurlyeq \rho'$. The ordering on $\succcurlyeq$ then demands that a passive position in ρ' must be passive in ρ as well and, conversely, that a terminating position in ρ must also be terminating in ρ'. Now, let $\rho_1 \succcurlyeq \ldots \succcurlyeq \rho_m$ be in descending order and $p \in \{1, \ldots, n\}$ a non-negative position in f. The position p can either be passive, active or terminating in $\rho_1, \ldots, \rho_m$, but in view of the observation above we have that for suitable $0 \leq k < \ell \leq m$ it holds that $p \in pass(\rho_i)$ for $1 \leq i \leq k$, $p \in act(\rho_i)$ for $k < i \leq \ell$ and $p \in term(\rho_i)$ for $\ell < i \leq m$. So, in the context of $f(x_1, \ldots, x_n)$, the variable x_p at position p has a life-cycle from passive, via active, to terminating (but, possibly, p doesn't start out as passive or doesn't reach the termination stage).

For a smooth and distinctive n-ary operation f we have that for closed terms of the form $f(t_1, \ldots, t_n)$ where each $t_i \equiv \varepsilon, \delta, a'.t'$ at most one of the transition rules for f applies: If ρ and ρ' are two distinct rules for f, we either have $rank(\rho) = rank(\rho')$ or, without loss of generality, $rank(\rho) \succ rank(\rho')$. From the requirements of Definition 8c above we then obtain in the first case that for some $i \in \{1, \ldots, n\}$, $t_i \equiv a'.t'$ with $a' = a_i$ (the action of the i-th premise for ρ), $a' = a'_i$ (the action of the i-th premise for ρ') but also $a_i \neq a'_i$. For the second case we obtain from $rank(\rho) \succ rank(\rho')$ that $act(\rho) \cap term(\rho') \neq \emptyset$. So, for some $i \in \{1, \ldots, n\}$ we have $t_i \equiv a_i.t'$ as t_i matches the source of the i-th premise of ρ, but also $t_i \equiv \varepsilon$ as according to the rule ρ' the term t_i should terminate. All cases thus lead to a contradiction, and we conclude that $f(t_1, \ldots, t_n)$ does not match two distinct transition rules ρ and ρ'.

If we consider only transition rules ρ with empty sets $term(\rho)$ and $pass(\rho)$, the notion of smooth and distinctive for the *tagh*-format specializes to this notion for *GSOS* as introduced in [ABV94]. Note that, in the absence of termination conditions, a non-active argument can be regarded as a negative one with an empty set of forbidden actions, so that the requirement for smoothness of an operation becomes trivial. In [ABV94] there is another requirement for smooth operations, viz. that the negative arguments of all transition rules coincide. In the set-up here, this is subsumed by the condition of total ordering for smooth and distinctive operations: if $\rho \succcurlyeq \rho'$ we have $neg(\rho) = neg(\rho')$. In the set-up presented here there is for smooth rules the demand that the index set I is non-empty, which is not required by the definition of [ABV94].

The requirement of at least one active position in a smooth transition rule will be needed in our proof of the soundness of the distributive laws for negative arguments, introduced below and that are superfluous in the setting of [ABV94] but are essential for our treatment of termination (cf. Lemma 17). Likewise the condition for a position p of a smooth operation to occur non-passively in some rule ρ will be needed in the proof of the head-normalization result Lemma 24. We stress that our primary aim is to deal with explicit termination as well as to allow for what we have baptized 'passive' variables, since this will lead, in many cases, to a more satisfactory axiomatization.

Examples 9

(a) The binary operation ';' of sequential composition comes equipped, in the set-up with explicit termination, with two transition rules and one termination rule:

$$(Seq_1)\ \frac{x \xrightarrow{a} x'}{x; y \xrightarrow{a} x'; y} \qquad (Seq_2)\ \frac{x{\downarrow} \quad y \xrightarrow{a} y'}{x; y \xrightarrow{a} y'} \qquad (Seq_\varepsilon)\ \frac{x{\downarrow} \quad y{\downarrow}}{(x; y){\downarrow}}$$

We check that ';' in our set-up (contrasting [ABV94]) is a smooth and distinctive operation.

- It holds that $rank(Seq_1) = \langle \{2\}, \{1\}, \emptyset, \emptyset \rangle \succ \langle \emptyset, \{2\}, \{1\}, \emptyset \rangle = rank(Seq_2)$. So, the set $\{ rank(Seq_1), rank(Seq_2) \}$ is totally ordered.
- There are no two distinct rules of equal rank. Hence the condition on actions is trivially satisfied.
- We have $term(Seq_\varepsilon) = \{1, 2\}$ and $1 \in act(Seq_1)$, $2 \in act(Seq_2)$, so $term(Seq_\varepsilon) \cap act(Seq_i) \neq \emptyset$ for $i = 1, 2$.

(b) The binary operation '$\lfloor\!\lfloor$', usually referred to as leftmerge, has one transition rule and one termination rule:

$$(Leftmerge_1)\ \frac{x \xrightarrow{a} x'}{x \lfloor\!\lfloor y \xrightarrow{a} x' \parallel y} \qquad (Leftmerge_\varepsilon)\ \frac{x{\downarrow} \quad y{\downarrow}}{(x \lfloor\!\lfloor y){\downarrow}}$$

We have $act(\lfloor\!\lfloor) = \{1\}$, $neg(\lfloor\!\lfloor) = \{2\}$ and $term(\lfloor\!\lfloor) = pass(\lfloor\!\lfloor) = \emptyset$. Note that the format (3) allows for an empty set of 'forbidden' actions. As the leftmerge has only one transition rule, it is clear that '$\lfloor\!\lfloor$' is a smooth and distinctive operation, since $\{1, 2\} \subseteq act(Leftmerge_1) \cup neg(Leftmerge_1)$.

In concrete examples, such as the examples above, we prefer the usage of the more colloquial variable names like x, x', y, y', etc. instead of the technical x_1, y_1, x_2, y_2, etc., respectively. Also note that, in fact, we have transition schemes for (Seq_1), (Seq_2) and $(Leftmerge_1)$ rather than transition rules, as we have transition rules (Seq_1), (Seq_2) and $(Leftmerge_1)$, respectively, for each action $a \in Act$.

Before we are ready to describe the axioms generated for a smooth and distinctive n-ary operation f for a *tagh*-transition system, we need some notation: If $m \in nonneg(f)$, there exists a, not necessarily unique, transition rule ρ, maximal in rank, such that $m \notin pass(\rho)$. In that situation we put $rank(m) = rank(\rho)$ and $act(m) = act(\rho)$, $neg(m) = neg(\rho)$, etc. Also, if, for a 4-tuple R, we have that $R = rank(\rho)$, we put $act(R) = act(\rho)$, $neg(R) = neg(\rho)$, etc. The index set $handle(m)$, the handle of m with respect to f and TS, is defined as $term(m)$ if $m \in nonneg(f)$, and as $nonneg(f)$ if $m \in neg(f)$.

The idea behind the notion of a handle is that for a smooth operation f and non-negative position $m \in \{1, \ldots, n\}$ the set $handle(m)$ consists of all positions that are required to be terminating when the position m becomes active, i.e.,

$$handle(m) = \bigcap \{ term(\rho) \mid m \in act(\rho), \rho \text{ transition rule for } f \}$$

For a negative position m for f, $handle(m)$ simply consists of all non-negative positions. The handles are used in the formulation of distributivity laws; the subset-ordering on the handles of an operation induces an ordering on the applicability of these laws.

The next definition describes the various laws associated with a smooth and distinctive operation.

Definition 10 Let f be a distinctive and smooth n-ary operation for a *tagh*-transition system *TS*.

(a) For a position $p \in \{1, \ldots, n\}$ the distributive law for p with respect to f is given as follows:

$$f(\zeta_1, \ldots, z'_p + z''_p, \ldots, \zeta_n) = f(\zeta_1, \ldots, z'_p, \ldots, \zeta_n) + f(\zeta_1, \ldots, z''_p, \ldots, \zeta_n) \tag{4}$$

where $\zeta_q \equiv \varepsilon$ for $q \in \mathit{handle}(p)$ and $\zeta_q \equiv z_q$ for $q \notin \{p\} \cup \mathit{handle}(p)$.

(b) For a transition rule ρ of the format (3) the action law for ρ is given as follows:

$$f(\zeta_1, \ldots, \zeta_n) = a.C[z'_i, z_j, z_\ell] \tag{5}$$

where $\zeta_i \equiv a_i.z'_i$ for $i \in \mathit{act}(\rho)$, $\zeta_j \equiv \partial^1_{B_j}(z_j)$ for $j \in \mathit{neg}(\rho)$ with $B_j \neq \emptyset$ and $\zeta_j \equiv z_j$ for $j \in \mathit{neg}(\rho)$ with $B_j = \emptyset$, $\zeta_k \equiv \varepsilon$ for $k \in \mathit{term}(\rho)$ and $\zeta_\ell \equiv z_\ell$ for $\ell \in \mathit{pass}(\rho)$.

