Transition system specifications in stalk format with bisimulation as a congruence

V. van Oostrom & E.P. de Vink
Department of Mathematics and Computer Science, Vrije Universiteit
De Boelelaan 1081a, NL–1081 HV Amsterdam
oostrom@cs.vu.nl & vink@cs.vu.nl

Abstract A many-sorted variant, called stalk format, of the single sorted tyft-format for transition system specifications, introduced by Groote and Vaandrager, is proposed. The stalk format is shown to be a convenient formalism to express continuation-style transition systems for languages incorporating concepts as, e.g., process creation or backtracking, for which the existing formats seem less adequate. It is proved, extending a similar result for the single sorted case, that for an appropriate generalization of (strong) bisimilarity for the present many-sorted setting, bisimulation with respect to a transition system specification in stalk format, is a congruence. It is argued that the several conditions, required for the type of transition system specification put forward in the paper, can not be relaxed without loosing this congruence. Finally, the present format is compared with several existing ones in the literature, viz. De Simone-, GSOS- and pure tyft-format.

Keywords and phrases semantics of programming languages, Structured Operational Semantics, transition system specifications, bisimulation equivalence

1 Introduction

In providing an operational semantics for programming language concepts the Structural Operational Semantics (SOS) or Plotkin-style approach [Plo81] based on labeled transition systems is likely to be the most widely used technique. The behavior of an abstract machine executing the language under consideration is described using a set of syntax-driven axioms and rules. The steps made by the abstract machine, transitions from one configuration to another, together with the performance of some externally observable action, are exactly those which can be derived from the axioms using the rules of this collection. Referring to the collection of steps as a *transition system* one may call the description itself, the axioms and rules, a *transition system specification.*

Presently, all core example languages for concurrency theory have been given an operational semantics using SOS, e.g., CCS [Mil80], TCSP [OH86], ACP [Gla87] and Meije [Bou85] and many more variants and extensions. However, the general theory of transition system specifications has, compared to the bulk of the applications, attracted little attention. Apart from some comparative studies between SOS and differently formalized computational models, e.g., [BD92, BD93], a small number of formats for transition system specifications have been proposed in the literature, [Sim84, BIM88, GV92, Gro90, Fok93], viz. so-called De Simone, GSOS, tyft/tyxt, ntyft/ntyxt and tree rule format, respectively. Main questions studied are the bisimulation and complete trace congruences induced by the type of transition system specifications.

In the present paper we propose yet another format for transition system specifications which is suitably expressive to describe continuation style transition systems, as has been frequently used in comparative (metric) semantics, [AB88, AR89, Rut90b, BV91b, BV89, Bak91, EV93, BV92b]. The transition system specifications of these papers do not fit in the formats mentioned above, but will fit into the format introduced by the present paper, that we will call the *stalk* format. One can view it as a many sorted extension of the single sorted tyft-format without lookahead, but allowing a restricted type of contexts, so-called smooth stalks, in the rules of a specification. It will be illustrated that transition system specifications for such concepts as process creation and backtracking fit neatly within the stalk format. As will be discussed below in Section 5, the stalk format does not precisely generalize the tyft format of [GV92] due to the absence of lookahead. A technical condition on variable occurrences, necessitated by the proof of the congruence theorem, prohibits the stalk format of having such a facility. However, the examples provide, to our opinion, convincing evidence for the stalk format to be an appropriate vehicle for defining continuation style transition systems.

In order to advocate the further adequacy of the stalk format we prove (strong) *bisimilarity,* translated to the many sorted setting of the format, to be a *congruence* for transition systems induced by specification adhering to the stalk format. Such a result is desired for the familiar reasons. Firstly, identification based on observations may not be dependent upon the incidental details of a particular transition system specification. Equivalent terms should yield the same behavior in all contexts. Secondly, in the presence of a general theorem ad hoc maneuvers for particular instances of transition systems can be avoided. Checking that the transition system fits into the format will give bisimulation as a congruence for free!

It is argued in this paper that the stalk format is suited for the operational modeling of intricate (abstract) programming concepts. On syntactical grounds the present format can be considered an extension of GSOS without negative premises and of tyft without lookahead. However, further investigation should clarify the relative position of the expressiveness of the stalk format. In particular a characterization of completed trace congruences in terms of a special bisimulation would facilitate the comparison with general GSOS and pure tyft.

On the other hand, for particular cases it seems possible, following [BV92a] to discard the many sortedness of the format. It therefore may very well be the case that the restrictions of the format (contexts in the shape of smooth stalks) can be relaxed. Also, if the congruence result of the paper can be generalized (which seems to require amendments as just indicated) to cope with lookahead as for tyft, an even more general format may be obtained.

The structure of the remainder is now as follows. After mathematical preliminaries in the next section, in Section 3 the stalk format will be introduced and examples will be given illustrating its convenience and expressiveness. Bisimilarity as a congruence is the subject of Section 4. The notion of contextual bisimilarity is defined and it is shown that this observational equivalence is substitutive. Furthermore, following [GV92], evidence is given that the conditions of the format can not be dropped. In conclusion, Section 5, the comparison of the stalk format and De Simone, GSOS and tyft together with open problems are discussed.

2 Mathematical preliminaries

This section collects some preliminary definitions, notations and properties concerning many-sorted signatures, transition system specifications and bisimilarity.

A *many-sorted signature* Σ is a triple $\Sigma = (\mathcal{S}, \mathcal{F}, \mathcal{V})$ where $\mathcal{S}$ is the non-empty set of sorts or types, $\mathcal{F}$ is the set of function symbols with associated functionality and $\mathcal{V}$ is the set of $\mathcal{S}$-sorted variables. We let $\mathbb{T}(\Sigma)$ denote the collection of well-sorted (not necessarily closed) terms over Σ, and let $\mathrm{T}(\Sigma)$ denote the collection of all closed well-sorted terms. A substitution $\sigma{:}\mathcal{V} \to \mathbb{T}(\Sigma)$ is called closed iff $\sigma(v) \in \mathrm{T}(\Sigma)$ for all $v \in \mathcal{V}$. The notions of a one-hole context $C[\cdot]{:}S \to S'$ and of a multi-hole context $C[\cdot]{:}S_1 \times \cdots \times S_k \to S$ are defined as usual.

Definition 2.1 Let Σ be a many-sorted signature and $\mathcal{A}$ be a set, of the elements are referred to as actions or labels. A *transition system specification* $\mathcal{T}$ for Σ over $\mathcal{A}$ is a triple $\mathcal{T} = (\Sigma, \mathcal{A}, \mathcal{R})$ where $\mathcal{R}$ is a collection of rules of the form

$$\frac{t_i \xrightarrow{a_i} t'_i \quad (i \in I)}{t \xrightarrow{a} t'}$$

with I some index-set and, for all $i \in I$, $t_i, t'_i, t, t' \in \mathbb{T}(\Sigma)$, $a_i, a \in \mathcal{A}$.

Example 2.2 (Basic Process Algebra with δ and ϵ, [GV92] Example 3.4) Let a be a typical element of a set Act of so-called actions. Put $Act_{\checkmark} = Act \cup \{\checkmark\}$ where $\checkmark$ is some new symbol. $Act_{\checkmark}$ is ranged over by α. The signature Σ_{bpa} is single-sorted, with only sort S, say, and the following function symbols:

$a, \delta, \epsilon{:}S$ for $a \in Act$
'$\cdot$', '+':$S \times S \to S$

employing infix notion for '$\cdot$' and '+'. The transition system specification $\mathcal{T}_{bpa}$ for Σ_{bpa} over Act has as its axioms and rules the schemes $(\mathrm{Act})_a$, (Eps), $(\mathrm{Seq})_1$, $(\mathrm{Seq})_2$, $(\mathrm{Alt})_1$ and $(\mathrm{Alt})_2$ given as follows:

$$a \xrightarrow{a} \epsilon \qquad (\mathrm{Act})_a \qquad\qquad \epsilon \xrightarrow{\checkmark} \delta \qquad (\mathrm{Eps})$$

$$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y} \qquad (\mathrm{Seq})_1 \qquad\qquad \frac{x \xrightarrow{\checkmark} x' \qquad y \xrightarrow{\alpha} y'}{x \cdot y \xrightarrow{\alpha} y'} \qquad (\mathrm{Seq})_2$$

$$\frac{x \xrightarrow{\alpha} x'}{x + y \xrightarrow{\alpha} x'} \qquad (\mathrm{Alt})_1 \qquad\qquad \frac{y \xrightarrow{\alpha} y'}{x + y \xrightarrow{\alpha} y'} \qquad (\mathrm{Alt})_2$$

with x, x', y, y' variables in Σ_{bpa}.

