Structural Operational Semantics and Bounded Nondeterminism

Wan Fokkink[1,2] and Thuy Duong Vu[3]

[1] CWI, Department of Software Engineering, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands, email: `wan@cwi.nl`

[2] Vrije Universiteit Amsterdam, Department of Theoretical Computer Science, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands, email: `wanf@cs.vu.nl`

[3] Universiteit van Amsterdam, Programming Research Group, Faculty of Science, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands, email: `tdvu@science.uva.nl`

Abstract. We present a rule format for structural operational semantics to guarantee that the associated labelled transition system is bounded nondeterministic.

1 Introduction

Structural operational semantics [22, 24] provides process algebras and specification languages with an interpretation. It generates a *labelled transition system* (LTS), in which states are the closed terms over a (single-sorted, first-order) signature, and transitions between states may be supplied with labels. The transitions between states are obtained from a *transition system specification* (TSS), being a signature together with a set of proof rules called *transition rules*.

Rule formats for structural operational semantics are a convenient approach for obtaining certain results on process algebras and LTSs in general (see [1] for an overview). Some rule formats guarantee that a behavioural equivalence relation is a *congruence* with respect to the LTS associated with a TSS, meaning that each function symbol respects this equivalence. For bisimulation equivalence there are the *De Simone* [26], *GSOS* [9] and *ntyft/ntyxt* [10, 17, 18] formats. Furthermore, congruence formats have been developed for behavioural equivalences based on decorated traces [6, 8, 12, 29] and weak bisimulations [7, 13, 27, 28].

Let a TSS and its signature be extended with new transition rules and function symbols, respectively. The rule formats put forward in [11, 17, 18, 31] guarantee that such an extension is *(operationally) conservative*, meaning that the provable transitions for a term over the original signature are the same both in the original and in the extended TSS.

Finally, some rule formats exist to guarantee that the LTS associated to a TSS is *bounded nondeterministic* (cf. [15]), meaning that each state in the LTS has only finitely many outgoing transitions. Vaandrager [30] investigated the properties of LTSs associated to TSSs in De Simone format. He introduced the notion of a *bounded* TSS (meaning that for each *type* and *trigger* there are only finitely many transition rules in the TSS), and proved that the LTS associated to a bounded TSS in De Simone format is bounded nondeterministic. Bloom [5] generalised the GSOS format to a higher-order setting, and formulated a notion of boundedness reminiscent to the one of Vaandrager. Bloom proved that the LTS associated to a bounded TSS in his higher-order GSOS format is bounded nondeterministic.

The requirements of both the De Simone and the higher-order GSOS format are in fact far too restrictive if one is only interested in bounded nondeterminism, because they were first of all meant to guarantee a congruence property. In this paper we present a more liberal rule format, called the *bounded nondeterminism* format, to guarantee that the LTS associated to a TSS is bounded nondeterministic. Following Vaandrager [30], we moreover need a restrictive notion of bounded TSSs. Finally, we require that TSSs allow a *strict stratification*, roughly meaning that there exists a certain order on the terms occurring at left-hand sides of transitions in a transition rule. A similar notion of a strict stratification (for transitions instead of terms) was previously used in the realm of structural operational semantics to associate an LTS to a TSS containing negative premises [10, 16, 17]. We prove that if a bounded TSS is in bounded nondeterminism format, and allows a strict stratification, then the LTS associated to this TSS is bounded nondeterministic.

2 Structural operational semantics

This chapter introduces the basic notions about structural operational semantics needed in the rest of this paper. More information on structural operational semantics can be found in [1].

2.1 Signatures

We start by defining the syntax, consisting of the terms over some algebraic signature. Throughout this text we assume the existence of a countably infinite set V of *variables* with typical elements x, y, z. A syntactic object is *closed* if it does not contain any variables.

Definition 1 (Signature). *A* signature Σ *is a set of* function symbols*, disjoint from* V*, together with an* arity *mapping that assigns a natural number* $ar(f)$ *to each function symbol* f*. A function symbol of arity zero is called* constant*, while function symbols of arity one and two are called* unary *and* binary*, respectively.*

Definition 2 (Term). *The set* $\mathbb{T}(\Sigma)$ *of* (open) terms *over a signature* Σ*, ranged over by* t, u, v*, is the least set such that:*

- *each variable is a term, and*
- *if* f *is a function symbol and* $t_1, \ldots, t_{ar(f)}$ *are terms, then* $f(t_1, \ldots, t_{ar(f)})$ *is a term.*

For terms t*,* $var(t)$ *denotes the set of variables that occur in* t*.* $T(\Sigma)$ *denotes the set of closed terms over* Σ*, ranged over by* p, q*.*

Definition 3 (Substitution). *A* substitution *is a partial mapping $\sigma : V \to \mathbb{T}(\Sigma)$. A substitution is* closed *if it maps each variable in its domain to a closed term. A substitution extends to a mapping from terms to terms as usual; the term $\sigma(t)$ is obtained by replacing occurrences in t of variables x in the domain of σ by $\sigma(x)$.*

2.2 Labelled transition systems

We assume a non-empty set of *actions* (or *transition labels*) A, ranged over by a, b.