(c) For a rank R for f the deadlock laws are given as follows:

$$f(\zeta_1, \ldots, \zeta_n) = \delta \tag{6}$$

where ζ_m is of the form ε, δ or $a'_m.z'_m$ for $m \in \mathit{act}(R) \cup \mathit{term}(R)$, ζ_j is of the form z_j, δ, $b'_j.z'_j$ or $z_j + b'_j.z_j$ for $j \in \mathit{neg}(R)$ and $\zeta_\ell \equiv z_\ell$ for $\ell \in \mathit{pass}(R)$ such that, for each rule ρ for f in *TS* of the format (3), there exists a position p such that one of the following cases holds:

* $p \in \mathit{act}(\rho)$ and $\zeta_p \equiv \varepsilon$, $\zeta_p \equiv \delta$ or $\zeta_p \equiv a'_p.z'_p$ with $a'_p \neq a_p$, or
* $p \in \mathit{neg}(\rho)$ and $\zeta_p \equiv b'_p.z'_p$ or $\zeta_p \equiv z_p + b'_p.z'_p$ with $b'_p \in B_p$, or
* $p \in \mathit{term}(\rho)$ and $\zeta_p \equiv \delta$ or $\zeta_p \equiv a'_p.z'_p$,

and, for each termination rule θ for f there exists a position $p \in \{1, \ldots, n\}$ such that $\zeta_p \equiv \delta$ or $\zeta_p \equiv a'_p.z'_p$.

(d) For a termination rule θ for f the termination law for θ is given as follows:

$$f(\zeta_1, \ldots, \zeta_n) = \varepsilon. \tag{7}$$

where $\zeta_p \equiv \varepsilon$ for $p \in \mathit{term}(\theta)$ and $\zeta_p \equiv z_p$ for $p \notin \mathit{term}(\theta)$.

In the distributive laws we demand a 'fingerprint of ε-s' for the particular position instead of allowing a variable for *handle*-arguments. This way, non-determinism at a position is only resolved if it is guaranteed that there is sufficient termination at other positions, as will be illustrated in the examples for sequential composition ';' and leftmerge '$\lfloor\!\lfloor$' below. Note that there is also a distributive law for negative positions (which is not present in [ABV94]). The action laws are similar to those of [ABV94]. Here, we also adopt the difference in the handling of a non-empty or empty set of negative actions B_j. For the deadlock laws, it should be syntactically guaranteed that no transition rule will match. If such can be established without instantiating passive arguments, this can be reflected by the rule having variables at that places. It should however be ascertained by the form of the term that no termination rule will apply. The termination laws themselves are straightforward translations of the corresponding termination rules.

Examples 11

(a) The transition system for ';' generates, according to the definitions above, the following equations:

$$\begin{array}{rcl} (x_1+x_2);y &=& (x_1;y)+(x_2;y) \\ \varepsilon;(y_1+y_2) &=& (\varepsilon;y_1)+(\varepsilon;y_2) \\ (a.x');y &=& a.(x';y) \\ \varepsilon;(a.y') &=& a.y' \end{array} \qquad \begin{array}{rcl} \varepsilon;\delta &=& \delta \\ \delta;y &=& \delta \\ \varepsilon;\varepsilon &=& \varepsilon \end{array}$$

Note that, apart from the equation $\delta;y=\delta$, the operation ';' has also other deadlock laws, viz. $\delta;\varepsilon=\delta$, $\delta;(a.y')=\delta$ and $\delta;(y+b.y')$, which are special cases of the displayed law $\delta;y=\delta$.

(b) Similarly, we obtain for the leftmerge '$\lfloor\!\lfloor$' the following axiom system:

$$\begin{array}{rcl} (x_1+x_2)\lfloor\!\lfloor y &=& (x_1\lfloor\!\lfloor y)+(x_2\lfloor\!\lfloor y) \\ \varepsilon\lfloor\!\lfloor(y_1+y_2) &=& (\varepsilon\lfloor\!\lfloor y_1)+(\varepsilon\lfloor\!\lfloor y_2) \\ (a.x')\lfloor\!\lfloor y &=& a.(x'\parallel y) \\ && \end{array} \qquad \begin{array}{rcl} \varepsilon\lfloor\!\lfloor\delta &=& \delta \\ \varepsilon\lfloor\!\lfloor(b.y') &=& \delta \\ \delta\lfloor\!\lfloor y &=& \delta \\ \varepsilon\lfloor\!\lfloor\varepsilon &=& \varepsilon \end{array}$$

Again we omit the superfluous instantiations of the axiom $\delta;y=\delta$. Note that actually we have exactly the preferred axiomatization, see e.g. [Vra97].

From the termination law $\varepsilon;\varepsilon=\varepsilon$ and $\varepsilon\lfloor\!\lfloor\varepsilon=\varepsilon$ in the examples above, one can see the necessity of a distributive law for a negative argument, here in both cases the second position. Without these distributive laws it is not possible to derive, e.g., $\varepsilon;(a.t+\varepsilon)=a.t+\varepsilon$ and $\varepsilon\lfloor\!\lfloor(a.t+\varepsilon)=\varepsilon$, which is desired for our interpretation of optional termination. Another observation here is that the handles indicate which distributivity law should be applied first in a rewriting procedure. In the case of the sequential composition ';' given by the rules in Example 9 we have that $handle(1)=\emptyset$, $handle(2)=\{1\}$. The distributivity law for the second position is only applicable when the term at the first position is terminating and hence deterministic.

The disrupt or disabling operator '$\gg$' is well-known, e.g., from Lotos [Bri89] (see also [BB00]). In the process $x\gg y$ the subprocess x may proceed, unless the subprocess y takes over control. It terminates when either of the subprocesses does so. Thus, the disrupt operator has the following transition system:

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

The disrupt operator, as can be seen from the transition rules, is a smooth but non-distinctive operation. However, if we split the operation '$\gg$' into two, introducing '$\gg_1$' and '$\gg_2$' say, for which the transition rules satisfy the distinctiveness restrictions, we end up with two smooth and distinctive operations:

$$\frac{x \xrightarrow{a} x'}{x \gg_1 y \xrightarrow{a} x' \gg y} \qquad \frac{x{\downarrow}}{(x \gg_1 y){\downarrow}} \qquad \frac{y \xrightarrow{a} y'}{x \gg_2 y \xrightarrow{a} y'} \qquad \frac{y{\downarrow}}{(x \gg_2 y){\downarrow}}$$

The idea of splitting up '$\gg$' is also present in the transition system for this operation in [BB00]. The relationship between the various disrupt operations is expressed by the law $x \gg y = (x \gg_1 y) + (x \gg_2 y)$. Another instance of this trick is the representation of the merge '$\|$' in terms of leftmerge '$\lfloor\!\lfloor$', rightmerge '$\rfloor\!\rfloor$' and communication merge '$|$' using the law $x \parallel y = (x \lfloor\!\lfloor y) + (x \rfloor\!\rfloor y) + (x \mid y)$.

The same approach, as pointed out in [ABV94] and also applicable for the *tagh*-format, of partitioning of the set of transition rules and introducing smooth and distinctive suboperations works in general to split a smooth but non-distinctive operation f into a number of smooth and distinctive ones, $f_1, \ldots, f_s$ say. Here we only present how the resulting equations can be derived. See Lemma 21 for the soundness of this law.

Definition 12 Let f be a smooth but non-distinctive n-ary operation for the *tagh*-transition system *TS*. The n-ary operations $f_1, \ldots, f_s$ are called distinctive versions of f in a disjoint extension *TS'* of *TS* if the transition and termination rules for each f_r in *TS'* ($1 \leq r \leq s$) form, after renaming of f_r in the source of the rules by f, a partitioning of all the rules for f in *TS*. The equation

$$f(\vec{z}) = f_1(\vec{z}) + \cdots + f_s(\vec{z}) \tag{8}$$

is then referred to as the distinctivity law for f.

The previous definition addresses smooth but non-distinctive operations. However, some operations are not smooth at all. There may be several ways in which the transition rules of an operation f can violate the various conditions of the definition of smooth operations: there can be a transition rule for f that is not of the format (3), thus, either there are multiple premises for an action-argument or an active or terminating variable occurs in the target or there is overlap of the index sets or there is no active premise. Additionally, there can be a position p for which there is no transition rule for f for which this p is non-passive.

The latter situation is harmless: If a position p occurs passively only in the transition rules of an operation f we can simply interpret p as a *negative* position with an empty set of forbidden transitions. Thus removing p from the index set L and adding it to the index set J.

If a transition rule for an n-ary operation f has an empty set of active premisses, we can consider an $n+1$-ary operation f' obtained from f by adding a dummy variable x_0. For the dummy variable we require a dummy transition. By extending the transition system with a constant Ω, say, with (non-smooth) transition rule

$$\frac{\emptyset}{\Omega \xrightarrow{\omega} \Omega}$$

instantiation of the dummy variable with Ω in $f'(x_0, x_1, \ldots, x_n)$ will yield a term bisimilar to $f(x_1, \ldots, x_n)$. We therefore add the law $f(x_1, \ldots, x_n) = f'(\Omega, x_1, \ldots, x_n)$ to the equational theory.