Definition 2.3 Let $\mathcal{T} = (\Sigma, \mathcal{A}, \mathcal{R})$ be a transition system specification for a many-sorted signature Σ over an action set $\mathcal{A}$.

(a) A transition τ is a triple $(t, a, t') \in \mathrm{T}(\Sigma) \times \mathcal{A} \times \mathrm{T}(\Sigma)$, notation $t \xrightarrow{a} t'$.

(b) A finite proof tree Π for $\mathcal{T}$ is a finite, transition labeled directed tree where for each node n there exist a rule

$$\frac{t_i \xrightarrow{a_i} t_i' \quad (i \in I)}{t \xrightarrow{a} t'}$$

in $\mathcal{R}$ and a closed substitution σ such that

- n is labeled with $\sigma(t) \xrightarrow{a} \sigma(t')$
- n has $|I|$-many sons, say $\{ n_i \mid i \in I \}$
- each n_i is labeled with $\sigma(t_i) \xrightarrow{a_i} \sigma(t_i')$

(c) A finite proof tree Π is said to be proof tree for a transition τ if its root is labeled with τ.

Definition 2.4 Let T and $\mathcal{A}$ be sets and $\rightarrow \subseteq \mathrm{T} \times \mathcal{A} \times \mathrm{T}$. A binary relation '$*$' on T is called a *bisimulation* iff $t * s$ for $t, s \in \mathrm{T}$ implies, for all $a \in \mathcal{A}$,

$$\exists t' {:} t \xrightarrow{a} t' \Rightarrow \exists s' {:} s \xrightarrow{a} s' \wedge t' * s' \text{ and } \exists s' {:} s \xrightarrow{a} s' \Rightarrow \exists t' {:} t \xrightarrow{a} t' \wedge t' * s'$$

Two elements $t, s \in \mathrm{T}$ are called *bisimilar,* notation $t \leftrightarrow s$, iff $t * s$ for some bisimulation '$*$'.

Property 2.5 Let T and $\mathcal{A}$ be sets and $\rightarrow \subseteq \mathrm{T} \times \mathcal{A} \times \mathrm{T}$.

(a) $\underline{\leftrightarrow} = \bigcup\{ * \subseteq \mathrm{T}^2 \mid \text{'}*\text{' a bisimulation} \}$ is a bisimulation relation.

(b) $\underline{\leftrightarrow}$ is an equivalence relation.

Proof See [Mil90]. □

3 The stalk format

In this section we introduce a special kind of many-sorted signatures, called layered signature, in which the set of sorts is partially ordered. On the basis of the ordering of the sorts, one can distinguish certain one-hole contexts, referred to as smooth stalks. This type of contexts is, together with restrictions on variables, the main ingredient in the formulation of the notion of a transition system specification in so-called stalk format.

Definition 3.1 A pair $(\Sigma, \leq)$ is called a *layered* signature iff Σ is a many-sorted signature and $\leq$ a partial ordering, with a least element, of the sorts of Σ. We refer to the least sort of $(\Sigma, \leq)$ as the root type, notation S_{rt}. With $\mathrm{T}_{rt}(\Sigma)$ we denote the collection of closed terms over Σ of type S_{rt}.

Examples 3.2

(1) Clearly, every single-sorted signature can be considered a layered signature with sort S and $\{S\}$ trivially ordered. In particular, the signature Σ_{bpa} of Example 2.2 is layered.

(2) (uniform language with process creation) Let $\mathcal{A}$ and $\mathcal{X}$ be fixed sets. $\mathcal{A}$ can be thought of as a collection of *atomic actions*, $\mathcal{X}$ can be thought of as a collection of *procedure names*. The many-sorted signature Σ_{pc} has two sorts, viz. S_r and S_s. The function symbols of Σ_{pc} are summarized as follows:

$$\begin{array}{lll} \mathrm{E}\colon S_r & \text{`}:\text{'}\colon S_s \times S_r \to S_r & (\cdot\,,\cdot)\colon S_r \times S_r \to S_r \\ a, X\colon S_s & \mathbf{new}(\cdot)\colon S_s \to S_s & \text{`};\text{'}, \text{`+'}\colon S_s \times S_s \to S_s \end{array}$$

(for each $a \in \mathcal{A}$, $X \in \mathcal{X}$). Σ_{pc} is made a layered signature $(\Sigma_{pc}, \leq_{pc})$ if we put $S_r <_{pc} S_s$.

(3) (abstract version of Prolog with cut) The layered signature $(\Sigma_{cut}, \leq_{cut})$ has sorts S_s, S_r, S_t, S_u ordered by $S_t <_{cut} S_r <_{cut} S_u <_{cut} S_s$, and function symbols, using the sets $\mathcal{A}$ and $\mathcal{X}$ of part (2),

$$\begin{array}{ll} a, X, \mathbf{fail}, \text{`!'}\colon S_s & \text{`};\text{'}, \text{`+'}\colon S_s \times S_s \to S_s \\ \mathrm{E}\colon S_r & \text{`}:\text{'}\colon S_u \times S_r \to S_r \\ \Delta\colon S_t & \text{`}*\text{'}\colon S_r \times S_t \to S_t \\ \text{`}\triangleleft\text{'}\colon S_s \times S_t \to S_u & \end{array}$$

In Examples 3.6 till 3.8 we will present transition systems in so-called stalk format for the layered signatures just given.

Remark There is no direct relationship between our use of the ordering '$\leq$' of the sorts for a layered signature $(\Sigma, \leq)$ and the use of an ordering for the notion of an order-sorted signature in the sense of [SNGM89].

Definition 3.3 Let $(\Sigma, \leq)$ be a layered signature.

(a) A *stalk* is a context $C[\cdot]$ with function symbols only from the root to the hole (considering a term as a downward rooted tree).

(b) A stalk is called *proper* iff every function symbol has type strictly less than the type of the hole.

(c) A stalk is called *smooth* iff it is proper and the type of the function symbols is strictly ascending from the root to the hole.

(d) A one-hole context $C[\cdot]$ is called *proper* iff it is a closed (variable) instance of a proper stalk, i.e., $C[\cdot] \equiv \sigma(C'[\cdot])$ for some proper stalk $C'[\cdot]$ and closed substitution σ.

So a proper context $C[\cdot]$ with a hole of sort S, admits, on the path from the hole to the root only function symbols $f{:}S_1 \times \cdots \times S_k \to S'$ such that $S' < S$ with respect to the ordering of the layered signature $(\Sigma, \leq)$. There are no restrictions on other function symbol appearances. For a smooth stalk we demand that the function symbols appear *only* from the root to the hole. In addition the sorts of the function symbols and the hole should be strictly descending going up from the hole to the root.

Examples 3.4 Based on the layered signatures Σ_{bpa}, Σ_{pc} and Σ_{cut} of Examples 2.2 and 3.2 we have, using infix notation, the smooth stalks

$$\Box,\ \Box_s : x_r,\ \Box_r \text{ and } ((\Box_s \lhd x_t) : x_r) * x'_t$$

where $\Box$, $\Box_s$, $\Box_r$ denote holes of type S, S_s, S_r, respectively, and x_r, x_t, x'_t denote one variable of type S_r and two variables of type S_t. Note that, although variables of type S_t occur both under '$*$' and under '$\lhd$', the stalk $((\Box_s \lhd x_t) : x_r) * x'_t$ for type S_s is *smooth,* since with respect to the ordering $\leq_{cut}$ on (the sorts of) Σ_{cut} we have '$*$':$S_t <_{cut}$ '$:$':$S_r <_{cut}$ '$\lhd$':$S_u <_{cut}$ '$\Box_s$':S_s.