Definition 4 (Labelled transition system). *A* labelled transition system *(LTS) is a pair $(Proc, \to)$ with $Proc$ a set of* states *(or* processes*) and $\to \subseteq Proc \times A \times Proc$. A triple $(s, a, s') \in \to$ is called a* transition.

We use the more suggestive notation $s \xrightarrow{a} s'$ for $(s, a, s') \in \to$.

Definition 5 (Bounded nondeterminism). *A state s in an LTS is* finitely branching *if it has only finitely many outgoing transitions $s \xrightarrow{a} s'$. An LTS is* bounded nondeterministic *if all its states are finitely branching.*

2.3 Transition system specifications

Transition system specifications are used as a general framework for defining an operational semantics. We assume a signature Σ and a set of actions A.

Definition 6 (Transition system specification). *A* transition rule *ρ is of the form $H/t \xrightarrow{a} t'$, where H a set of* positive premises *$u \xrightarrow{b} u'$ and* negative premises *$v \overset{c}{\not\longrightarrow}$, with $t, t', u, u', v \in \mathbb{T}(\Sigma)$. The left- and the right-hand side of the* conclusion *$t \xrightarrow{a} t'$ are the* source *and the* target *of ρ, respectively. A* transition system specification *(TSS) is a set of transition rules.*

From now on, *Proc* will consist of $T(\Sigma)$, and the transitions are the ones provable from a TSS over Σ.

We define when a transition rule is provable from a TSS. The derivation of a transition $p \xrightarrow{a} p'$ corresponds to the derivation of the closed transition rule $\emptyset/p \xrightarrow{a} p'$.

Definition 7 (Proof). *A* proof *of a transition rule $H/t \xrightarrow{a} t'$ from a TSS R is an upwardly branching tree without infinite branches, whose nodes are labelled by expressions $u \xrightarrow{b} u'$, such that:*

- *the root is labelled by $t \xrightarrow{a} t'$, and*
- *if K is the set of labels of the nodes directly above a node with label $u \xrightarrow{b} u'$, then*
  1. *either $K = \emptyset$ and $u \xrightarrow{b} u' \in H$,*

2. or $K/u \xrightarrow{b} u'$ is a substitution instance of a transition rule in R.

A TSS without negative premises specifies an LTS in a straightforward way as the set of all provable transitions. But it is much less trivial to associate an LTS to a TSS with negative premises. In [16] eleven answers to the questions "*Which TSSs are meaningful, and which transition relations do they specify?*" were reviewed. The "most general solution without undesirable properties" is the well-founded semantics [14], which was introduced in the setting of logic programming, and later adapted to TSSs [10]. We use *three-valued stable models*, introduced by Przymusinski [25], to associate an LTS to a TSS with negative premises.

Definition 8 ($\models$). *Let N be a set of expressions of the form $p \overset{a}{\not\longrightarrow}$ with $p \in T(\Sigma)$ and $a \in A$, and let $\mathtt{P}$ be a set of transitions. We write $\mathtt{P} \models N$ if for each $p \overset{a}{\not\longrightarrow} \in N$, $p \xrightarrow{a} q \notin \mathtt{P}$ for all $q \in T(\Sigma)$.*

Definition 9 (Three-valued stable model). *Let the collection of all transitions $p \xrightarrow{a} p'$ with $p, p' \in T(\Sigma)$ and $a \in A$ be partitioned into three disjoint sets: the set $\mathtt{C}$ of transitions that are certainly true, the set $\mathtt{U}$ of the transitions for which it is unknown whether or not they are true, and the set of remaining transitions that are false. Such a partitioning (which is determined by the pair $(\mathtt{C}, \mathtt{U})$) constitutes a* three-valued stable model *for a TSS R if:*

- *a transition $p \xrightarrow{a} p'$ is in $\mathtt{C}$ if and only if a proof from R exists for a closed transition rule $N/p \xrightarrow{a} p'$ where N contains only negative premises and $\mathtt{C} \cup \mathtt{U} \models N$;*
- *a transition $p \xrightarrow{a} p'$ is in $\mathtt{C} \cup \mathtt{U}$ if and only if a proof from R exists for a closed transition rule $N/p \xrightarrow{a} p'$ where N contains only negative premises and $\mathtt{C} \models N$.*

A TSS may allow more than one three-valued stable model.

Example 1. Assume two constants c and d, and an action a. The TSS consisting of the transition rules $d \overset{a}{\not\longrightarrow} /c \xrightarrow{a} c$ and $c \overset{a}{\not\longrightarrow} /d \xrightarrow{a} d$ allows three three-valued stable models: $(\emptyset, \{c \xrightarrow{a} c, d \xrightarrow{a} d\})$, $(\{c \xrightarrow{a} c\}, \emptyset)$ and $(\{d \xrightarrow{a} d\}, \emptyset)$.

Przymusinski [25] noted that each TSS allows a *least* three-valued stable model, in the sense that the set $\mathtt{U}$ of unknown transitions is maximal; this least three-valued stable model coincides with the well-founded semantics from [14]. In this paper, the LTS specified by a TSS is the set $\mathtt{C}$ in its least three-valued stable model.