Let us consider, in order to illustrate this, the so-called don't care choice denoted by '$\oplus$'. It is modelled by the transition rules with no premisses below. Therefore we interpret the first and second position to occur negatively in the two rules.

$$\frac{\emptyset}{x \oplus y \xrightarrow{\ell} x} \qquad \frac{\emptyset}{x \oplus y \xrightarrow{r} y}$$

This way the operation '$\oplus$' is not smooth. It is lacking an active premise. The defect, though, can be overcome easily; we simply add a dummy variable and extend the transition system with a fresh constant Ω with only an $\Omega \xrightarrow{\omega} \Omega$-transition and expand the equational theory with the ω-law $x \oplus y = \oplus'(\Omega, x, y)$. This will not contribute essentially to the dynamics of the operation '$\oplus'$' compared to '$\oplus$', nor to its termination behavior. We thus arrive at

$$\frac{w \xrightarrow{\omega} w'}{\oplus'(w, x, y) \xrightarrow{\ell} x} \qquad \frac{w \xrightarrow{\omega} w'}{\oplus'(w, x, y) \xrightarrow{r} y}$$

Now, both the first and second position of '$\oplus$' are negative and the adapted left and right rule both have an active transition. Thus '$\oplus$' is a smoothened version of the operation '$\oplus$'.

To illustrate the countermeasure for multiple active transitions, overlap over index sets and trespassing variable in the target, consider the following, synthesized, one-rule transition system adapted from [ABV94]. The operation f is non-smooth because there are multiple transitions for an active variable (viz. $x \xrightarrow{a} y_1$ and $x \xrightarrow{b} y_2$), the active and terminating variable x occurs in the target $x + y_1$, the index sets overlap (its only position 1 occurs as active, as terminating and as negative argument).

$$\frac{x \xrightarrow{a} y_1 \quad x \xrightarrow{b} y_2 \quad x \stackrel{c}{\nrightarrow} \quad x{\downarrow}}{f(x) \xrightarrow{d} x + y_1} \qquad \frac{x{\downarrow}}{f(x){\downarrow}}$$

The key idea is not to split f into new operations, but to split the variable x into new variables, i.e., we introduce separate copies x_1, x_2, x_3, x_4 of the variable x to relieve the overlap and multiplicity. The rules for f are translated into rules for a fresh operation f'. This yields the following transition system for which f' is a smooth operation:

$$\frac{x_1 \xrightarrow{a} y_1 \quad x_2 \xrightarrow{b} y_2 \quad x_3 \stackrel{c}{\nrightarrow} \quad x_4{\downarrow}}{f'(x_1, x_2, x_3, x_4) \xrightarrow{d} x_3 + y_1} \qquad \frac{x_1{\downarrow} \quad x_2{\downarrow} \quad x_3{\downarrow} \quad x_4{\downarrow}}{f'(x_1, x_2, x_3, x_4){\downarrow}}$$

As connecting law for f we have $f(x) = f'(x, x, x, x)$ which enforces that in the right-hand side we indeed have copies of the original argument.

In the next definition we will formalize the idea for the general case. In the presentation below we introduce mappings ϕ and ψ to make the correspondence explicit between a variable x_i and its splittings $\{ x'_{i'} \mid \phi(i') = i \}$ and the actions a_{ip} and output variables y_{ip} and their new names $a'_{i'}$ and $y'_{i'}$ with $\psi(i') = (i, p)$.

Definition 13 Let f be a non-smooth n-ary operation of a *tagh*-transition system *TS*. The m-ary operation f' is called the smooth version of f in a disjoint extension *TS'* of *TS*, if there exist mappings $\phi \colon \{1, \ldots, m\} \to \{1, \ldots, n\}$ and $\psi \colon \{1, \ldots, m\} \to \{1, \ldots, n\} \times \{1, \ldots, m\}$ and a 1–1 correspondence between the rules of f and f', such that

(a) a transition rule ρ for f in *TS* of the form

$$\frac{\{ x_i \xrightarrow{a_{ip}} y_{ip} \mid i \in I, p \in P_i \} \quad \{ x_j \stackrel{b_{jq}}{\nrightarrow} \mid j \in J, q \in Q_j \} \quad \{ x_k{\downarrow} \mid k \in K \}}{f(x_1, \ldots, x_n) \xrightarrow{a} C[x_i, x_j, x_k, x_\ell, y_{ip}]} \tag{9}$$

corresponds to a smooth transition rule ρ' for f' in *TS'* of the form

$$\frac{\{ x'_i \xrightarrow{a'_i} y'_i \mid i \in I' \} \quad \{ x'_j \stackrel{b'_{jq}}{\nrightarrow} \mid j \in J', q \in Q'_j \} \quad \{ x'_k{\downarrow} \mid k \in K' \}}{f'(x'_1, \ldots, x'_m) \xrightarrow{a} C'[x'_j, x'_\ell, y'_{ip}]} \tag{10}$$

such that the mapping $x'_i \xrightarrow{a'_i} y'_i \mapsto x_{\phi(i)} \xrightarrow{a'_i} y_{\psi(i)}$, $x'_j \overset{b'_{jq}}{\not\rightarrow} \mapsto x_{\phi(j)} \overset{b'_{jq}}{\not\rightarrow}$, $x'_k\!\downarrow \mapsto x_{\phi(k)}\!\downarrow$ is a bijection between the premises of ρ and the premises of ρ' and $C[x_i, x_j, x_k, x_\ell, y_{ip}] \equiv \chi(C'[x'_j, x'_\ell, y'_i])$ for a substitution χ with $\chi(x'_j) = x_{\phi(j)}, \chi(x'_\ell) = x_{\phi(\ell)}, \chi(y'_i) = y_{\psi(i)}$,

(b) a termination rule θ for f in TS of the form on the left below corresponds to a termination rule for f' in TS' of the form on the right below

$$\frac{\{\, x_k\!\downarrow \mid k \in K \,\}}{f(x_1, \ldots, x_n)\!\downarrow} \qquad\qquad \frac{\{\, x'_k\!\downarrow \mid k \in K' \,\}}{f'(x'_1, \ldots, x'_m)\!\downarrow}$$

where $K' = \phi^{-1}(K)$.

The equation

$$f(z_1, \ldots, z_n) = f'(\zeta_1, \ldots, \zeta_m), \tag{11}$$

with $\zeta_p \equiv z_{\phi(p)}$ for $p \in \{1, \ldots, n\}$, is called the smoothening law for f. In case the index set I' is empty, f' will be an $m+1$-ary operation and to its transition rules we add the active premise $x'_0 \xrightarrow{\omega} y'_0$. The transition system TS' is assumed to contain the transition $\Omega \xrightarrow{\omega} \Omega$ as only transition for the label ω. In this case the equation

$$f(z_1, \ldots, z_n) = f'(\Omega, \zeta_1, \ldots, \zeta_m), \tag{12}$$

is referred to as the smoothening law for f.

Example 14 The 'classical' example of a non-smooth operation is the priority operator θ of [BBK86]. Assuming a partial ordering on '$>$' on Act, the action rules of the unary θ and its binary smoothening θ' are the following:

$$\frac{x \xrightarrow{a} x' \quad x \overset{b}{\not\rightarrow} \quad (b > a)}{\theta(x) \xrightarrow{a} \theta(x')} \qquad \frac{x\!\downarrow}{\theta(x)\!\downarrow} \qquad \frac{x \xrightarrow{a} x' \quad y \overset{b}{\not\rightarrow} \quad (b > a)}{\theta'(x,y) \xrightarrow{a} \theta(x')} \qquad \frac{x\!\downarrow \quad y\!\downarrow}{\theta'(x,y)\!\downarrow}$$

The smoothening law for the priority operator θ is $\theta(x) = \theta'(x, x)$.

In the above we have defined how to transform a non-smooth operation into a smooth one and how to split a smooth but non-distinctive operation into several smooth and distinctive ones. In these situations the transition system will be extended disjointly, i.e., the dynamics and termination of operations already in the transition system remain unaffected. Also we have defined the smoothening law (11) and its variant (12) and the distinctivity law (8) that connects the original and new operations. For smooth and distinctive operations we have introduced various equations describing distributivity, dynamics, deadlock and termination. Collecting this all together induces the notion of the transition system and the set of equations generated by a *tagh*-transition system.

Definition 15 Let TS be a *tagh*-transition system. The *tagh*-transition system TS' generated by TS and the equational theory ET' generated by TS are given by the following procedure:

Step 0 Let TS' disjointly extend TS and TS^1_∂. Let ET' contain the equations for '+' and '∂^1_B'.

Step 1 For every non-smooth operation f of TS not in TS^1_∂, extend TS' with the smooth version f' of f and add to ET' the corresponding smoothening law (11) or (12).

Step 2 For every smooth but non-distinctive operation f of TS' (as obtained after Step 1) but not in TS^1_∂, extend TS' with the distinctive versions $f_1, \ldots, f_s$ and add to ET' the distinctivity law (8).

Step 3 For each smooth and distinctive operation f of TS' (as obtained after Step 2) but not in TS^1_∂ add to ET' the distributive laws (4), the action laws (5), the deadlock laws (6) and the termination laws (7).