Assuming the same notation and s, r, t and t' to be closed terms of sort S_s, S_r and twice S_t,

$$\Box_s : (s : r),\ (\Box_s : \mathrm{E}, r) \text{ and } ((s \lhd t) : (\Box_s \lhd t : r)) * t'$$

are proper contexts but no (smooth) stalks.

We are now ready to give the definition of the stalk format for many-sorted transition system specifications. In the next section we will show, for a proper notion of bisimulation, that bisimilarity with respect to a transition system specification in stalk format is a congruence.

Definition 3.5 Let $(\Sigma, \leq)$ be a layered signature. A transition system specification $\mathcal{T} = (\Sigma, \Lambda, \mathcal{R})$ over Σ is said to be in *stalk format,* with respect to $(\Sigma, \leq)$, iff every rule in $\mathcal{R}$ is of the format

$$\frac{B_i \xrightarrow{a_i} \gamma_i \quad (i \in I)}{B[g(x_1, \ldots, x_\ell)] \xrightarrow{a} B'}$$

where

(1) B_i, γ_i, $B[g(x_1, \ldots, x_\ell)]$, B' are of the root type

(2) $\mathit{var}(B_i) \subseteq \mathit{var}(B[g(x_1, \ldots, x_\ell)])$ $(i \in I)$

(3) every variable occurs at most once in $B[g(x_1, \ldots, x_\ell)]$

(4) γ_i, $i \in I$ are pairwise distinct variables

(5) $\mathit{var}(B[g(x_1, \ldots, x_\ell)]) \cap \{\, \gamma_i \mid i \in I \,\} = \emptyset$

(6) $B[\cdot]$ is a *smooth* stalk

The first condition (1) captures the idea that transitions model the steps taken by some abstract machine executing the (abstract) language under consideration. Terms of the root sort represent the configurations of this machine. The last condition (6) means, intuitively, that the rules are not allowed to look inside the structure further than the first occurrence of a sort, i.e., deeper than the function symbol g. The remaining conditions, (2)–(5), are of a technical nature. See the discussion following Theorem 4.8.

Example 3.6 The transition system specification $\mathcal{T}_{bpa}$ over the single-sorted signature Σ_{bpa} of Example 2.2 is in stalk format. The only stalk involved is the empty context. (In fact, $\mathcal{T}_{bpa}$ is in the so-called De Simone format of [Sim84], which is a special subcase of a transition system specification in stalk format. See Section 5.)

Example 3.7 For a more illustrative example we turn to a simple uniform language with process creation. This language is a prominent example of an (abstract) imperative programming language, for which the use of a many-sorted signature in describing its transitional behavior is pivotal when aiming for a comparison with an equivalent continuation-style denotational semantics. We base our treatment on the presentation of [AB88, BV91a] and refer to these papers for a more detailed explanation of the various transition schemes.

The transition system specification $\mathcal{T}_{pc}$ is given by $\mathcal{T}_{pc} = (\Sigma_{pc}, \mathcal{A}, \{\, (\mathrm{Act})_a, (\mathrm{Rec})_X, (\mathrm{Seq}), (\mathrm{Alt})_1, (\mathrm{Alt})_2, (\mathrm{New}), (\mathrm{Par})_1, (\mathrm{Par})_2 \mid a \in \mathcal{A}, X \in \mathcal{X}\,\})$. Thus the configurations are terms of the layered signature Σ_{pc} of type S_r, the labels are taken from the pre-assumed set $\mathcal{A}$, ranged over by α. Furthermore, we have besides the transition schemes (Seq), (New), $(\mathrm{Alt})_1$, $(\mathrm{Alt})_2$, (New), $(\mathrm{Par})_1$ and $(\mathrm{Par})_2$, a rule $(\mathrm{Act})_a$ and a rule $(\mathrm{Rec})_X$ for each constant $a \in \mathcal{A}$ and $X \in \mathcal{X}$. The set of rules $(\mathrm{Rec})_X$ for $X \in \mathcal{X}$, is based on some given declaration, that determines a body D_X, i.e., a closed term of sort S_s, for the procedure name X. This scheme models the usual body replacement rule. Below s, s_1, s_2 (r, r', r_1, r_1', r_2, r_2') are variables of type S_s (S_r).

$$a : r \xrightarrow{a} r \qquad (\mathrm{Act})_a \qquad\qquad \frac{D_X : r \xrightarrow{\alpha} r'}{X : r \xrightarrow{\alpha} r'} \qquad (\mathrm{Rec})_X$$

$$\frac{s_1 : (s_2 : r) \xrightarrow{\alpha} r'}{(s_1 \,;\, s_2) : r \xrightarrow{\alpha} r'} \qquad (\mathrm{Seq}) \qquad\qquad \frac{(s : \mathrm{E}, r) \xrightarrow{\alpha} r'}{\mathbf{new}(s) : r \xrightarrow{\alpha} r'} \qquad (\mathrm{New})$$

$$\frac{s_1 : r \xrightarrow{\alpha} r'}{(s_1 + s_2) : r \xrightarrow{\alpha} r'} \qquad (\mathrm{Alt})_1 \qquad\qquad \frac{s_2 : r \xrightarrow{\alpha} r'}{(s_1 + s_2) : r \xrightarrow{\alpha} r'} \qquad (\mathrm{Alt})_2$$

$$\frac{r_1 \xrightarrow{\alpha} r_1'}{(r_1, r_2) \xrightarrow{\alpha} (r_1', r_2)} \qquad (\mathrm{Par})_1 \qquad\qquad \frac{r_2 \xrightarrow{\alpha} r_2'}{(r_1, r_2) \xrightarrow{\alpha} (r_1, r_2')} \qquad (\mathrm{Par})_2$$

Note that we have both rules for the S_s-type (viz. the schemes (Act) through (New)) as well as for the S_r-type (viz. the schemes $(\mathrm{Par})_1$ and $(\mathrm{Par})_2$). Most striking are the rules for procedures, sequential composition and the **new**-construct, for which the left-hand side of the premise is not structurally simpler than the left-hand side of the conclusion. For the constant E, representing termination, there is no rule or axiom.

Example 3.8 The next transition system specification exemplifying the stalk format, is a specification for $(\Sigma_{cut}, \leq_{cut})$. The system is used, in the literature, for an operational model for the flow of control of Prolog with cut. See [Bak91, Vin90, BV89]. The various classes of terms playing a role, can (as alternative to the description of Example 3.2) be summarized by the BNF

$$\begin{array}{llcl}
s & (\in S) & ::= & \alpha \mid \xi \mid \mathbf{fail} \mid s \,;\, s \mid s + s \mid \,! \\
r & (\in R) & ::= & \mathrm{E} \mid u : r \\
t & (\in T) & ::= & \Delta \mid r * t \\
u & (\in U) & ::= & s \triangleleft t
\end{array}$$

with α a meta-variable ranging over $\mathcal{A}$, ξ a meta-variable over $\mathcal{X}$. A typical configuration for the particular abstract machine is a stack of alternatives (of type S_r) and has the format

$$s \lhd t : r * t'$$

Here, for notational convenience, we assume '$\lhd$' to bind tighter than ':', which, in turn, is assumed to bind tighter than '$*$'. The part $s \lhd t : r$ embodies what remains to be solved for the current alternative; t' contains the other alternatives, the elaboration of which is suspended for the moment (and may be reactivated upon backtracking; see rule (Fail)). The pairs in U consist of a statement s and a so-called dump stack t' attached to s for the treatment of the cut.

Note the intertwining of the clauses for r, t and u. It is the incorporation, in the BNF above, of the type S_u (holding the pairs $S_s \times S_t$) that makes the stalk $((\cdot \lhd t) : r) * t'$, or phrased differently, $((\Box_s \lhd x_t) : x_r) * x'_t$, a smooth stalk.