If R is a TSS, and the TSS R' is obtained by eliminating all negative premises from transition rules in R, then the LTS associated to R is included in the LTS associated to R'. So if the LTS associated to R' is bounded nondeterministic, then the LTS associated to R is bounded nondeterministic too.

The bounded nondeterminism format presented in this paper does not impose restrictions on negative premises. Therefore, we take the liberty to ignore negative premises, and focus on TSSs that only contain positive premises. If a TSS does not contain negative premises, then the set $\mathtt{C}$ in its least three-valued stable model coincides with the set of provable transitions.

However, by the observation above, the theorem that the LTS associated to a TSS in bounded nondeterminism format is bounded nondeterministic does apply to TSSs containing negative premises.

3 Bounded nondeterminism format

We present a new syntactic format for TSSs, called the *bounded nondeterminism* format, with the aim to guarantee that the LTS associated to a TSS is bounded nondeterministic.

Definition 10 (Bounded nondeterminism format). *A transition rule is in* bounded nondeterminism *format if:*

1. *all variables occurring in the left-hand side of its positive premises also occur in its source, and*
2. *all variables occurring in its target also occur in its source or in the right-hand side of one of its positive premises.*

A TSS is in bounded nondeterminism format if all its transition rules are.

Two counter-examples are given to show that the syntactic restrictions of the bounded nondeterminism format are both necessary. In each counter-example, by violating one of the conditions, it is shown that the associated LTS need not be bounded nondeterministic. The next example shows that each variable in the target must occur in the source or in the right-hand side of a positive premise.

Example 2. Let Σ consist of a constant c and a unary function symbol f, let $A = \{a\}$ and consider the TSS

$$\frac{}{c \xrightarrow{a} y}$$

The first restriction of the bounded nondeterminism format is satisfied, as there are no (variables in left-hand sides of) positive premises. However, the second restriction is violated: the variable y in the target occurs neither in the source nor in the right-hand side of a positive premise.

$c \xrightarrow{a} f^n(c)$ for $n \in \mathbb{N}$, so the state c is infinitely branching.

The next example shows that each variable at the left-hand side of a positive premise must occur in the source.

Example 3. Let Σ consist of a constant c and a unary function symbol f, let $A = \{a\}$, and consider the TSS

$$\frac{f(y) \xrightarrow{a} y}{c \xrightarrow{a} y} \qquad \frac{}{f(p) \xrightarrow{a} p}, \qquad p \in T(\Sigma).$$

The second restriction of the bounded nondeterminism format is satisfied, as in each transition rule, variables in the target also occur at the right-hand side of a positive premise. However, the first restriction is violated: in the first transition rule, the variable y at the left-hand side of the positive premise does not occur in the source.

$c \xrightarrow{a} p$ for $p \in T(\Sigma)$, so the state c is infinitely branching.

3.1 Boundedness and strict stratifications

The bounded nondeterminism format is not yet sufficient to guarantee bounded nondeterminism, as is shown by the next example. We therefore proceed to present two key notions for the main theorem, namely *bounded* TSSs and *strict stratifications*.

Example 4. Let Σ consist of a single constant c, let A be infinite, and consider the TSS consisting of the transition rules

$$\frac{}{c \xrightarrow{a} c}, \qquad a \in A.$$

This TSS is in bounded nondeterminism format. However, the state c in the associated LTS is infinitely branching.

Clearly, this violation of bounded nondeterminism is caused by the fact that there are infinitely many transition rules with the same source. Prohibiting this in a simple-minded way would exclude many TSSs from the literature that do generate bounded nondeterministic LTSs. For example, the TSS of Basic Process Algebra [4] (see Section 4.1) contains the following transition rules for the binary alternative composition operator $+$:

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

Again, if A is infinite, then there are infinitely many transition rules with the same source. Still, the LTS associated to Basic Process Algebra is bounded non-deterministic, which is due to the positive premise $x \xrightarrow{a} x'$; if a term p can perform only finitely many transitions, then the transition rules above give rise to only finitely many transitions for terms $p + q$.

We proceed to formalise this observation. The *type* of a transition rule consists of its source t together with the set of transition labels of positive premises with their left-hand side in a predefined finite set of terms. This latter set, denoted by $\eta(t)$, may depend on the source t.

Definition 11 (η-type). *Let R be a TSS, and let η map each term in $\mathbb{T}(\Sigma)$ to a finite subset of $\mathbb{T}(\Sigma)$. Consider a transition rule $\rho \in R$ with source t and positive premises $\{t_i \xrightarrow{a_i} t_i' \mid i \in I\}$. The mapping $\varphi : \eta(t) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$ is defined as follows:*

$$\varphi(u) = \{a_i \mid i \in I \wedge t_i = u\}.$$

*$\langle t, \varphi \rangle$ is said to be the η-*type *of ρ.*

Definition 12 (Uniform). *A TSS R is* uniform *if sources of transition rules in R that are α-convertible are always syntactically equal.*

It is not hard to see that each TSS R can be converted into a uniform TSS R' that generates the same LTS. In the same spirit as Vaandrager's proposal [30], we now define the notion of a bounded TSS.