Examples 16 Application of the above procedure yields for the disrupt operator '$\gg$' and the priority operator θ the following generated equational theories:

$$\begin{aligned} x \gg y &= (x \gg_1 y) + (x \gg_2 y) & \delta \gg_1 y &= \delta \\ (x_1 + x_2) \gg_1 y &= (x_1 \gg_1 y) + (x_2 \gg_1 y) & \varepsilon \gg_1 y &= \varepsilon \\ (a.x') \gg_1 y &= a.(x \gg y) & &\text{similar rules for '}\gg_2\text{'} \end{aligned}$$

$$\begin{aligned} \theta(x) &= \theta'(x, x) & \theta'(\delta, y) &= \delta \\ \theta'(x_1 + x_2, y) &= \theta'(x_1, y) + \theta'(x_2, y) & \theta'(\varepsilon, b.y') &= \delta \\ \theta'(\varepsilon, y_1 + y_2) &= \theta'(\varepsilon, y_1) + \theta'(\varepsilon, y_2) & \theta'(x, \delta) &= \delta \\ \theta'(a.x', \partial^1_{b>a}(y)) &= a.\theta(x') & \theta'(a.x, \varepsilon) &= \delta \\ \theta'(a.x', b.y + z) &= \delta \ \text{ if } b > a & \theta'(\varepsilon, \varepsilon) &= \varepsilon \end{aligned}$$

Note that the above axiomatizations are quite natural and improve upon the corresponding theory synthesized in [ABV94]. The equations for the disrupt operation coincide with those of [BB00]. The axiomatization for the priority operator avoids equations for the auxiliary 'unless' operation '$\triangleleft$' (cf. [BBK86]). However, one may want, as also discussed in [ABV94], to optimize the equations regarding their rewriting properties by introducing a rule $x \gg_2 y = y$ or to replace $\theta'(a.x', \partial^1_{b>a}(y)) = a.\theta(x')$ by the laws $\theta'(a.x, b.y + z) = \theta'(a.x, z)$ if $b \not> a$, $\theta'(a.x, \varepsilon) = a.\theta(x)$ and $\theta'(a.x, \delta) = a.\theta(x)$.

4 Soundness

In this section we first address the soundness of the laws generated for a smooth and distinctive operation: distributive laws, action laws, deadlock laws and termination laws. Next, we address the distinctivity law for a smooth but non-distinctive operation and the smoothening law for a non-smooth operation. Taking all results together we obtain a soundness result for the generated equational theory with respect to the generated disjoint extension of the original transition system.

As a direct consequence of the incorporation of explicit termination in our set-up, both in form of termination rules and in the form of having the possibility for termination premises in a transition rule, the proofs presented in this and in the next section are, at places, technically more involved. In particular, compared to the proofs of [ABV94], there are more cases in the analysis of arguments, and our format demands for distributive laws for negative positions and also for termination laws (both are not present in the framework of Aceto et al.). The latter is necessary to deal with termination, as was illustrated by the leftmerge law $\varepsilon \lfloor\!\lfloor \varepsilon = \varepsilon$ above.

Lemma 17 Let f be an n-ary smooth and distinctive operation of a *tagh*-transition system *TS*. Then it holds that the distributive laws for f are sound.

Proof

(a) Suppose $m \in \mathit{nonneg}(f)$ and $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n) = f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n)$ is a closed instance of the distributive law (4). So $t_p \equiv \varepsilon$ for $p \in \mathit{handle}(m)$. We use Lemma 3 to show that the terms $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)$ and $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n)$ are bisimilar.

(i) Assume $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n) \xrightarrow{a} t$ via the rule ρ for some a and t. It holds that $m \notin \mathit{pass}(\rho)$: For, if $m \in \mathit{pass}(\rho)$, then $\mathit{rank}(\rho) \succ \mathit{rank}(m)$. So we can choose $p \in \mathit{act}(\rho) \cap \mathit{term}(m)$. Then, on the one hand, $t_p \xrightarrow{a_p} t'_p$ for suitable t'_p, but, on the other hand, $p \in \mathit{handle}(m)$ and $t_p \equiv \varepsilon$. Contradiction. So $m \notin \mathit{pass}(\rho)$ and thus either $m \in \mathit{act}(\rho)$ or $m \in \mathit{term}(\rho)$.

Suppose $m \in \mathit{act}(\rho)$. Then $t'_m + t''_m \xrightarrow{a_m} t_m$ for some t_m. By inspection of TS_+ we derive that $t'_m \xrightarrow{a_m} t_m$ or $t''_m \xrightarrow{a_m} t_m$. Since all other premises of ρ with respect to $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)$, $f(t_1, \ldots, t'_m, \ldots, t_n)$ and $f(t_1, \ldots, t''_m, \ldots, t_n)$ are the same, it follows that $f(t_1, \ldots, t'_m, \ldots, t_n) \xrightarrow{a} t$ or $f(t_1, \ldots, t''_m, \ldots, t_n) \xrightarrow{a} t$ and thus $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n) \xrightarrow{a} t$.

Suppose $m \in \mathit{term}(\rho)$. Then we have that $(t'_m + t''_m){\downarrow}$. It follows by definition of '$\downarrow$' for '$+$' that $t'_m{\downarrow}$ or $t''_m{\downarrow}$. Since all other premises of ρ with respect to $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)$, $f(t_1, \ldots, t'_m, \ldots, t_n)$ and $f(t_1, \ldots, t''_m, \ldots, t_n)$ are the same, we derive that $f(t_1, \ldots, t'_m, \ldots, t_n) \xrightarrow{a} t$ or $f(t_1, \ldots, t''_m, \ldots, t_n) \xrightarrow{a} t$ and thus $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n) \xrightarrow{a} t$.

(ii) Assume that there is a transition $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n) \xrightarrow{a} t$ via the transition rule ρ for some a and t. By inspection of TS^1_∂ it follows that $f(t_1, \ldots, t'_m, \ldots, t_n) \xrightarrow{a} t$ or $f(t_1, \ldots, t''_m, \ldots, t_n) \xrightarrow{a} t$. Without loss of generality we can assume $f(t_1, \ldots, t'_m, \ldots, t_n) \xrightarrow{a} t$. As before it holds that $m \in \mathit{act}(\rho)$ or $m \in \mathit{term}(\rho)$.

Suppose $m \in \mathit{act}(\rho)$. Then $t'_m \xrightarrow{a_m} t_m$ for some t_m. So $t'_m + t''_m \xrightarrow{a_m} t_m$. As the premises for positions different from m coincide, we obtain $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n) \xrightarrow{a} t$. Suppose $m \in \mathit{term}(\rho)$. Then $t'_m{\downarrow}$. So, by definition of '$\downarrow$' for '$+$', $(t'_m + t''_m){\downarrow}$, hence, as all other premises for ρ with respect to $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)$ are satisfied, it follows that $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n) \xrightarrow{a} t$.

(iii) Suppose $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n){\downarrow}$ by some termination rule θ. If $m \notin \mathit{term}(\theta)$, then also $f(t_1, \ldots, t'_m, \ldots, t_n){\downarrow}$ and $f(t_1, \ldots, t''_m, \ldots, t_n){\downarrow}$, so $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n){\downarrow}$. Suppose $m \in \mathit{term}(\theta)$. Then we have that $(t'_m + t''_m){\downarrow}$ and $t_p{\downarrow}$ for $p \in \mathit{term}(\theta) \setminus \{m\}$. It follows by definition of $\downarrow$ for '$+$' that $t'_m{\downarrow}$ or $t''_m{\downarrow}$. So, by application of θ, $f(t_1, \ldots, t'_m, \ldots, t_n){\downarrow}$ or $f(t_1, \ldots, t''_m, \ldots, t_n){\downarrow}$. Again by definition of $\downarrow$ for '$+$', we obtain $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n){\downarrow}$.

Suppose $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n){\downarrow}$. By definition of $\downarrow$ for $+$, we then have $f(t_1, \ldots, t'_m, \ldots, t_n){\downarrow}$ or $f(t_1, \ldots, t''_m, \ldots, t_n){\downarrow}$. Assume $f(t_1, \ldots, t'_m, \ldots, t_n){\downarrow}$ by application of the termination rule θ. If $m \notin \mathit{term}(\theta)$ then also $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n){\downarrow}$ by application of θ. If $m \in \mathit{term}(\theta)$, we have $t'_m{\downarrow}$ and $t_p{\downarrow}$ for $p \in \mathit{term}(\theta) \setminus \{m\}$. We conclude, by definition of $\downarrow$ for '+', $(t'_m + t''_m)\downarrow$ and hence $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)\downarrow$ by application of θ.

(b) Suppose $m \in neg(f)$ and consider a closed instance $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n) = f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n)$ of the distributive law for m with respect to f. So $t_p \equiv \varepsilon$ for $p \in nonneg(f)$. Clearly, as $act(\rho) \subseteq nonneg(f)$ and $act(\rho) \neq \emptyset$, by definition, for every transition rule ρ for f, both $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)$ and $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n)$ have no transitions. Also $(t'_m + t''_m)\downarrow$ iff $t'_m\downarrow$ or $t''_m\downarrow$. From this it follows that $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)\downarrow$ iff $f(t_1, \ldots, t'_m, \ldots, t_n)\downarrow$ or $f(t_1, \ldots, t''_m, \ldots, t_n)\downarrow$, and therefore $f(t_1, \ldots, t'_m + t''_m, \ldots, t_n)\downarrow$ iff $f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n)\downarrow$. □

Note the observation that $act(\rho) \neq \emptyset$ which follows directly from Definition 8 in the last paragraph of the proof of the lemma.

Next we consider the action laws. It is here that the notion of distinctivity comes into play. In short, distinctivity captures that for a source $f(t_1, \ldots, t_n)$, with f smooth and distinctive, at most one rule can apply. As can be seen from the proof sketch for the lemma, all conditions of Definition 8c regarding distinctivity are exploited.