We comment briefly on the backtracking construct $s_1 + s_2$ and the cut operator '!'. (The reader may consult the papers cited above for a more detailed exposition of the transition system.) The construct $s_1 + s_2$ denotes backtracking: first try to evaluate s_1, backtrack to s_2 if this fails. (Note the copying of the dump stack t in (Back).) The cut operator '!' acts as a skip, but, as a side effect, it replaces the stacked alternatives, i.e. t' in $! \lhd t : r * t'$, with the dump stack t. This dump stack encodes the alternatives that were open upon introduction of the particular statement, as can be seen from the rule $(\text{Rec})_x$.

We again assume a declaration assigning to each procedure name $X \in \mathcal{X}$ a closed term $D_X{:}S_s$. We use s, s_1, s_2, r, t, t', t'' as variables of the types as expected. The transition system specification $\mathcal{T}_{cut}$ is then given as $\mathcal{T}_{cut} = (\Sigma_{cut}, \mathcal{A}, \{ (\text{Act})_a, (\text{Rec})_X, (\text{Fail}), (\text{Seq}), (\text{Back}), (\text{Cut}) \mid a \in \mathcal{A},\ X \in \mathcal{X} \})$ where the transition schemes are given by

$$a \lhd t : r * t' \xrightarrow{a} r * t' \qquad (\text{Act})_a \qquad\qquad \frac{D_X \lhd t' : r * t' \xrightarrow{\alpha} t''}{X \lhd t : r * t' \xrightarrow{\alpha} t''} \qquad (\text{Rec})_X$$

$$\frac{t' \xrightarrow{\alpha} r'}{\mathbf{fail} \lhd t : r * t' \xrightarrow{\alpha} t''} \qquad (\text{Fail}) \qquad\qquad \frac{(s_1 \lhd t) : (s_2 \lhd t : r) * t' \xrightarrow{\alpha} t''}{(s_1 \ ; \ s_2) \lhd t : r * t' \xrightarrow{\alpha} t''} \qquad (\text{Seq})$$

$$\frac{(s_1 \lhd t : r) * (s_2 \lhd t : r * t') \xrightarrow{\alpha} t''}{(s_1 + s_2) \lhd t : r * t' \xrightarrow{\alpha} t''} \qquad (\text{Back}) \qquad\qquad \frac{r * t \xrightarrow{\alpha} t''}{! \lhd t : r * t' \xrightarrow{\alpha} t''} \qquad (\text{Cut})$$

Again we remark that, in the rules $(\text{Rec})_X$, (Seq) and (Back), the left-hand side of the premise is structurally not simpler than the left-hand side of the conclusion. No transitions are available for configurations of the form $\mathrm{E} * t$ (successful termination) and Δ (global failure).

4 Contextual bisimulation as a congruence

The concern of this section is to show that for a transition system specification over a layered signature $(\Sigma, \leq)$ in stalk format the observational equivalence called "contextual bisimulation" is a congruence for Σ. First, the concept of contextual bisimulation is introduced and its adequacy is verified as a generalization of the familiar notion of strong bisimulation. The technical work for showing contextual bisimulation a congruence is embodied in Lemma 4.7, where a certain transfer property is proved. The section concludes with a discussion of the necessity of the conditions involved in the definition of a transition system specification in stalk format.

Definition 4.1 Let $(\Sigma, \leq)$ be a layered signature, $\mathcal{A}$ a set and $\rightarrow \subseteq \mathrm{T}_{rt}(\Sigma) \times \mathcal{A} \times \mathrm{T}_{rt}(\Sigma)$.

(a) A binary relation $[*] \subseteq \mathrm{T}(\Sigma)^2$ is called a *contextual bisimulation* with respect to '$\rightarrow$' iff

  (i) '$[*]$' is sort consistent, i.e., if $u\ [*]\ v$ then u, v are of the same sort;

  (ii) $u\ [*]\ v$ for $u, v \in \mathrm{T}(\Sigma)$ of sort S, implies for all proper contexts $C[\cdot]{:}S \rightarrow S_{rt}$ and all $a \in \mathcal{A}$
  $C[u] \xrightarrow{a} D \Rightarrow \exists E{:}C[v] \xrightarrow{a} E \wedge D\ [*]\ E$ & $C[v] \xrightarrow{a} E \Rightarrow \exists D{:}C[u] \xrightarrow{a} D \wedge D\ [*]\ E$

(b) Two elements $u, v \in \mathrm{T}(\Sigma)$ are called *contextual bisimilar,* notation $u\ [\underline{\leftrightarrow}]\ v$, iff $u\ [*]\ v$ for some contextual bisimulation '$[*]$'.

The significance of the properness of the contexts $C[\cdot]$ is discussed below. We will show that contextual bisimilarity '$[\underline{\leftrightarrow}]$' indeed extends the usual notion of bisimilarity '$\underline{\leftrightarrow}$'. First we give a characterization of a relation as a contextual bisimulation.

Property 4.2 Let $(\Sigma, \leq)$ be a layered signature, $\mathcal{A}$ a set of actions and $\rightarrow \subseteq \mathrm{T}_{rt}(\Sigma) \times \mathcal{A} \times \mathrm{T}_{rt}(\Sigma)$. If a binary relation '$[*]$' is a contextual bisimulation with respect to '$\rightarrow$' then

(i) '$[*]$' is sort consistent

(ii) '$[*]$' is closed under proper contexts

(iii) '$[*]$' $\cap\, \mathrm{T}_{rt}(\Sigma)^2$ is a bisimulation

Proof ('$\Rightarrow$') (i) Sort consistency is clear by definition. (ii) If $C[\cdot]$ is proper for u and $B[\cdot]$ is proper for $C[u]$ then $B[C[\cdot]]$ is proper for u. Hence '$[*]$' is closed under proper contexts. (iii) Transitions involve terms of the sort S_{rt} only. The empty context $\square_{rt}$ is the only proper context for the root sort (for S_{rt} is the least element of Σ). Hence $[*]\ \cap \mathrm{T}_{rt}(\Sigma)^2$ is, by definition, a bisimulation. ('$\Leftarrow$') Let $[*] \subseteq \mathrm{T}(\Sigma)^2$ satisfy (i)–(iii). Pick u, v such that $u\ [*]\ v$ and $C[\cdot]{:}S \rightarrow S_{rt}$, proper. Since '$[*]$' is closed under proper contexts, we have $C[u]\ [*]\ C[v]$ for the root terms $C[u], C[v]$. Now from the fact that $[*]\ \cap \mathrm{T}_{rt}(\Sigma)^2$ is a bisimulation the result follows. $\square$

We continue the justification of the terminology of Definition 4.1.

Property 4.3 Let $(\Sigma, \leq)$ be a layered signature, $\mathcal{A}$ a set and $\rightarrow \subseteq \mathrm{T}_{rt}(\Sigma) \times \mathcal{A} \times \mathrm{T}_{rt}(\Sigma)$.

(a) $[\underline{\leftrightarrow}] = \bigcup \{\, [*] \subseteq \mathrm{T}(\Sigma)^2 \mid \text{'}[*]\text{' a contextual bisimulation} \,\}$ is a contextual bisimulation.

(b) For the bisimilarity '$\underline{\leftrightarrow}$' induced by '$\rightarrow$' we have

$$\underline{\leftrightarrow} = [\underline{\leftrightarrow}] \cap \mathrm{T}_{rt}(\Sigma)^2$$

Proof

(a) Clear by definition.

(b) ('$\subseteq$') Bisimilarity '$\underline{\leftrightarrow}$' is sort consistent, closed under proper contexts (only the empty context is proper for S_{rt}) and $\underline{\leftrightarrow} \cap \mathrm{T}_{rt}(\Sigma)^2 = \underline{\leftrightarrow}$ is a bisimulation relation. Hence, by Property 4.2 and part (a), $\underline{\leftrightarrow} \subseteq [\underline{\leftrightarrow}]$. Thus $\underline{\leftrightarrow} \subseteq [\underline{\leftrightarrow}] \cap \mathrm{T}_{rt}(\Sigma)^2$. ('$\supseteq$') From part (a) and Property 4.2 we have that $[\underline{\leftrightarrow}] \cap \mathrm{T}_{rt}(\Sigma)^2$ is a bisimulation. So, $\underline{\leftrightarrow} \supseteq [\underline{\leftrightarrow}] \cap \mathrm{T}_{rt}(\Sigma)^2$.