Definition 13 (Bounded). *A TSS R is* bounded *if it is uniform and for each η-type the corresponding set of transition rules in R is finite.*

Note that the TSS in Example 4 is not bounded; for each mapping $\eta : \mathbb{T}(\Sigma) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$, all transition rules (and there are infinitely many) have the same η-type $\langle c, \varphi \rangle$ where φ maps each term in $\eta(c)$ to $\emptyset$. Also note that the TSSs in Examples 2 and 3 are bounded.

In the main theorem, we will restrict ourselves to bounded TSSs. The next example shows that in order to guarantee that the LTS associated to a TSS is bounded nondeterministic, it is important that η maps each term to a *finite* set of terms.

Example 5. Let Σ consist of constants from $A \cup \{c\}$ where A is infinite and $c \notin A$, and consider the TSS

$$\frac{}{a \xrightarrow{a} a} \qquad \frac{a \xrightarrow{a} y}{c \xrightarrow{a} y}, \qquad a \in A.$$

Since $c \xrightarrow{a} a$ for $a \in A$, the associated LTS is not bounded nondeterministic.

The TSS is in bounded nondeterminism format. Furthermore, if we were allowed to define $\eta(c) = A$, then the transition rules with source c would all have different η-types. However, since $\eta(c)$ is required to be a finite set, the TSS is not bounded; infinitely many transition rules will have the same η-type $\langle c, \varphi \rangle$ where φ maps each term in $\eta(c)$ to $\emptyset$.

The next example shows that it is important that bounded TSSs are uniform.

Example 6. Let Σ consist of a constant c, let A consist of actions a_x for $x \in V$, and consider the TSS

$$\frac{}{x \xrightarrow{a_x} x}, \qquad x \in V.$$

Since $c \xrightarrow{a_x} c$ for $x \in V$, the associated LTS is not bounded nondeterministic.

The TSS is in bounded nondeterminism format. Furthermore, since the sources of the transition rules are all distinct, they have different types. However, the TSS is not uniform.

The next example shows that the liberty to choose a different finite set $\eta(t)$ for each $t \in \mathbb{T}(\Sigma)$ expands the class of bounded TSSs.

Example 7. Let A be infinite, and let Σ consist of constants from A together with unary function symbols f_n for $n \in \mathbb{N}$. Consider the TSS

$$\frac{}{a \xrightarrow{a} a} \qquad \frac{x \xrightarrow{a} x}{f_0(x) \xrightarrow{a} x} \qquad \frac{f_n(x) \xrightarrow{a} x}{f_{n+1}(x) \xrightarrow{a} x}, \qquad a \in A,\ n \in \mathbb{N}.$$

This TSS is bounded if we take $\eta(a) = \emptyset$ for $a \in A$, $\eta(f_0(x)) = \{x\}$, and $\eta(f_{n+1}(x)) = \{f_n(x)\}$ for $n \in \mathbb{N}$. However, if $\eta(t)$ had been required to be the same finite set for all terms t, then this TSS would not have been bounded.

The restriction to bounded TSSs in bounded nondeterminism format is not yet sufficient to guarantee bounded nondeterminism, as is shown by the next example.

Example 8. Let Σ consist of a constant c and a unary function symbol f, let $A = \{a\}$, and consider the TSS

$$\frac{}{c \xrightarrow{a} c} \qquad \frac{x \xrightarrow{a} y}{x \xrightarrow{a} f(y)}$$

Clearly, this TSS is bounded and in bounded nondeterminism format. However, $c \xrightarrow{a} f^n(c)$ for $n \in \mathbb{N}$, so the state c is infinitely branching.

The next definition introduces the notion of a *strict stratification* of a TSS. Strict stratifications were used by Groote [17] to give meaning to a TSS with negative premises. For our purpose, the definition of a strict stratification will be slightly modified: here, a strict stratification maps closed terms (instead of transitions) to ordinal numbers.

Definition 14 (Strict stratification). *A mapping S from $T(\Sigma)$ to ordinal numbers is a* strict stratification *of a TSS R if for every transition rule $H/t \xrightarrow{a} t'$ and for every closed substitution σ:*

$$S(\sigma(u)) < S(\sigma(t)) \text{ for all } u \xrightarrow{b} u' \in H.$$

Note that the TSS in Example 8 does not allow a strict stratification. Also note that the TSSs in Examples 2, 3, 4, 5 and 6 do allow a strict stratification. In particular, a strict stratification S for the TSS in Example 3 is defined by $S(c) = 1$ and $S(f(p)) = 0$ for $p \in T(\Sigma)$.

3.2 The main theorem

Finally we are in a position to formulate and prove our main theorem. The TSSs in Examples 2, 3, 4, 5, 6 and 8 each violate only one of the restrictions in Theorem 1, thus showing that all restrictions are needed.

Theorem 1. *Let R be a bounded TSS in bounded nondeterminism format that allows a strict stratification S. The LTS associated to R is bounded nondeterministic.*

Proof: Since R is bounded, there is a mapping $\eta : \mathbb{T}(\Sigma) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$ such that for each η-type the corresponding set of transition rules in R is finite.

We prove that each $p \in T(\Sigma)$ is finitely branching, by induction on $S(p)$.