Lemma 18 Let f be an n-ary smooth and distinctive operation of a *tagh*-transition system *TS*. Then it holds that the action laws for the operation f are sound.

Proof Let ρ be a transition rule for f of the format (3). Let $f(t_1, \ldots, t_n) = a.C[t'_i, t'_j, t_\ell]$ be a closed instance of the action law (5) for the rule ρ for f. Hence $t_i \equiv a_i.t'_i$ for $i \in I$, $t_j \equiv \partial^1_{B_j}(t'_j)$ for $j \in J$ and $t_k \equiv \varepsilon$ for $k \in K$. Again we apply Lemma 3 to show that $f(t_1, \ldots, t_n)$ and $a.C[t'_i, t'_j, t_\ell]$ are bisimilar.

(i) Clearly, $f(t_1, \ldots, t_n) \xrightarrow{a} C[t'_i, t'_j, t_\ell]$ by application of ρ. This transition is matched by $a.C[t'_i, t'_j, t_\ell] \xrightarrow{a} C[t'_i, t'_j, t_\ell]$. Next we show, appealing to the distinctiveness of f, that $f(t_1, \ldots, t_n)$ admits no other transitions than the one based on ρ.

Suppose $f(t_1, \ldots, t_n) \xrightarrow{c} t$ via some rule ρ' for f of the format (3) with $\rho' \neq \rho$. First we derive that $rank(\rho') = rank(\rho)$ by falsification of the two cases $rank(\rho) \succ rank(\rho')$ and $rank(\rho) \prec rank(\rho')$: (1) Assume $rank(\rho) \succ rank(\rho')$, then either $pass(\rho) \setminus pass(\rho') \neq \emptyset$ or $act(\rho) \cap term(\rho') \neq \emptyset$. In the first case we have, by distinctiveness of f (cf. the 2nd bullet of Definition 8b), that $act(\rho) \cap term(\rho') \neq \emptyset$. Hence, in both cases, we can choose a position $q \in act(\rho) \cap term(\rho')$. But then we have $t_q \equiv a_q.t'_q$ as $q \in act(\rho)$ and $t_1\downarrow$ as $q \in term(\rho')$. Contradiction. (2) Assume $rank(\rho) \prec rank(\rho')$. As before we can choose a position $q \in act(\rho') \cap term(\rho)$. But then we have $t_q \xrightarrow{a'_q} t'_q$ as $q \in act(\rho')$ and $t_q \equiv \varepsilon$ as $q \in term(\rho)$. Contradiction. Since neither $rank(\rho) \succ rank(\rho')$ nor $rank(\rho) \prec rank(\rho')$ we conclude that $rank(\rho) = rank(\rho')$ by distinctiveness of f (cf. the 1st bullet of Definition 8c).

From $rank(\rho) = rank(\rho')$ we obtain $act(\rho) = act(\rho')$. If $\rho \neq \rho'$ we can choose, distinctiveness of f (cf. the 2nd bullet of Definition 8c), an index i such that $a_i \neq a'_i$. But then we have both $t_i \equiv a_i.t'_i$ and $t_i \xrightarrow{a'_i} t''_i$ for some term t''_i. Contradiction. We conclude that ρ and ρ' must coincide and that $f(t_1, \ldots, t_n)$ only admits the transition based on the transition rule ρ.

(ii) The term $a.C[t'_i, t'_j, t_\ell]$ admits exactly one transition, viz. $a.C[t'_i, t'_j, t_\ell] \xrightarrow{a} C[t'_i, t'_j, t_\ell]$ which is matched by $f(t_1, \ldots, t_n) \xrightarrow{a} C[t'_i, t'_j, t_\ell]$.

(iii) For every termination rule θ, we have $act(\rho) \cap term(\theta) \neq \emptyset$. Therefore, for each θ, $\exists i \in act(\rho) \cap term(\theta)$: $t_i \equiv a_i.t'_i$. Hence, by definition of $\downarrow$ for f, we have $f(t_1, \ldots, t_n)\not\downarrow$. Note also $a.C[t_i', t_j', t_\ell]\not\downarrow$. $\square$

The soundness of the deadlock laws is straightforward. The particular rank, for which a deadlock law is formulated, does not play a role here, but will become important for the head-normalization result (see Lemma 24) in the next section.

Lemma 19 Let f be an n-ary smooth and distinctive operation of a *tagh*-transition system *TS*. Then it holds that the deadlock laws for the operation f are sound.

Proof Let R be a rank of f and let $f(t_1, \ldots, t_n) = \delta$ be a closed instance of a deadlock law for R. Hence $t_m \equiv \varepsilon$, $t_m \equiv \delta$ or $t_m \equiv a_m'.t_m'$ for some a_m', t_m' for $m \in act(R)$ and for each rule ρ for f of the format (3) one of the following cases holds: (1) $\exists i \in act(\rho)$: $t_i \equiv \delta$ or $t_i \equiv a_i'.t_i'$ and $a_i' \neq a_i$, (2) $\exists j \in neg(\rho)$: $t_j \equiv b_j'.t_j'$ or $t_j \equiv t_j'' + b.t_j'$ for some t_j'', b, t_j' with $b \in B_j$, (3) $\exists k \in term(p)$: $t_k \equiv \delta$ or $t_k \equiv a_k'.t_k'$. It follows that for each rule ρ for f of the format (3): (1) $\exists i \in act(\rho)$: $t_i \stackrel{a_i}{\not\to}$, (2) $\exists j \in neg(\rho)$: $t_j \xrightarrow{b} t_j'$ for some action $b \in B_j$ and some term t_j', or (3) $\exists k \in term(\rho)$: $t_k\not\downarrow$.

We conclude that $f(t_1, \ldots, t_n)$ has no transitions, just as δ does. Moreover, by definition of the deadlock law, we have, for each termination rule θ, $\exists p \in term(\theta)$: $t_p \equiv \delta$ or $t_p \equiv a_p'.t_p'$. Hence $f(t_1, \ldots, t_n)\not\downarrow$. By definition of $\downarrow$ for δ, we also have $\delta\not\downarrow$. $\square$

The proof of the last soundness lemma regarding a smooth and distinctive operation makes use of the fact that for a transition rule ρ of the format (3), it holds that $act(\rho) \cap term(\theta) \neq \emptyset$. So, the termination rule θ guarantees the term ε at a position where the transition rule ρ demands an action.

Lemma 20 Let f be an n-ary smooth and distinctive operation of a *tagh*-transition system *TS*. Then it holds that the termination laws for the operation f are sound.

Proof Let $f(t_1, \ldots, t_n)$ be a closed instance of a termination law for a termination rule θ for f. Hence $t_p \equiv \varepsilon$ for all $p \in term(\theta)$. For all rules ρ for f we have that $act(\rho) \cap term(\theta) \neq \emptyset$ by distinctiveness of f. So $f(t_1, \ldots, t_n)$ has no transitions, just as ε does. Moreover, both $f(t_1, \ldots, t_n)\downarrow$, since $\forall p \in term(f)$: $t_p\downarrow$, and $\varepsilon\downarrow$ by definition. $\square$

The next result concerns the soundness of the distinctivity law for a smooth but non-distinctive operation. The construction and its proof are a modest extension of the corresponding lemma of [ABV94]. As only extra we need for the termination condition of Definition 2 that a sum can terminate iff one its summands can terminate, a fact which directly follows from the termination rules for '+' in TS^1_∂.

Lemma 21 For an n-ary smooth operation f in a *tagh*-transition system *TS*, there exists a disjoint extension TS' with smooth and distinctive n-ary operations f_1 thru f_s, say, such that $f(z_1, \ldots, z_n) = f_1(z_1, \ldots, z_n) + \cdots + f_s(z_1, \ldots, z_n)$ is sound for bisimulation modulo TS'.

Proof Start, as in [ABV94], from a partitioning $R_1, \ldots, R_s$ of the rules for f in TS such that f is smooth and distinctive with respect to each of the parts. Introduce, for each part R_r, a fresh n-ary operation f_r with as its rules the collection R_r with f replaced by f_r. Then f_r is a smooth operation. Moreover, we have that $f(t_1, \ldots, t_n) \xrightarrow{a} t$ iff $f_r(t_1, \ldots, t_n) \xrightarrow{a} t$ for some $r \in \{1, \ldots, s\}$, and $f(t_1, \ldots, t_n)\downarrow$ iff $f_r(t_1, \ldots, t_n)\downarrow$ for some $r \in \{1, \ldots, s\}$. $\square$

The soundness proof of the final building block, viz. the transition from a non-smooth operation to a smooth one, is based on the construction of Definition 13. For simplicity we suppress the issue of absence of active transitions. Two points remain: (i) to establish the number of copies that should be introduced for each argument, and (ii) to verify that the two operations admit the same transitions.

Lemma 22 Let f be a non-smooth n-ary operation of a transition system TS. Then there exist a disjoint extension TS' of TS with a smooth m-ary operation f' and an equation $f(z_1, \ldots, z_n) = f'(\zeta_1, \ldots, \zeta_m)$ with z_p, for $p \in \{1, \ldots, n\}$, all different and $\zeta_q \in \{ z_1, \ldots, z_n \}$, for $q \in \{1, \ldots, m\}$ that is sound for bisimulation modulo TS'.