$\square$

Remark If, in Definition 4.1, we would drop the requirement for the contexts $C[\cdot]$ of being proper, the resulting relation would always be a congruence, as can be seen as follows: Let, for the moment, $[*]$ be sort consistent and such that $C[u] \underline{\leftrightarrow} C[v]$ for *every* context $C[\cdot]{:}S \to S_{rt}$ for $u, v{:}S$ satisfying $u\ [*]\ v$. Now let $f{:}S_1 \times \cdots \times S_k \to S$ be a k-ary function symbol in Σ and suppose $u_i\ [*]\ v_i$, for $u_i, v_i{:}S_i$ $(1 \leq i \leq k)$. Then we have

$$C[f(u_1, \ldots, u_{i-1}, u_i, v_{i+1}, \ldots, v_k)] \underline{\leftrightarrow} C[f(u_1, \ldots, u_{i-1}, v_i, v_{i+1}, \ldots, v_k)]$$

for arbitrary $C[\cdot]{:}S \to S_{rt}$. (Note that, generally, $C[f(u_1, \ldots, u_{i-1}, \cdot, v_{i+1}, \ldots, v_k)]$ need not be proper for u_i, v_i. This is the case if the type S of f is not less than S_i, the type of u_i, v_i.) Using the transitivity of $\underline{\leftrightarrow}$ we derive $C[f(u_1, \ldots, u_k)] \underline{\leftrightarrow} C[f(v_1, \ldots, v_k)]$. Hence $f(u_1, \ldots, u_k)\ [*]\ f(v_1, \ldots, v_k)$. (Cf., e.g., [Wec92], Sc. 2.1.2 for a similar type of argument.)

In general, the notion of contextual bisimilarity, as given in Definition 4.1, is not always a congruence. However, the main objection against the adaptation of Definition 4.1 as just proposed, is that it does not correctly generalize the usual notion of (strong) bisimilarity, whereas, according to Property 4.3, the notion given by Definition 4.1 does extend $\underline{\leftrightarrow}$. We will show (Theorem 4.8) that contextual bisimulation *is* substitutive when the transition relation is obtained from a transition system specification in stalk format of Definition 3.5. First we need some auxiliary machinery.

Definition 4.4

(a) The *contextual closure* $[\simeq] \subseteq \mathrm{T}(\Sigma)^2$ of $[\underline{\leftrightarrow}]$ is given by

- $[\underline{\leftrightarrow}] \subseteq [\simeq]$
- $\forall f \in \Sigma, f{:}S_1 \times \cdots \times S_k \to S\ \forall u_i, v_i{:}S_i,\ (1 \leq i \leq k)$
  $u_i\ [\simeq]\ v_i\ (1 \leq i \leq k) \ \Rightarrow f(u_1, \ldots, u_k)\ [\simeq]\ f(v_1, \ldots, v_k)$

(b) The reflexive and transitive closure of $[\simeq]$ is denoted by $[\simeq]^*$.

The binary relation $[\simeq]$ on $\mathrm{T}(\Sigma)$ is the least *substitutive* relation containing contextual bisimulation. Part (a) of the next property is therefore immediate. Part (b) then follows.

Property 4.5 Let t be a term over Σ and σ, τ be closed substitutions.

(a) $\sigma(x)\ [\simeq]\ \tau(x)$, for all $x \in var(t) \ \Rightarrow \sigma(t)\ [\simeq]\ \tau(t)$

(b) $\sigma(x)\ [\simeq]^*\ \tau(x)$, for all $x \in var(t) \ \Rightarrow \sigma(t)\ [\simeq]^*\ \tau(t)$

We occasionally write "σ $[\simeq]$ τ on X", for some collection of variables X, meaning $\sigma(x)$ $[\simeq]$ $\tau(x)$ for all $x \in X$. We employ a similar abbreviation for $[\simeq]^*$.

A last observation preparing the way in proving our main result Theorem 4.8 via Lemma 4.7 concerns a rephrasing of the transfer property, i.e., condition (ii) of Definition 4.1(a), for contextual bisimulation. Subsitutivity of $[\simeq]$, enforced by its definition, implies that two closed terms u, v satisfy u $[\simeq]$ v iff they are syntactically equal modulo contextually *bisimilar* subterms. Therefore comparing one-hole contexts $C[u]$ and $C[v]$ with u $[\simeq]$ v boils down to comparing multi-hole contexts $C[u_1, \ldots, u_k]$ and $C[v_1, \ldots, v_k]$ with u_i $[\underline{\leftrightarrow}]$ v_i.

Property 4.6

(a) For $u, v \in \mathrm{T}(\Sigma)$ it holds that

$$\begin{aligned} u\ [\simeq]\ v \quad \Leftrightarrow \quad & \exists C[\cdot]{:}S_1 \times \cdots \times S_k \to S \exists u_i, v_i{:}S_i\ (1 \leq i \leq k) : \\ & u \equiv C[u_1, \ldots, u_k] \wedge v \equiv C[v_1, \ldots, v_k] \wedge u_i\ [\underline{\leftrightarrow}]\ v_i\ (1 \leq i \leq k) \end{aligned}$$

(b) The following two statements are equivalent:

(i) $\forall C[\cdot]{:}S \to S_{rt}$, proper, $\forall u, v{:}S$:
$u\ [\simeq]\ v \wedge C[u] \xrightarrow{a} D \Rightarrow \exists E{:}\, C[v] \xrightarrow{a} E \wedge D\ [\simeq]^*\ E$

(ii) $\forall C'[\cdot]{:}S_1 \times \cdots \times S_k \to S_{rt}\ \forall u_i, v_i{:}S_i\ (1 \leq i \leq k)$:
$u_i\ [\underline{\leftrightarrow}]\ v_i\ (1 \leq i \leq k)\ \wedge\ C'[u_1, \ldots, u_k] \xrightarrow{a} D \Rightarrow \exists E{:}\, C'[v_1, \ldots, v_k] \xrightarrow{a} E \wedge D\ [\simeq]^*\ E$

Proof

(a) Induction on the structure of u and on the structure of $C[\cdot]$.

(b) ('$\Rightarrow$') Take $u \equiv C'[u_1, \ldots, u_k]$, $v \equiv C'[v_1, \ldots, v_k]$ and $C[\cdot] \equiv \Box$. Then, by part (a), we have u $[\simeq]$ v. ('$\Leftarrow$') Clear.

$\Box$

Remark Note that in part (a) above the left-hand side involves the relation $[\simeq]$ whereas the right-hand side involves $[\underline{\leftrightarrow}]$. Therefore the multi-hole contexts $C[\cdot]$ are not required to be proper.

We are now ready for proving contextual bisimulation, with respect to a transition system specification in stalk format, a congruence.

Lemma 4.7 Fix a transition system specification $\mathcal{T} = (\Sigma, A, \mathcal{R})$ in stalk format for the layered signature $(\Sigma, \leq)$ and let $\underline{\leftrightarrow}$, $[\underline{\leftrightarrow}]$, $[\simeq]$ and $[\simeq]^*$ be given with respect to the transition relation induced by $\mathcal{T}$. Then we have

$$u\ [\simeq]\ v \wedge C[u] \xrightarrow{a} D \Rightarrow \exists E{:}\, C[v] \xrightarrow{a} E \wedge D\ [\simeq]^*\ E$$

for $u, v{:}S$ in $\mathrm{T}(\Sigma)$ and a proper context $C[\cdot]{:}S \to S_{rt}$.