1. Let $S(p) = 0$.
   Since R is uniform, there are only finitely many sources t_i in R and closed substitutions $\sigma_i : \mathit{var}(t_i) \to T(\Sigma)$, where i ranges over some finite index set I, such that $\sigma_i(t_i) = p$.
   $S(\sigma_i(t_i)) = 0$ for $i \in I$. Since S is a strict stratification for R, the transition rules in R with source t_i have the form:
   $$\frac{}{t_i \xrightarrow{a_{ij}} t'_{ij}}, \qquad\qquad j \in J_i.$$
   These transition rules all have η-type $\langle t_i, \varphi\rangle$ where φ maps each term in $\eta(t_i)$ to $\emptyset$. By boundedness of R there are only finitely many such transition rules, meaning that the index sets J_i are finite.
   Since R is in bounded nondeterminism format, $\mathit{var}(t'_{ij}) \subseteq \mathit{var}(t_i)$ for $i \in I$ and $j \in J_i$. This implies that the transition rules with a source in $\{t_i \mid i \in I\}$ give rise to only finitely many outgoing transitions of p: $\sigma_i(t_i) \xrightarrow{a_{ij}} \sigma_i(t'_{ij})$ for $i \in I$ and $j \in J_i$.
   Concluding, p is finitely branching.
2. Let $S(p) > 0$. By the induction hypothesis, each $q \in T(\Sigma)$ with $S(q) < S(p)$ is finitely branching. We prove that p is finitely branching.
   Since R is uniform, there are only finitely many sources t_i in R and closed substitutions $\sigma_i : \mathit{var}(t_i) \to T(\Sigma)$, where i ranges over some finite index set I, such that $\sigma_i(t_i) = p$.
   For each $i \in I$ and $\varphi : \eta(t_i) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$, the transition rules with η-type $\langle t_i, \varphi\rangle$ have the form
   $$\frac{\{u_k \xrightarrow{b_k} u'_k \mid k \in K_j\}}{t_i \xrightarrow{a_{ij}} t'_j}, \qquad\qquad j \in J_{i\varphi}$$
   (where the $J_{i\varphi}$ and K_j are taken to be disjoint). By boundedness of R there are only finitely many such transition rules, meaning that the index sets $J_{i\varphi}$ are finite.
   Fix an $i \in I$ and $\varphi : \eta(t_i) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$. Since R is in bounded nondeterminism format, $\mathit{var}(u_k) \subseteq \mathit{var}(t_i)$ for $j \in J_{i\varphi}$ and $k \in K_j$, so the $\sigma_i(u_k)$ are closed terms. Furthermore, as S is a strict stratification for R, $S(\sigma_i(u_k)) < S(\sigma_i(t_i))$, so by the induction hypothesis the $\sigma_i(u_k)$ are finitely branching. This means that for each $j \in J_{i\varphi}$ there are only finitely many closed substitutions $\tau_\ell : (\bigcup_{k\in K_j} \mathit{var}(u'_k))\backslash \mathit{var}(t_i) \to T(\Sigma)$, where ℓ ranges over some finite index set L_j, such that $\sigma_i(u_k) \xrightarrow{b_k} \tau_\ell\sigma_i(u'_k)$ is provable from R for $k \in K_j$.
   R is in bounded nondeterminism format, so $\mathit{var}(t'_j) \subseteq \mathit{var}(t_i) \cup (\bigcup_{k\in K_j} \mathit{var}(u'_k))$ for $j \in J_{i\varphi}$. Thus, the $\tau_\ell\sigma_i(t'_j)$ are closed terms. Hence, for the fixed $i \in I$ and

mapping φ, the transition rules with η-type $\langle t_i, \varphi\rangle$ give rise to only finitely many outgoing transitions of p: $\sigma_i(t_i) \xrightarrow{a_{ij}} \tau_\ell \sigma_i(t'_{ij})$ for $j \in J_{i\varphi}$ and $\ell \in L_j$.
There are only finitely many transition rules with η-type $\langle t_i, \varphi\rangle$, and the index set I for i is finite. So the only remaining possible cause for infinite branching of p is that, for $i \in I$, there may be infinitely many different mappings $\varphi : \eta(t_i) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$. However, we show that, for any $i \in I$, there are only finitely many mappings φ such that for all $v \in \eta(t_i)$, $\sigma_i(v)$ can perform a c-transition for each $c \in \varphi(v)$. Since each transition rule of type $\langle t_i, \varphi\rangle$ contains a premise of the form $v \xrightarrow{c} v'$ for each $v \in \eta(t_i)$ and $c \in \varphi(v)$, it follows that there are only finitely many mappings φ such that the transition rules of type $\langle t_i, \varphi\rangle$ give rise to transitions of p.
Assume towards a contradiction that, for some $i \in I$, there are infinitely many mappings $\varphi_n : \eta(t_i) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$ with $n \in \mathbb{N}$ such that for all $v \in \eta(t_i)$, $\sigma_i(v)$ can perform a c-transition for each $c \in \varphi(v)$. Since $\eta(t_i)$ is finite, and the φ_n are distinct, there exists a $v_0 \in \eta(t_i)$ such that $\bigcup_{n=1}^{\infty} \varphi_n(v_0)$ is infinite. By assumption, $\sigma_i(v_0)$ can perform a c-transition for each $c \in \bigcup_{k=1}^{\infty} \varphi_{i_k}(v_0)$, so $\sigma_i(v_0)$ is infinitely branching. Since v_0 is the left-hand side of (infinitely many) transition rules of type $\langle t_i, \varphi_n\rangle$ with $\varphi_n(v_0) \neq \emptyset$, we have $S(\sigma_i(v_0)) < S(p)$. This contradicts the induction hypothesis.
Concluding, p is finitely branching.