Proof The proof follows the reasoning of [ABV94]: First we have to establish the number of 'copies' for each argument, using the so-called barb-factor. Then, we need a technical result (Lemma 4.12 of [ABV94]) to show that the copies will generate the same terms as their sources, i.e., that $f(z_1, \ldots, z_n) \xrightarrow{a} t \iff f'(\zeta_1, \ldots, \zeta_m) \xrightarrow{a} t$. Finally we have to check that the termination condition for bisimulation with explicit termination holds. □

By now we have addressed all the laws raised in the previous section. Concatenation of the above lemmata now yields the desired soundness result.

Theorem 23 *Let TS be a transition system in tagh-format with generated transition system TS' and generated equational theory ET'. Then the theory ET' is sound with respect to TS' modulo bisimulation.* □

5 Completeness

In this section we show, for a *tagh*-transition system TS, the completeness of the generated set of equations ET' for the generated transition system TS' modulo bisimulation. We follow the outline as provided in [ABV94].

The first result concerns head-normalization of the generated equational theory ET' and will be used as a tool to find a 'projection' t'/σ^n (see below) of a term t over the signature $\{\varepsilon, \delta, a.\ , +\}$ in the process algebra, such that $ET' \vdash t'/\sigma^n = t$. The proof of the result requires a detailed case analysis that exploits the full machinery of *handle*, *rank* and the ordering $\succcurlyeq$ on transition rules.

Lemma 24 Let TS be a transition system in *tagh*-format with generated transition system TS' and equational theory ET'. Then the theory ET' is head-normalizing for terms over TS'.

Proof It suffices to show that for any n-ary smooth and distinctive operation f and closed terms $t_1, \ldots, t_n$ in head-normal form, we have that $ET' \vdash f(t_1, \ldots, t_n) = t$ for some closed term t in head normal-form. We elaborate a detailed case analysis:

1. Assume that $\exists m \in nonneg(f)$: t_m is nondeterministic. Choose the index m maximal such that t_m is nondeterministic, say $t_m \equiv t'_m + t''_m$. We distinguish two subcases:
   (a) $[\forall p \in handle(m) \colon t_p \equiv \varepsilon]$ Put $t \equiv f(t_1, \ldots, t'_m, \ldots, t_n) + f(t_1, \ldots, t''_m, \ldots, t_n)$ and apply the distributive law for m.

(b) [$\exists p \in handle(m)$: $t_p \equiv \delta$ or $t_p \equiv a'_p.t'_p$ for some a'_p, t'_p] Note that if $p \in handle(m)$ then t_p must be deterministic, since $p \in handle(m)$ implies $rank(p) \succ rank(m)$ and m was chosen to be maximal with t_m nondeterministic.

Suppose $f(t_1, \ldots, t_n) \xrightarrow{a} t'$ for some term t' and rule ρ. By the assumption we then have $term(\rho) \not\subseteq term(m)$, so $rank(\rho) \succ rank(m)$. If $i \in act(\rho)$, then $rank(i) \succ rank(m)$, so t_i is deterministic, and, hence, $t_i \equiv a_i.t'_i$. For $j \in neg(\rho)$ we have $t_j \stackrel{b}{\not\rightarrow}$ for $b \in B_j$, hence, by Lemma 5, $t_j = \partial^1_{B_j}(t_j)$. If $k \in term(\rho)$, then $rank(k) \succ rank(m)$ by distinctiveness of f, so t_k is deterministic and therefore $t_k \equiv \varepsilon$. Now, put $t \equiv a.C[t'_i, t_j, t_\ell]$ and apply the action law for ρ.

Suppose $f(t_1, \ldots, t_n)$ admits no rules. For each rule ρ for f such that $rank(\rho) \succ rank(m)$ we have that t_q is deterministic for $q \in act(\rho) \cup term(\rho) \subseteq nonneg(f)$. As such ρ does not match $f(t_1, \ldots, t_n)$ it holds that (1) $\exists i \in act(\rho)$: $t_i \stackrel{a_i}{\not\rightarrow}$, (2) $\exists j \in neg(\rho)$: $t_j \xrightarrow{b} t'_j$ for some action $b \in B_j$ and some term t'_j, or (3) $\exists k \in term(\rho)$: $t_k\not\downarrow$. From this we derive that (1) $\exists i \in act(\rho)$: $t_i \equiv \varepsilon$, $t_i \equiv \delta$ or $t_i = a'_i.t'_i$ with $a'_i \neq a_i$, (2) $\exists j \in neg(\rho)$: $t_j \equiv b.t'_j$ or $t_j \equiv t''_j + b.t'_j$ for some action $b \in B_j$ and some term t'_j, or (3) $\exists k \in term(\rho)$: $t_k \equiv \delta$ or $t_k \equiv a'_k.t'_k$.

For each rule ρ for f such that $rank(\rho) \preccurlyeq rank(m)$ we have that $term(\rho) \supseteq handle(m)$. Since, by assumption, $\exists p \in handle(m)$: $t_p \equiv \delta$ or $t_p \equiv a'_p.t'_p$, it follows that $\exists k \in term(\rho)$: $t_k \equiv \delta$ or $t_k \equiv a'_k.t'_k$. If, for all termination rules θ, $\exists p \in term(\theta)$: $t_p \not\equiv \varepsilon$, put $t \equiv \delta$ and apply the corresponding deadlock law for $rank(m)$. If not, there exists a termination rule θ for f such that $\forall p \in term(\theta)$: $t_p \equiv \varepsilon$. Put $t \equiv \varepsilon$ and apply the termination law for θ.

2. Assume that $\forall m \in nonneg(f)$: $t_m \equiv \varepsilon$, $t_m \equiv \delta$ or $t_m \equiv a'_m.t'_m$ for some a'_m, t'_m. We distinguish three subcases:

   (i) [$f(t_1, \ldots, t_n)$ has a transition] Suppose $f(t_1, \ldots, t_n) \xrightarrow{a} C[t'_i, t_j, t_\ell]$ for some rule ρ of the form (3). Put $t \equiv a.C[t'_i, t_j, t_\ell]$ and apply the action law for ρ.

   (ii) [$\forall p \in term(\theta)$: $t_m \equiv \varepsilon$ for some termination rule θ]. Put $t \equiv \varepsilon$ and apply the corresponding termination law for θ.

   (iii) [$f(t_1, \ldots, t_n)$ admits no transition rule and, for no termination rule θ, $\forall p \in term(\theta)$: $t_m \equiv \varepsilon$] If $\forall m \in nonneg(f)$: $t_m \equiv \varepsilon$ and $\exists j \in neg(f)$: t_j is nondeterministic, apply the distributive law for j. If not, put $t \equiv \delta$ and apply the deadlock law for a rank of f with $pass(R) = \emptyset$ (which, by smootheness of f, exists as every position p is negative or becomes active or terminating eventually). □

Having the head-normalization result in place, we can conclude, using standard arguments, the completeness of the generated theory for finite processes. However, in order to deal with infinite behaviour, we need, in line with [ABV94], some extra machinery. First, we introduce a syntactic version of the Approximation Induction Principle (cf. Lemma 25). Next, we show that all 'projections' can be represented by a term for the basic transition system TS^1_∂ (cf. Lemma 26). The results are then combined (see Theorem 27) to obtain the announced completeness result.

Let TS be a *tagh*-transition system. The transition system $TS_/$ is the disjoint extension of TS and TS^1_∂ with only one binary operation '/', referred to as the hourglass operation. This hourglass operation is defined by the following rules:

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

Let σ be an arbitrary action from Act, that we think as indicating a sandgrain for the hourglass. For $n \in \mathbb{N}$, the term σ^n is defined by $\sigma^0 \equiv \delta$, $\sigma^{n+1} \equiv \sigma.\sigma^n$. The Approximation Induction Principle, *AIP* for short, can now be reformulated in terms of the hourglass and sandgrains:

$$\frac{x/\sigma^n = y/\sigma^n \quad (\forall n \in \mathbb{N})}{x = y}$$

We then have the following basic result, based on the finite branching of a *tagh*-transition system, i.e., that for all closed terms t over TS the set $\{\,(a, t') \mid t \xrightarrow{a} t' \text{ in } TS\,\}$ is finite.

Lemma 25 Let TS be a disjoint extension of $TS_/$. Then $TS \models AIP$, i.e., if, for closed terms t_1, t_2 over TS, it holds that $\forall n \in \mathbb{N}$: $t_1/\sigma^n \sim t_2/\sigma^n$ with respect to TS, then also $t_1 \sim t_2$. □

The hourglass operation '/', as can be directly seen from its rules, is smooth and distinctive. Therefore, by the results of the previous section, we have that, amongst others, the following equations hold (with respect to any disjoint extension of $TS_/$):

$$\begin{array}{rclcrcl} (x_1 + x_2)/y & = & (x_1/y) + (x_2/y) & \qquad & \delta/y & = & \delta \\ (a.x')/(b.y') & = & a.(x'/y') & & \varepsilon/y & = & \varepsilon \end{array}$$

Lemma 26 Let TS be a *tagh*-transition system and TS' the disjoint extension of $TS_/$ with accompanying equational theory ET' generated by the procedure 15. Then it holds for any closed term t' for TS' and any $n \in \mathbb{N}$, that there exists a closed term t for TS^1_δ such that $ET' \vdash t'/\sigma^n = t$ and $t'/\sigma^n \sim t$ with respect to TS'.