Proof By the above Property 4.7, the equivalence of (i) and (ii), is suffices to show for $C[\cdot]{:}S_1\times\cdots\times S_k \to S$ and $u_i \ [\underline{\leftrightarrow}]\ v_i$ $(1\le i\le k)$

$$C[u_1,\ldots,u_k] \xrightarrow{a} D \Rightarrow \exists E{:}\, C[v_1,\ldots,v_k] \xrightarrow{a} E \wedge D\ [\simeq]^*\ E$$

We reason by induction on the height of a (closed) proof for $C[u_1,\ldots,u_k] \xrightarrow{a} D$ and sub-induction on k. Let

$$\frac{B_i \xrightarrow{a_i} \gamma_i \quad (i\in I)}{B[g(x_1,\ldots,x_\ell)] \xrightarrow{a} B'} \tag{4.1}$$

be the last rule from $\mathcal{R}$ in the proof for $C[u_1,\ldots,u_k] \xrightarrow{a} D$, and σ be the last closed substitution. Thus we have

$$\sigma(B[g(x_1,\ldots,x_\ell)]) \equiv C[u_1,\ldots,u_k]$$

$$\sigma(B_i \xrightarrow{a_i} \gamma_i)\ (i\in I)$$

$$\sigma(B') \equiv D$$

We will show the existence of a root term E such that $C[v_1,\ldots,v_k] \xrightarrow{a} E \wedge D\ [\simeq]^*\ E$. For this we distinguish two cases: (I) the displayed occurrence of g in $B[g(x_1,\ldots,x_\ell)]$ is in $C[\cdot]$; (II) the occurrence of g is in one of the displayed u_i in $C[u_1,\ldots,u_k]$ for some i, $(1\le i\le k)$.

(I) g in C: since $B[\cdot]$ is a stalk it follows that every u_i $(1\le i\le k)$ is in the scope of some variable of $B[g(x_1,\ldots,x_\ell)]$. All variables of $B[g(x_1,\ldots,x_\ell)]$ occur exactly once (condition (3) of Definition 3.5), hence we may choose a closed substitution τ such that

$$\tau(B[g(x_1,\ldots,x_\ell)]) \equiv C[v_1,\ldots,v_k]$$
$$\sigma\ [\simeq]\ \tau \text{ on } \mathit{var}(B[g(x_1,\ldots,x_\ell)])$$

(τ is obtained from σ by replacing the particular occurrences of u_i in $\sigma(x)$, for x a variable, by v_i. The second clause of the definition for $[\simeq]$ then guarantees $\sigma(x)\ [\simeq]\ \tau(x)$.) Since $\mathit{var}(B_i) \subseteq \mathit{var}(B[g(x_1,\ldots,x_\ell)])$ we obtain, using Property 4.5, $\sigma(B_i)\ [\simeq]\ \tau(B_i)$, $(i\in I)$. By the induction hypothesis, it follows from $\sigma(B_i) \xrightarrow{a_i} \sigma(\gamma_i)$ —which have shorter proof trees—, that $\tau(B_i) \xrightarrow{a_i} B_i'$ for some suitable B_i' such that $\sigma(\gamma_i)\ [\simeq]^*\ B_i'$ $(i\in I)$. Since $\mathit{var}(B[g(x_1,\ldots,x_\ell)]) \cap \{\,\gamma_i \mid i\in I\,\} = \emptyset$ and the γ_i $(i\in I)$ are pairwise distinct, we may assume, without loss of generality, $\tau(\gamma_i) \equiv B_i'$ $(i\in I)$. Let furthermore $\tau(y) \equiv \sigma(y)$ for $y \in \mathit{var}(B') \setminus (\mathit{var}(B[g(x_1,\ldots,x_\ell)]) \cup \{\,\gamma_i \mid i\in I\,\})$. Note, under this assumption, $\sigma\ [\simeq]^*\ \tau$ on $\{\,\gamma_i \mid i\in I\,\}$. Application of the rule of equation (4.1) then yields

$$\tau(B[g(x_1,\ldots,x_\ell)]) \xrightarrow{a} \tau(B')$$

Thus, since $\tau(B[g(x_1,\ldots,x_\ell)]) \equiv C[v_1,\ldots,v_k]$ and $\sigma\ [\simeq]^*\ \tau$ on $\mathit{var}(B')$ we have for $E \equiv \tau(B')$:

$$C[v_1,\ldots,v_k] \xrightarrow{a} E \wedge D\ [\simeq]^*\ E$$

(II) g in u_i: Since $B[\cdot]$ is a smooth stalk, it follows that $C[\cdot]$ is a proper context for u_i. By induction hypothesis (for k) it follows from

$$C[\ldots, u_{i-1}, u_i, u_{i+1}, \ldots] \xrightarrow{a} D$$

that we may (replacing u_j by v_j for $j \neq i$) choose F such that

$$C[\ldots, v_{i-1}, u_i, v_{i+1}, \ldots] \xrightarrow{a} F \wedge D \; [\simeq]^* \; F$$

(Taking u_i fixed, $C[\ldots, u_i, \ldots]$ is a (k–1)-hole context assuming $k > 1$; the statement trivially holds for $k = 1$.) Now, by definition of $u_i \; [\underline{\leftrightarrow}] \; v_i$ and properness of $C[\cdot]$ for u_i, we can find E such that

$$C[\ldots, v_{i-1}, v_i, v_{i+1}, \ldots] \xrightarrow{a} E \wedge F \underline{\leftrightarrow} E$$

Since $\underline{\leftrightarrow} \subseteq [\underline{\leftrightarrow}] \subseteq [\simeq] \subseteq [\simeq]^*$ by Property 4.2 and $D \; [\simeq]^* \; F \underline{\leftrightarrow} E$, we conclude

$$C[v_1, \ldots, v_k] \xrightarrow{a} E \wedge D \; [\simeq]^* \; E$$

Thus, in both cases, $u_i \; [\simeq] \; v_i$ $(1 \leq i \leq k)$ and $C[u_1, \ldots, u_k] \xrightarrow{a} D$ imply the existence of E such that $C[v_1, \ldots, v_k] \xrightarrow{a} D$ and $D \; [\simeq]^* \; E$. □

Using Lemma 4.7 the main result now follows directly.

Theorem 4.8 Let $\mathcal{T} = (\Sigma, A, \mathcal{R})$ be a transition system specification over the layered signature $(\Sigma, \leq)$ in stalk format. Then it holds that contextual bisimulation $[\underline{\leftrightarrow}]$ with respect to $\mathcal{T}$ is a congruence for Σ.

Proof From the lemma it follows (by an easy inductive argument and by symmetry): for $u \; [\simeq]^* \; v$ and $C[\cdot]$ proper for u, v:

$$(C[u] \xrightarrow{a} D \Rightarrow \exists E\colon C[v] \xrightarrow{a} E \wedge D \; [\simeq]^* \; E) \;\wedge\; (C[v] \xrightarrow{a} E \Rightarrow \exists D\colon C[u] \xrightarrow{a} D \wedge D \; [\simeq]^* \; E)$$

Hence, $[\simeq]^*$ is a contextual bisimulation. So, by Property 4.3, $[\simeq]^* \subseteq [\underline{\leftrightarrow}]$. By definition of $[\simeq]$, we have $[\underline{\leftrightarrow}] \subseteq [\simeq] \subseteq [\simeq]^*$. Therefore $[\underline{\leftrightarrow}] = [\simeq] = [\simeq]^*$. In particular, by the second clause for $[\simeq]$, if $u_i, v_i{:}S_i$ such that $u_i \; [\underline{\leftrightarrow}] \; v_i$ $(1 \leq i \leq k)$ and $f \in \Sigma$, $f{:}S_1 \times \cdots \times S_k \to S$, then $f(u_1, \ldots, u_i) \; [\underline{\leftrightarrow}] \; f(v_1, \ldots, v_k)$. □

We conclude this section with a further explanation of the various restrictions on the transition schemes imposed by the stalk format.

The first condition of Definition 3.5 reflects the point of view that the transition system specification should describe transitions between *configurations* of some abstract machine for the particular language. If one wishes to do so, the definition of contextual bisimulation (Definition 4.1) can be adapted to arbitrary typed (and proper) contexts; (the proof of) Lemma 4.7 and Theorem 4.8 remain valid under this modification.