In short, the LTS associated to R is bounded nondeterministic because every state $p \in T(\Sigma)$ is finitely branching. □

4 Applications

The bounded nondeterminism format from Definition 10 subsumes the De Simone format [26] and the GSOS format [9]. Furthermore, TSSs in De Simone or GSOS format always allow a strict stratification (see Definition 14). So by Theorem 1, bounded TSSs in De Simone or GSOS format generate bounded nondeterministic LTSs. See for instance [1] for examples of TSSs from the literature that satisfy the restrictions of Theorem 1. In this section we consider three such examples.

4.1 Basic process algebra

In this section we study the Basic Process Algebra with empty process (BPA$_{\delta\epsilon}$) [4]. This process algebra will serve as the basis for examples to come. First we show that the LTS associated to the TSS of BPA$_{\delta\epsilon}$ is bounded nondeterministic.

The signature Σ of BPA$_{\delta\epsilon}$ is defined as follows. *Act* is a non-empty set of constants representing atomic actions, and δ and ϵ are constants representing deadlock and successful termination, respectively. The set A of transition labels consists of $Act \cup \{\checkmark\}$, where $\checkmark$ represents successful termination. Furthermore, there are two binary operators: $p \cdot q$ denotes sequential composition and $p + q$ alternative composition. The TSS of BPA$_{\delta\epsilon}$ is given in Table 1.

Table 1. TSS of $\mathrm{BPA}_{\delta\epsilon}$ ($\alpha \in Act$, $a \in Act \cup \{\checkmark\}$)

$$\frac{}{\epsilon \xrightarrow{\checkmark} \delta} \qquad \frac{}{\alpha \xrightarrow{\alpha} \epsilon}$$

$$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \quad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'} \quad \frac{x \xrightarrow{\alpha} x'}{x \cdot y \xrightarrow{\alpha} x' \cdot y} \quad \frac{x \xrightarrow{\checkmark} x' \quad y \xrightarrow{a} y'}{x \cdot y \xrightarrow{a} y'}$$

Corollary 1. *The LTS associated to $BPA_{\delta\epsilon}$ is bounded nondeterministic.*

Proof: The TSS of $\mathrm{BPA}_{\delta\epsilon}$ is in bounded nondeterminism format. That is, for each transition rule, all variables occurring in the left-hand side of its positive premises also occur in its source, and all variables occurring in its target also occur in its source or in the right-hand side of one of its positive premises.

The TSS of $\mathrm{BPA}_{\delta\epsilon}$ is uniform. Let $\eta : \mathbb{T}(\Sigma) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$ with $\eta(a) = \emptyset$ for $a \in Act \cup \{\checkmark\}$, $\eta(x + y) = \{x, y\}$ and $\eta(x \cdot y) = \{x, y\}$. For each η-type there is at most one corresponding transition rule in Table 1. Hence, the TSS of $\mathrm{BPA}_{\delta\epsilon}$ is bounded.

Let $S : T(\Sigma) \to \mathbb{N}$, where $S(p)$ produces the number of occurrences of the binary operators $+$ and $\cdot$ in p. Clearly $S(p) < S(p+q)$, $S(q) < s(p+q)$, $S(p) < S(p \cdot q)$ and $S(q) < S(p \cdot q)$. So S is a strict stratification for the TSS of $\mathrm{BPA}_{\delta\epsilon}$.

So by Theorem 1, the LTS associated to $\mathrm{BPA}_{\delta\epsilon}$ is bounded nondeterministic. □

4.2 Binary Kleene star

The *binary Kleene star* p^*q [19] repeatedly executes p until it executes q. Using Theorem 1, we prove that the LTS associated to $\mathrm{BPA}^*_{\delta\epsilon}$ is bounded nondeterministic. We note that one of transition rules for the binary Kleene star below is not in De Simone format, so that Vaandrager's result in [30] on bounded nondeterminism of LTSs associated to bounded TSSs in De Simone format does not apply here.

The transition rules for the binary Kleene star, taken from [3], are as follows, where α ranges over Act and a over $Act \cup \{\checkmark\}$:

$$\frac{x \xrightarrow{\alpha} x'}{x^*y \xrightarrow{\alpha} x' \cdot (x^*y)} \qquad \frac{y \xrightarrow{a} y'}{x^*y \xrightarrow{a} y'}$$

These transition rules are added to the TSS of $\mathrm{BPA}_{\delta\epsilon}$ in Table 1.