Proof The proof goes, as in [ABV94], by induction on n using the head-normalization result Lemma 24 and the equations for the operation '/' above. □

We are now in a position to provide the completeness result for the equations synthesized by the generation procedure. The proof of the theorem below is similar to the proof presented in [ABV94] for the *GSOS*-format. It is included here to show the reader the interplay of the various results presented above.

Theorem 27 *Let TS be a tagh-transition system. Let TS' be the disjoint extension of $TS_/$ and the generated extension of TS. Let ET' be the generated equational theory. Then ET' and AIP are sound and complete for equality modulo TS'.*

Proof If $ET', AIP \vdash t' = t''$ for two closed terms t', t'' over TS', then it follows from the soundness results of Section 4 and Lemma 25 that t' and t'' are bisimilar modulo TS'.

Suppose t', t'' are closed and bisimilar terms over TS'. Then we have that t'/σ^n and t''/σ^n are bisimilar modulo TS', for all $n \in \mathbb{N}$. Now, by virtue of *AIP*, it suffices to show that $ET' \vdash t'/\sigma^n = t''/\sigma^n$ for each n in $\mathbb{N}$. So, pick $n \in \mathbb{N}$. Choose, using Lemma 26 two closed terms t_1, t_2 over TS_+ such that $ET' \vdash t'/\sigma^n = t_1$ and $ET' \vdash t''/\sigma^n = t_2$. From the soundness of ET' we derive that t'/σ^n and t_1 are bisimilar modulo TS' that and t''/σ^n and t_2 are bisimilar modulo TS'. Hence, t_1 and t_2 are bisimilar modulo TS'. Note that TS' is a disjoint extension of TS^1_∂. We thus obtain that t_1 and t_2 are bisimilar modulo TS^1_∂. By the completeness result for TS^1_∂, Lemma 4, it follows that $ET^1_\partial \vdash t_1 = t_2$ and, a fortiori, $ET' \vdash t_1 = t_2$. We conclude that $ET' \vdash t'/\sigma^n = t''/\sigma^n$, as was to be shown. □

6 Concluding remarks

We have introduced the *tagh*-format for structured operational semantics. The *tagh*-format enhances the well-known *GSOS*-format with explicit termination. The format additionally allows for a finer distinction between the modes of the argument (viz. active, negative, terminating, passive). The method of automatic generation of axiomatizations as developed by Aceto, Bloom and Vaandrager for *GSOS* is extended for the case of *tagh*. We have shown that for a transition system in *tagh*-format the synthesized theory is sound and complete modulo bisimulation. Examples illustrate the technique and indicate the strength of the approach. The resulting laws are equal or close to hand-crafted axiomatizations.

Many other examples than the ones mentioned have been examined already. E.g., the projection operator, renaming operator, encapsulation, restriction, state operator, generalized state operator and process creation operator can be treated within the framework of the *tagh*-format. Following the practical thread, our aim is to experiment with more extensive transition systems and to investigate the impact of the axiomatization method, for example for timed transition systems. A theoretical issue here is the adaptation of the techniques for the *tagh*-format to deal with implicit termination of the form $x \xrightarrow{a} \surd$, a format at present also often used within process algebra. Note, since then we do not have the constant ε, we loose the syntactical expression of termination at the term level.

Another, theoretically important question concerns the application of the *tagh*-format in the setting of metric semantics and co-induction (cf. [BV96, Rut00]). In this paper we have focussed on transition systems and their axiomatizations. Another view is to consider transition systems and denotational models (see, for example, [Rut90, AI96, TP97]). We believe, having the correspondence of the syntactic ε with the empty semantical process p_ε of metric domain equations, it should be feasible to automatically construct higher-order or co-inductive definitions for semantical operators and a denotational semantics that is correct with respect to a transition system in *tagh*-format.

References

[ABV94] L. Aceto, B. Bloom, and F.W. Vaandrager. Turning SOS rules into equations. *Information and Computation*, 111:1–52, 1994.

[AFV01] L. Aceto, W.J. Fokkink, and C. Verhoef. Structural operational semantics. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, pages 197–292. Elsevier Science, 2001.

[AI96] L. Aceto and A. Ingólfsdóttir. CPO models for GSOS-languages–Part I: compact GSOS languages. *Information and Computation*, 129:107–141, 1996.

[Bae00] J.C.M. Baeten. Embedding untimed into timed process algebra: the case for explicit termination. In L. Aceto and B. Victor, editors, *Proc. EXPRESS'00*, pages 45–62. ENTCS 39, 2000. See http://www.elsevier.nl/locate/entcs/volume39.html.

[BB00] J.C.M. Baeten and J.A. Bergstra. Mode transfer in process algebra. Technical Report Rapport CSR 00-01, Division of Computer Science, Technische Universiteit Eindhoven, 2000. See http://www.win.tue.nl/~wwwst/fm/misc/pubbaeten.html.

[BBK86] J.C.M. Baeten, J.A. Bergstra, and J.W. Klop. Syntax and defining equations for an interrupt mechanism in process algebra. *Fundamenta Informaticae*, IX:127–168, 1986.

[BHR84] S.D. Brookes, C.A.R. Hoare, and A.W. Roscoe. A theory of Communicating Sequential Processes. *Journal of the ACM*, 31(3):560–599, 1984.

[BIM95] B. Bloom, S. Istrail, and A.R. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42:232–268, 1995. Preliminary version in Proc. POPL'88.

[BK84] J.A. Bergstra and J.W. Klop. Process algebra for synchronous communication. *Information and Control*, 60:109–137, 1984.

[BR97] D.A. van Beek and J.E. Rooda. Specification and simulation of industrial systems using an executable mathematical specification language. In *Proc. 15th IMACS World Congress, Vol. 2 Numerical Mathematics*, pages 721–726, Berlin, 1997.

[Bri89] E. Brinksma, editor. *Information Processing Systems, Open Systems Interconnection, LOTOS – A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour.* ISO Standard IS–8807, 1989.

[BV95] J.C.M. Baeten and C. Verhoef. Concrete process algebra. In S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 4, Syntactical Methods, pages 149–268. Oxford University Press, 1995.

[BV96] J.W. de Bakker and E.P. de Vink. *Control Flow Semantics.* Foundations of Computing Series. MIT Press, 1996.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*, volume 18 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 1990.

[EHS97] J. Ellsberger, D. Hogrefe, and A. Sarma. *SDL: Formal Object-Oriented Language for Communicating Systems.* Prentice Hall, 1997.

[Fok00] W.J. Fokkink. *Introduction to Process Algebra.* Texts in Theoretical Computer Science, An EATCS Series. Springer, 2000.

[JBM01] H.M.A. van Beek J.C.M. Baeten and S. Mauw. Specifying internet applications with DiCons. In *Proc. ACM Symp. on Applied Computing*, pages 576–584, Las Vegas, 2001. ACM.

[Mil80] R. Milner. *A Calculus of Communicating Systems.* LNCS 92, 1980.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice-Hall International, 1989.

[Rut90] J.J.M.M. Rutten. Deriving denotational models for bisimulation from structured operational semantics. In M. Broy and C.B. Jones, editors, *Proc. IFIP Working Group 2.2/2.3 Working Conference*, pages 155–177, 1990.

[Rut00] J.J.M.M. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249:3–80, 2000.

[TP97] D. Turi and G. Plotkin. Towards a mathematical operational semantics. In *Proc. LICS'97*, pages 280–291. IEEE, 1997.

[Uli95] I. Ulidowski. Axiomatisations of weak equivalences for De Simone languages. In I. Lee and S. Smolka, editors, *Proc. CONCUR'95*, pages 219–233. LNCS 962, 1995.

[Uli00] I. Ulidowski. Finite axiom systems for testing preorder and De Simone process languages. *Theoretical Computer Science*, 239:97–139, 2000.

[Ver95] C. Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. *Nordic Journal of Computing*, 2:274–302, 1995.

[Vra97] J.L.M. Vrancken. The algebra of communicating processes with empty process. *Theoretical Computer Science*, 177:287–328, 1997.