The second clause for the stalk format, viz. $\mathit{var}(B_i) \subseteq \mathit{var}(B[g(x_1, \ldots, x_\ell)])$ $(i \in I)$, is crucial for the set-up of the proof as is given for Lemma 4.7. (Incorporation of this condition implies that the stalk format does not strictly subsumes the tyft-format for the single-sorted case. See Section 5.) For the induction hypothesis to be applicable in case (I) of the induction step, we need to know $\sigma(B_i)$ $[\simeq]$ $\tau(B_i)$. If variables among γ_i $(i \in I)$ are allowed in the B_i's (as is the case under certain non-circularity conditions for the tyft-format of [GV92]) we can only conclude $\sigma(B_i)$ $[\simeq]^*$ $\tau(B_i)$. The observation $\sigma(B_i) \equiv B'_0$ $[\simeq]$ $\cdots$ $[\simeq]$ $B'_n \equiv \tau(B_i)$ for suitable B'_j $(1 \leq j \leq n)$ does not help, since, generally, the height of the derivations of the transitions of the intermediate B'_j is not bounded by the height of the derivation for $\sigma(B[g(x_1, \ldots, x_\ell)]) \xrightarrow{a} \sigma(B')$.

In [GV92] various counter examples (Examples 5.11.1 through 5.11.5) are constructed which show that, generally, for the single-sorted tyft-format, bisimulation is no longer a congruence in case *any* of the conditions (3) through (6) in Definition 3.5 is omitted. We observe that the transition system specification of $P(BPA^{\epsilon}_{\delta})$, on which the counter examples are based, fits into the stalk format (cf. Example 3.6). Therefore the examples given in [GV92] illustrate that also for the present format, the conditions are essential for Theorem 4.8 to hold. With respect to condition (2) of Definition 3.5, Groote and Vaandrager show, translated to the presentation here, the necessity of

$$var(B_i) \subseteq var(B[g(x_1, \ldots, x_\ell)]) \cup \{\, \gamma_i \mid i \in I \,\} \tag{4.2}$$

(in combination with a non-circularity condition that we suppress). As discussed above, we need, for technical reasons, a stronger requirement. At the moment, it is an open question whether or not the more liberal version, equation (4.2), can be used instead. We will not copy the counter examples of [GV92] but refer the reader to the particular paper.

5 Comparison and future work

In positioning the proposed stalk format, let us first describe, translated to our present setting, the De Simone, GSOS and tyft format.

The De Simone format [Sim84, Sim85], which is historically the first format introduced, allows rules of the form

$$\frac{x_i \xrightarrow{a_i} y_i \quad (i \in I)}{f(x_1, \ldots, x_\ell) \xrightarrow{a} t}$$

where the variables x_k, y_i $(1 \leq k \leq \ell, i \in I)$ are all distinct, $I \subseteq \{\, 1, \ldots, \ell \,\}$ and the term t is a so-called architectural expression (the definition of which does not matter for the discussion here), which amongst others, imposes a linearity condition on the variables of t, and, more particularly, satisfies $var(t) \subseteq \{\, z_1, \ldots z_\ell \,\}$ where $z_k \equiv x_k$ if $k \notin I$ and $z_k \equiv y_k$ otherwise.

The form of a GSOS rule or, as it is called in [BIM88], a Structured Transition Rule is the following

$$\frac{x_k \xrightarrow{a_{ij}} y_{ij} \quad (i \in I, j \in J) \qquad x_k \stackrel{b_{ik}}{\not\rightarrow} (i \in I', k \in K)}{f(x_1, \ldots, x_\ell) \xrightarrow{a} t}$$

where x_i, y_{ij} $(1 \leq k \leq \ell, j \in J)$ are all different variables $I, I' \subseteq \{\, 1, \ldots, \ell \,\}$ and $var(t) \subseteq \{\, x_i, y_{ij} \mid 1 \leq i \leq \ell, j \in J \,\}$. The antecedents $x_i \stackrel{b_{ik}}{\not\rightarrow}$ are so-called *negative* premises and express the *absence* of any transition $x_i \xrightarrow{b_{ik}} t_{ik}$ (for the current instance of x_i).

Finally, a tyft rule of [GV92] fits into the format

$$\frac{t_i \xrightarrow{a_i} y_i \quad (i \in I)}{f(x_1, \ldots, x_\ell) \xrightarrow{a} t}$$

where the variables x_k, y_i ($1 \leq k \leq \ell, i \in I$) are all distinct, and following from an additional well-foundedness and freeness condition, $var(t) \subseteq \{ x_k, y_i \mid 1 \leq k \leq \ell, i \in I \}$. The auxiliary condition demands that a variable may occur as right-hand side of a premise only if any variable in the left-hand side of the premise is either one of the variables $x_1, \ldots, x_\ell$ or, otherwise, may already have occurred as right-hand side of a premise. (This is a rephrasing of the inductivity requirement in the presentation of the tyft format in [Rut92].) Thus a rule of the form, taken from [BK88],

$$\frac{x \xrightarrow{a_1} x_1 \qquad x_1 \xrightarrow{a_2} x_2 \qquad \ldots \qquad x_n \xrightarrow{a_n} \mathrm{E}}{[x] \xrightarrow{a} \mathrm{E}}$$

where $x, x_1, \ldots, x_n$ are variables, E is a constant and $[\cdot]$ is unary function symbol, is permitted in the tyft format. This feature, referred to as lookahead, is not allowed by the stalk-format (for otherwise the proof of Theorem 4.8 would not hold). In the transition system specifications of [GV92] also rules in the so-called tyxt-format are allowed. Since such rules can be replaced using rules in tyft-format only without changing the induced set of transitions, we do not consider them in the comparison below.

A first difference, in comparing the stalk format to the three formats given above, concerns the many-sortedness of the former vs. the single-sortedness of the latter three. The many-sortedness is not an essential distinction, since following [BV92a], many-sortedness can be, so to speak, coded into the transition schemes by ensuring the well-sortedness of the $x_1, \ldots, x_\ell$ by adding extra premises $x_i \xrightarrow{\checkmark} sort(S_i)$ and rules

$$\frac{x_i \xrightarrow{\checkmark} sort(S_i) \quad (1 \leq i \leq k)}{f(x_1, \ldots, x_k) \xrightarrow{\checkmark} sort(S)}$$

for $f{:}S_1 \times \cdots \times S_k \to S$, where $\checkmark$ is a new symbol and the $sort(S')$ are new constants, one for each $S' \in \mathcal{S}$. However, the many-sortedness is natural in the context of continuation-style operational semantics. In fact, main inspiration for the formulation of a transition system specification in stalk format for which bisimulation is a congruence are the studies in comparative semantics dealing with resumptions/continuations. Cf. [AB88, AR89, Bak91, BK88, BV91a, BV91b, BV92b, BV89, EV93, Rut90b]. There transition system specifications not fitting in the De Simone, GSOS and tyft formats are used as a basis for an operational semantics to be compared with a denotational one.

A second, more decisive, difference between the stalk formats and the others concerns the function symbols allowed to occur in the left-hand side of the conclusion of a rule. Note that the De Simone, GSOS and tyft format only allow flat expressions $f(x_1, \ldots, x_\ell)$ with a *single* function symbol to occur, whereas the proposed format admits more general contexts, viz. smooth stalks (from which the format inherits its name). Several function symbols in the left-hand side of the conclusion of rules are the main attraction of the stalk format in that they facilitate a convenient description of a transition system, e.g., for the programming concepts discussed in the papers cited above. Thus, from a syntactic point of view, the stalk format is incomparable to the GSOS and tyft format and subsumes the De Simone format, as can be directly verified from the definitions.