Corollary 2. *The LTS associated to $BPA^*_{\delta\epsilon}$ is bounded nondeterministic.*

Proof: The transition rules for the binary Kleene star are in bounded nondeterminism format. Furthermore, it is easy to see that the TSS of $\mathrm{BPA}^*_{\delta\epsilon}$ is bounded and allows a strict stratification. So by Theorem 1, the LTS associated to this TSS is bounded nondeterministic. □

4.3 Priority

Finally, we study a TSS that includes negative premises. *Priority* [2] is a unary function symbol that assumes an ordering on transition labels. The term $\Theta(p)$ executes the transitions of p, with the restriction that a transition $p \xrightarrow{a} p'$ only gives rise to a transition $\Theta(p) \xrightarrow{a} \Theta(p')$ if there does not exist a transition $p \xrightarrow{b} p''$ with $a < b$. This intuition is captured by the transition rule for the priority operator below, which is added to the TSS of BPA$_{\delta\varepsilon}$ in Table 1.

$$\frac{x \xrightarrow{a} x' \qquad x \not\xrightarrow{b} \text{ for } a < b}{\Theta(x) \xrightarrow{a} \Theta(x')}$$

Corollary 3. *The LTS associated to BPA$^{\Theta}_{\delta\epsilon}$ is bounded nondeterministic.*

Proof: The transition rule for priority is in bounded nondeterminism format. Furthermore, it is easy to see that the TSS of BPA$^{\Theta}_{\delta\epsilon}$ is bounded and allows a strict stratification. So by Theorem 1, the LTS associated to this TSS is bounded nondeterministic. □

5 Counter-examples

In this section we investigate some TSSs from the literature for which the associated LTS is not bounded nondeterministic. Thus, these TSSs do not satisfy all the requirements in Theorem 1.

5.1 Replication

The *parallel* operator $p \parallel q$ executes p and q in parallel.

$$\frac{x \xrightarrow{\alpha} x'}{x \parallel y \xrightarrow{\alpha} x' \parallel y} \qquad \frac{y \xrightarrow{\alpha} y'}{x \parallel y \xrightarrow{\alpha} x \parallel y'} \qquad \frac{x \xrightarrow{\checkmark} x' \quad y \xrightarrow{\checkmark} y'}{x \parallel y \xrightarrow{\checkmark} x' \parallel y'}$$

These transition rules are in bounded nondeterminism format, and the TSS of BPA$_{\delta\epsilon}$ extended with the three transition rules above is bounded and allows a strict stratification. Hence, by Theorem 1, the LTS associated to this TSS is finitely branching.

The *replication* operator $!p$, which is used to express recursion in the π-calculus [23], can be thought of as an infinite parallel composition $p \parallel p \parallel \cdots$. It is defined by

$$\frac{x \parallel !x \xrightarrow{\alpha} x'}{!x \xrightarrow{\alpha} x'}$$

We extend BPA$_{\delta\epsilon}$ with both the parallel operator and replication. The LTS associated to this process algebra is not bounded nondeterministic. For example, if $\alpha \in Act$, then $!\alpha \xrightarrow{\alpha} p_n$ for $n \in \mathbb{N}$, where $p_0 = \epsilon \parallel !\alpha$ and $p_{n+1} = \alpha \parallel p_n$.

Clearly, the TSS for this process algebra is in bounded nondeterminism format and bounded. However, the transition rules for replication disallow a strict stratification.

5.2 Unguarded recursion

A recursive specification consists of a finite set of recursive equations

$$\begin{aligned} X_1 &= t_1(X_1, \ldots, X_k) \\ X_2 &= t_2(X_1, \ldots, X_k) \\ &\vdots \\ X_k &= t_k(X_1, \ldots, X_k) \end{aligned}$$

where the left-hand sides X_i are recursive variables, and the right-hand sides $t_i(X_1, \ldots, X_k)$ are process terms in BPA with possible occurrences of $X_1, \ldots, X_k$.

If E is a recursive specification, and X a recursive variable in E, then $< X|E >$ denotes the process that has to be substituted for X in the solution for E. The transition rules for recursion, taken from [15], are of the form

$$\frac{t_i(< X_1|E >, \ldots, < X_k|E >) \xrightarrow{a} y}{< X_i|E > \xrightarrow{a} y}$$

We extend the TSS of $\mathrm{BPA}_{\delta\epsilon}$ with the transition rules for recursion.

The LTS associated to the TSS of $\mathrm{BPA}_{\delta\epsilon}$ with recursion is not bounded nondeterministic. For example, consider the (unguarded) recursive specification E consisting of the recursive equation $X = X \cdot \alpha + \alpha$. Then $< X|E >$ can perform an α-transition to α^n for any $n \in \mathbb{N}$ (cf. [15]).

Clearly, the TSS for this process algebra is in bounded nondeterminism format and bounded. However, the transition rules for recursion disallow a strict stratification.

5.3 Timed deadlock

Deadlock in real-time Basic Process Algebra [20], denoted by $\delta[r]$, has an infinite "ultimate delay" of $r \in \mathbb{R}_{>0}$. This can be expressed by transition rules

$$\frac{}{\delta[r] \xrightarrow{\delta[s]} \checkmark}$$

for $0 < s < r$, where s ranges over the real numbers. Then $\delta[r]$ is infinitely branching for $r \in \mathbb{R}_{>0}$, which is due to the fact that its transition rules are not bounded.

Other examples of operators that give rise to infinite branching due to unboundedness of their transition rules are alternative quantification [21] and value passing [22].