Computer Science Reports **Department of Mathematics and Computer Science**
Technische Universiteit Eindhoven

If you want to receive reports, send an email to: m.m.j.l.philips@tue.nl (we cannot guarantee the availability of the requested reports)

In this series appeared:

97/02	J. Hooman and O. v. Roosmalen	A Programming-Language Extension for Distributed Real-Time Systems, p. 50.
97/03	J. Blanco and A. v. Deursen	Basic Conditional Process Algebra, p. 20.
97/04	J.C.M. Baeten and J.A. Bergstra	Discrete Time Process Algebra: Absolute Time, Relative Time and Parametric Time, p. 26.
97/05	J.C.M. Baeten and J.J. Vereijken	Discrete-Time Process Algebra with Empty Process, p. 51.
97/06	M. Franssen	Tools for the Construction of Correct Programs: an Overview, p. 33.
97/07	J.C.M. Baeten and J.A. Bergstra	Bounded Stacks, Bags and Queues, p. 15.
97/08	P. Hoogendijk and R.C. Backhouse	When do datatypes commute? p. 35.
97/09	Proceedings of the Second International Workshop on Communication Modeling, Veldhoven, The Netherlands, 9-10 June, 1997.	Communication Modeling- The Language/Action Perspective, p. 147.
97/10	P.C.N. v. Gorp, E.J. Luit, D.K. Hammer E.H.L. Aarts	Distributed real-time systems: a survey of applications and a general design model, p. 31.
97/11	A. Engels, S. Mauw and M.A. Reniers	A Hierarchy of Communication Models for Message Sequence Charts, p. 30.
97/12	D. Hauschildt, E. Verbeek and W. van der Aalst	WOFLAN: A Petri-net-based Workflow Analyzer, p. 30.
97/13	W.M.P. van der Aalst	Exploring the Process Dimension of Workflow Management, p. 56.
97/14	J.F. Groote, F. Monin and J. Springintveld	A computer checked algebraic verification of a distributed summation algorithm, p. 28
97/15	M. Franssen	λP-: A Pure Type System for First Order Loginc with Automated Theorem Proving, p.35.
97/16	W.M.P. van der Aalst	On the verification of Inter-organizational workflows, p. 23
97/17	M. Vaccari and R.C. Backhouse	Calculating a Round-Robin Scheduler, p. 23.
97/18	Werkgemeenschap Informatiewetenschap redactie: P.M.E. De Bra	Informatiewetenschap 1997 Wetenschappelijke bijdragen aan de Vijfde Interdisciplinaire Conferentie Informatiewetenschap, p. 60.
98/01	W. Van der Aalst	Formalization and Verification of Event-driven Process Chains, p. 26.
98/02	M. Voorhoeve	State / Event Net Equivalence, p. 25
98/03	J.C.M. Baeten and J.A. Bergstra	Deadlock Behaviour in Split and ST Bisimulation Semantics, p. 15.
98/04	R.C. Backhouse	Pair Algebras and Galois Connections, p. 14
98/05	D. Dams	Flat Fragments of CTL and CTL*: Separating the Expressive and Distinguishing Powers. P. 22.
98/06	G. v.d. Bergen, A. Kaldewaij V.J. Dielissen	Maintenance of the Union of Intervals on a Line Revisited, p. 10.
98/07	Proceedings of the workhop on Workflow Management: Net-based Concepts, Models, Techniques and Tools (WFM'98) June 22, 1998 Lisbon, Portugal	edited by W. v.d. Aalst, p. 209
98/08	Informal proceedings of the Workshop on User Interfaces for Theorem Provers. Eindhoven University of Technology ,13-15 July 1998	edited by R.C. Backhouse, p. 180

98/09	K.M. van Hee and H.A. Reijers	An analytical method for assessing business processes, p. 29.
98/10	T. Basten and J. Hooman	Process Algebra in PVS
98/11	J. Zwanenburg	The Proof-assistemt Yarrow, p. 15
98/12	Ninth ACM Conference on Hypertext and Hypermedia Hypertext '98 Pittsburgh, USA, June 20-24, 1998 Proceedings of the second workshop on Adaptive Hypertext and Hypermedia.	Edited by P. Brusilovsky and P. De Bra, p. 95.
98/13	J.F. Groote, F. Monin and J. v.d. Pol	Checking verifications of protocols and distributed systems by computer. Extended version of a tutorial at CONCUR'98, p. 27.
99/01	V. Bos and J.J.T. Kleijn	Structured Operational Semantics of χ, p. 27
99/02	H.M.W. Verbeek, T. Basten and W.M.P. van der Aalst	Diagnosing Workflow Processes using Woflan, p. 44
99/03	R.C. Backhouse and P. Hoogendijk	Final Dialgebras: From Categories to Allegories, p. 26
99/04	S. Andova	Process Algebra with Interleaving Probabilistic Parallel Composition, p. 81
99/05	M. Franssen, R.C. Veltkamp and W. Wesselink	Efficient Evaluation of Triangular B-splines, p. 13
99/06 66	T. Basten and W. v.d. Aalst	Inheritance of Workflows: An Approach to tackling problems related to change, p.
99/07	P. Brusilovsky and P. De Bra	Second Workshop on Adaptive Systems and User Modeling on the World Wide Web, p. 119.
99/08 VFM'99	D. Bosnacki, S. Mauw, and T. Willemse	Proceedings of the first international syposium on Visual Formal Methods -
99/09	J. v.d. Pol, J. Hooman and E. de Jong	Requirements Specification and Analysis of Command and Control Systems
99/10	T.A.C. Willemse	The Analysis of a Conveyor Belt System, a case study in Hybrid Systems and timed μCRL, p. 44.
99/11	J.C.M. Baeten and C.A. Middelburg	Process Algebra with Timing: Real Time and Discrete Time, p. 50.
99/12	S. Andova	Process Algebra with Probabilistic Choice, p. 38.
99/13	K.M. van Hee, R.A. van der Toorn, J. van der Woude and P.A.C. Verkoulen	A Framework for Component Based Software Architectures, p. 19
99/14	A. Engels and S. Mauw	Why men (and octopuses) cannot juggle a four ball cascade, p. 10
99/15	J.F. Groote, W.H. Hesselink, S. Mauw, R. Vermeulen	An algorithm for the asynchronous *Write-All* problem based on process collision*, p. 11.
99/16	G.J. Houben, P. Lemmens	A Software Architecture for Generating Hypermedia Applications for Ad-Hoc Database Output, p. 13.
99/17	T. Basten, W.M.P. v.d. Aalst	Inheritance of Behavior, p.83
99/18	J.C.M. Baeten and T. Basten	Partial-Order Process Algebra (and its Relation to Petri Nets), p. 79
99/19	J.C.M. Baeten and C.A. Middelburg	Real Time Process Algebra with Time-dependent Conditions, p.33.
99/20	Proceedings Conferentie Informatiewetenschap 1999 Centrum voor Wiskunde en Informatica 12 november 1999, p.98	edited by P. de Bra and L. Hardman
00/01	J.C.M. Baeten and J.A. Bergstra	Mode Transfer in process Algebra, p. 14
00/02	J.C.M. Baeten	Process Algebra with Explicit Termination, p. 17.
00/03	S. Mauw and M.A. Reniers	A process algebra for interworkings, p. 63.
00/04	R. Bloo, J. Hooman and E. de Jong	Semantical Aspects of an Architecture for Distributed Embedded Systems*, p. 47.
00/05	J.F. Groote and M.A. Reniers	Algebraic Process Verification, p. 65.

00/06	J.F. Groote and J. v. Wamel	The Parallel Composition of Uniform Processes wit Data, p. 19
00/07	C.A. Middelburg	Variable Binding Operators in Transition System Specifications, p. 27.
00/08	I.D. van den Ende	Grammars Compared: A study on determining a suitable grammar for parsing and generating natural language sentences in order to facilitate the translation of natural language and MSC use cases, p. 33.
00/09	R.R. Hoogerwoord	A Formal Development of Distributed Summation, p. 35
00/10	T. Willemse, J. Tretmans and A. Klomp	A Case Study in Formal Methods: Specification and Validation on the OM/RR Protocol, p. 14.
00/11	T. Basten and D. Bošnački	Enhancing Partial-Order Reduction via Process Clustering, p. 14
00/12	S. Mauw, M.A. Reniers and T.A.C. Willemse	Message Sequence Charts in the Software Engineering Process, p. 26
00/13	J.C.M. Baeten, M.A. Reniers	Termination in Timed Process Algebra, p. 36
00/14	M. Voorhoeve, S. Mauw	Impossible Futures and Determinism, p. 14
00/15	M. Oostdijk	An Interactive Viewer for Mathematical Content based on Type Theory, p. 24.
00/16	F. Kamareddine, R. Bloo, R. Nederpelt	Characterizing λ-terms with equal reduction behavior, p. 12
00/17	T. Borghuis, R. Nederpelt	Belief Revision with Explicit Justifications: an Exploration in Type Theory, p. 30.
00/18	T. Laan, R. Bloo, F. Kamareddine, R. Nederpelt	Parameters in Pure Type Systems, p. 41.
00/19	J. Baeten, H. van Beek, S. Mauw	Specifying Internet applications with *DiCons*, p. 9
00/20	Editors: P. v.d. Vet and P. de Bra	Proceedings: Conferentie Informatiewetenschap 2000, De Doelen, Utrecht, 5 april 2000, p. 98
01/01	H. Zantema and J. v.d. Pol	A Rewriting Approach to Binary Decision Diagrams, p. 27
01/02	T.A.C. Willemse	Interpretations of Automata, p. 41
01/03	G.I. Joigov	Systems for Open Terms: An Overview, p. 39
01/04	P.J.L. Cuijpers, M.A. Reniers, A.G. Engels	Beyond Zeno-behaviour, p. 15
01/05	D.K. Hammer, J. Hooman, M.A. Reniers, O. van Roosmalen, A. Sintotski	Design of the mine pump control system, p. 69
01/06	J.C.M. Baeten and E.P. de Vink	Axiomatizing GSOS with termination, p. 22
01/10	7 ͤ Nederlandse Testdag editors L.M.G. Feijs, N. Goga, S. Mauw, T.A.C. Willemse	
01/11	Editors: P. de Bra, P. Brusilovsky and A. Kobsa	Proceedings: Third Workshop on Adaptive Hypertext and Hypermedia Sonthofen, Germany, July 14, 2001 Århus, Denmark, August 15, 2001

TU/e technische universiteit eindhoven

P.O. Box 513
5600 MB Eindhoven
The Netherlands

/ department of mathematics and computing science