In [GV92] it is argued that, from the semantic point of view of completed trance congruences, the tyft format is more distinctive then GSOS, which, in turn, is more distinctive than the De Simone format. By characterizing trace congruence by a specialized (bi)simulation one can classify (restricting to image finite transition system specifications) the various formats: completed trace congruence for De Simone format corresponds to failure equivalence $\equiv_F$ [Sim84, BKO88], for GSOS it corresponds to $\frac{2}{3}$-bisimulation equivalence $\underline{\leftrightarrow}_{\frac{2}{3}}$ [BIM88, LS89], while for the tyft format it corresponds to 2-nested simulation equivalence $\leftrightarrows^2$ [GV92]. Since $\equiv_F \subseteq \underline{\leftrightarrow}_{\frac{2}{3}} \subseteq \leftrightarrows^2$ the result then follows. In the perspective of such a relationship, future work includes the characterization of completed trace congruence for the stalk format. In order to see whether expressiveness is added, with respect to the GSOS and tyft formats, by the feature of multiple function symbols such a specialized (bi)simultion should be compared to $\frac{2}{3}$-bisimulation and 2-nested simulation equivalence.

Further research spawning off now involves the incorporation of negative premises along the lines of the ntyft/ntyxt format [Gro90, BG91]. One might be optimistic about incorporation of the GSOS format by the stalk-format with negative premises, if such can be achieved. More detailed re-examination of the mere shape of a stalk (and the proof of Theorem 4.8) may lead to certain tree-like contexts replacing smooth stalk in the definition of the format. Also, such questions as finite vs. infinite expressibility as in an issue in comparing the tree rule and tyft format in [Fok93] should be considered. Furthermore, related decidability and termrewriting questions should be explored.

A final direction into which to extend the research concerns the derivation of denotational semantics from transition system specifications. In [Rut90a] such a construction is achieved in a setting of metric spaces for the De Simone format. In [Rut92] this is extended, in a setting of non-wellfounded sets, to the (image finite) tyft-format. In both cases, the denotational semantics obtained is of a direct nature. The stalk format is inspired by continuation style models. It is open how to adapt the method of Rutten to deal with the stalk format in general and to deal, in particular, with the resumption/continuation operational and denotational meanings referred to above.

Acknowledgment The second author is grateful to Jaco de Bakker, Franck van Breugel, Jan Rutten, Frits Vaandrager and Herbert Wiklicky for their comments on the draft of this paper.

References

[AB88] P. America and J.W. de Bakker. Designing equivalent models for process creation. *Theoretical Computer Science*, 60:109–176, 1988.

[AR89] P.H.M. America and J.J.M.M. Rutten. A parallel object-oriented language: Design and semantic foundations. In J.W. de Bakker, editor, *Languages for Parallel Architectures: Design, Semantics, Implementation Models*, pages 1–49. Wiley, 1989.

[Bak91] J.W. de Bakker. Comparative semantics for flow of control in logic programming without logic. *Information and Computation*, 91:123–179, 1991.

[BD92] E. Badouel and P. Darondeau. Structural operational specifications and trace automata. In W.R. Cleaveland, editor, *Proc. CONCUR'92*, pages 302–316. LNCS 630, 1992.

[BD93] E. Badouel and P. Darondeau. Trace nets. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Semantics: Foundations and Applications*, pages 21–50. LNCS 666, 1993.

[BG91] R. Bol and J.F. Groote. The meaning of negative premises in transition system specifications. In J. Leach Albert, B. Monien, and M. Rodríguez Artalejo, editors, *Proc. ICALP'91*, pages 481–494. LNCS 510, 1991.

[BIM88] B. Bloom, S. Istrail, and A.R. Meyer. Bisimulation can't be traced. In *Proc. Principles of Programming Languages*, pages 229–239. San Diego, 1988.

[BK88] J.W. de Bakker and J.N. Kok. Uniform abstraction, atomicity and contractions in the comparative semantics of Concurrent Prolog. In *Proc. of the International Conference on Fifth Generation Computer Systems*, pages 347–355. Institute for New Generation Computer Technology, 1988.

[BKO88] J.A. Bergstra, J.W. Klop, and E.-R. Olderog. Readies and failures in the algebra of communicating processes. *SIAM Journal on Computing*, 17:1134–1177, 1988.

[Bou85] G. Boudol. Notes on algebraic calculi of processes. In K. Apt, editor, *Logics and Models of Concurrent Systems*, pages 261–303. Springer, NATO ASI Series F13, 1985.

[BV89] A. de Bruin and E.P. de Vink. Continuation semantics for Prolog with cut (extended abstract). In J. Díaz and F. Orejas, editors, *Proc. Theory and Practice of Software Development '89*, volume 1, pages 178–192. LNCS 351, 1989.

[BV91a] J.W. de Bakker and E.P. de Vink. CCS for OO and LP. In S. Abramsky and T.S.E. Maibaum, editors, *Proc. TAPSOFT'91, volume 2*, pages 1–28. LNCS 494, 1991.

[BV91b] J.W. de Bakker and E.P. de Vink. Rendez-vous with metric semantics. In E.H.L. Aarts, J. van Leeuwen, and M. Rem, editors, *Proc. Parallel Architectures and Languages '91*, pages 27–57. LNCS 506, 1991.

[BV92a] J.C.M. Baeten and F.W. Vaandrager. An algebra for process creation. *Acta Informatica*, 29:303–334, 1992.

[BV92b] J.W. de Bakker and E.P. de Vink. Bisimulation semantics for concurrency with atomicity and action refinement. Technical Report CS–R9210, CWI, 1992.

[EV93] A. Eliëns and E.P. de Vink. Asynchronous rendez-vous in Distributed Logic Programming. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *Semantics: Foundations and Applications*, pages 174–203. LNCS 666, 1993.

[Fok93] W.J. Fokkink. The tyft/tyxt format reduces to tree rules. Technical report, CWI, 1993. Draft. Extended abstract to appear in proc. TACS'94.

[Gla87] R.J. van Glabbeek. Bounded nondeterminism and the approximation induction principle in process algebra. In F.J. Brandenburg, G. Vidal-Naquet, and M. Wirsing, editors, *Proc. STACS'87*, pages 336–347. LNCS 247, 1987.

[Gro90] J.F. Groote. Transition system specifications with negative premises. In J.C.M. Baeten and J.W. Klop, editors, *Proc. CONCUR'90, Theories of Concurrency: Unification and Extension*, pages 332–342. LNCS 458, 1990.

[GV92] J.F. Groote and F.W. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100:202–260, 1992.

[LS89] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. In *Proc. Principles of Programming Languages*, pages 344–352, 1989.

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

[Mil90] R. Milner. Operational and algebraic semantics of concurrent processes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 1201–1242. Elsevier, 1990.

[OH86] E.-R. Olderog and C.A.R. Hoare. Specification-oriented semantics for communicating processes. *Acta Informatica*, 23:9–66, 1986.

[Plo81] G.D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN–19, Computer Science Department, Aarhus University, 1981.

[Rut90a] J.J.M.M. Rutten. Deriving denotational models for bisimulation from Structured Operational Semantics. In M. Broy and C.B. Jones, editors, *Programming concepts and methods, proc. of the IFIP Working Group 2.2/2.3 Working Conference*, pages 155–177. North-Holland, 1990.

[Rut90b] J.J.M.M. Rutten. Semantic correctness for a parallel object-oriented language. *SIAM Journal on Computing*, 19:341–383, 1990.

[Rut92] J.J.M.M. Rutten. Processes as terms: non-well-founded models for bisimulation. *Mathematical Structures in Computer Science*, 2:257–275, 1992.

[Sim84] R. de Simone. *Calculabilité et Expressivité dans l'Algèbre de Processus Parallèles MEIJE.* PhD thesis, Thèse de 3eme cycle, Université de Paris VII, 1984.

[Sim85] R. de Simone. Higher-level synchronising devices in Meije-SCCS. *Theoretical Computer Science*, 37:245–267, 1985.

[SNGM89] G. Smolka, W. Nutt, J. Goguen, and J. Meseguer. Order sorted equational computation. In H. Aït-Kaci and M. Nivat, editors, *Resolution of Equations in Algebraic Structures Volume 2: Rewriting Techniques*, pages 297–367. Academic Press, 1989.

[Vin90] E.P. de Vink. Comparative semantics for Prolog with cut. *Science of Computer Programming*, 13:341–383, 1990.

[Wec92] W. Wechler. *Universal Algebra for Computer Scientists*, volume 25 of *EATCS Monographs on Theoretical Computer Science.* Springer, 1992.