References

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

2. J. Baeten, J. Bergstra, and J.W. Klop. Syntax and defining equations for an interrupt mechanism in process algebra. *Fundamenta Informaticae*, IX(2):127–168, 1986.
3. J. Bergstra, I. Bethke, and A. Ponse. Process algebra with iteration and nesting. *The Computer Journal*, 37(4):243–258, 1994.
4. J. Bergstra and J.W. Klop. Process algebra for synchronous communication. *Information and Control*, 60(1/3):109–137, 1984.
5. B. Bloom. CHOCOLATE: Calculi of Higher Order COmmunication and LAmbda TErms. In *Conference Record 21st ACM Symposium on Principles of Programming Languages,* Portland, Oregon, pages 339–347. ACM Press, 1994.
6. B. Bloom. When is partial trace equivalence adequate? *Formal Aspects of Computing*, 6(3):317–338, 1994.
7. B. Bloom. Structural operational semantics for weak bisimulations. *Theoretical Computer Science*, 146(1/2):25–68, 1995.
8. B. Bloom, W. Fokkink, and R. van Glabbeek. Precongruence formats for decorated trace preorders. In *Proceedings 15th Symposium on Logic in Computer Science,* Santa Barbara, California, pages 107–118. IEEE Computer Society Press, 2000.
9. B. Bloom, S. Istrail, and A. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268, 1995.
10. R. Bol and J.F. Groote. The meaning of negative premises in transition system specifications. *Journal of the ACM*, 43(5):863–914, 1996.
11. P. D'Argenio and C. Verhoef. A general conservative extension theorem in process algebras with inequalities. *Theoretical Computer Science*, 177(2):351–380, 1997.
12. W. Fokkink. Language preorder as a precongruence. *Theoretical Computer Science*, 243(1/2):391–408, 2000.
13. W. Fokkink. Rooted branching bisimulation as a congruence. *Journal of Computer and System Sciences*, 60(1):13–37, 2000.
14. A. van Gelder, K. Ross, and J. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.
15. R. van Glabbeek. Bounded nondeterminism and the approximation induction principle in process algebra. In F.J. Brandenburg, G. Vidal-Naquet, and M. Wirsing, editors, *Proceedings 4th Symposium on Theoretical Aspects of Computer Science,* Passau, Germany, volume 247 of *Lecture Notes in Computer Science*, pages 336–347. Springer, 1987.
16. R. van Glabbeek. The meaning of negative premises in transition system specifications II (extended abstract). In F. Meyer auf der Heide and B. Monien, editors, *Proceedings 23rd Colloquium on Automata, Languages and Programming,* Paderborn, Germany, volume 1099 of *Lecture Notes in Computer Science*, pages 502–513. Springer, 1996.
17. J.F. Groote. Transition system specifications with negative premises. *Theoretical Computer Science*, 118(2):263–299, 1993.
18. J.F. Groote and F. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, 1992.
19. S.C. Kleene. Representation of events in nerve nets and finite automata. In C.E. Shannon and J. McCarthy, editors, *Automata Studies*, pages 3–41. Princeton University Press, 1956.
20. A.S. Klusener. *Models and Axioms for a Fragment of Real Time Process Algebra.* PhD thesis, Eindhoven University of Technology, 1993.
21. S.P. Luttik. *Choice Quantification in Process Algebra.* PhD thesis, University of Amsterdam, 2002.

22. R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980.
23. R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, part I + II. *Information and Computation*, 100(1):1–77, 1992.
24. G. Plotkin. A structural approach to operational semantics. Report DAIMI FN-19, Computer Science Department, Aarhus University, 1981.
25. T. Przymusinski. The well-founded semantics coincides with the three-valued stable semantics. *Fundamenta Informaticae*, 13(4):445–463, 1990.
26. R. de Simone. Higher-level synchronising devices in MEIJE–SCCS. *Theoretical Computer Science*, 37:245–267, 1985.
27. I. Ulidowski. Equivalences on observable processes. In *Proceedings 7th Symposium on Logic in Computer Science,* Santa Cruz, California, pages 148–159. IEEE Computer Society Press, 1992.
28. I. Ulidowski and I.C.C. Phillips. Ordered SOS rules and process languages for branching and eager bisimulations. *Information and Computation*, 178(1):180–213, 2002.
29. F. Vaandrager. On the relationship between process algebra and input/output automata (extended abstract). In *Proceedings 6th Symposium on Logic in Computer Science,* Amsterdam, The Netherlands, pages 387–398. IEEE Computer Society Press, 1991.
30. F. Vaandrager. Expressiveness results for process algebras. In J.W. de Bakker, W.P. de Roever, and G. Rozenberg, editors, *Proceedings REX Workshop on Semantics: Foundations and Applications,* Beekbergen, The Netherlands, volume 666 of *Lecture Notes in Computer Science*, pages 609–638. Springer, 1993.
31. C. Verhoef. A general conservative extension theorem in process algebra. In E.-R. Olderog, editor, *Proceedings 3rd IFIP Working Conference on Programming Concepts, Methods and Calculi,* San Miniato, Italy, IFIP Transactions A-56, pages 149–168. Elsevier, 1994.