A unified rule format for bounded nondeterminism in SOS with terms as labels[☆]

L. Aceto[a,*], I. Fábregas[a], A. García-Pérez[a], A. Ingólfsdóttir[a]

[a]*ICE-TCS, School of Computer Science, Reykjavik University, Menntavegur 1, IS-101, Reykjavik, Iceland.*

Abstract

We present a unified rule format for structural operational semantics with terms as labels that guarantees that the associated labelled transition system has some bounded-nondeterminism property. The properties we consider include finite branching, initials finiteness and image finiteness.

Keywords: rule formats, bounded nondeterminism, structural operational semantics, labelled transition systems

1. Introduction

Structural operational semantics (SOS) [30, 32] is a widely used formalism for defining the formal semantics of computer programs and for proving properties of the corresponding programming languages. In the SOS formalism a transition system specification (TSS) [21], which consists of a signature together with a set of inference rules, specifies a labelled transition system (LTS) [24] whose states (i.e., processes) are closed terms over the signature and whose transitions are those that can be proved using the inference rules.

Rule formats [3, 28] are (syntactically checkable) restrictions on the inference rules of a TSS that guarantee some useful property of the associated LTS. In this paper we focus on the finiteness of the number of outgoing transition from a given state, which is referred to as bounded nondeterminism in [19]. Broadly, bounded nondeterminism is taken as a synonym of finite branching [16]. Finite branching breaks down into the more elementary properties of initials finiteness and image finiteness [1, 23]. Intuitively, initials finiteness focuses on the finiteness of the number of initial actions a process can perform, whereas image finiteness focuses on the finiteness of the number of states a process can reach by performing a given action.

The wealth of research on various kinds of bounded nondeterminism properties has been motivated by results witnessing desirable features enjoyed by the LTSs that afford them. For example, bisimulation-based preorders over finitely branching LTSs with divergence are "finitary", and this property makes it possible to use them as behavioural yardsticks to prove full abstraction results for denotational semantics for process calculi—see, for instance, [2, 5]. This state of affairs is in stark contrast with the general argument presented in [6] to the effect that no continuous least fixed-point semantics, satisfying a certain full abstraction property, can exist in the presence of countable nondeterminism. Another classic example of the use of a bounded nondeterminism property from the literature on concurrency theory and modal logic is provided by the Hennessy-Milner Theorem, to the effect that two processes in an image-finite LTS are bisimilar exactly

[☆]This research has been supported by the project 'Nominal Structural Operational Semantics' (nr. 141558-051) of the Icelandic Research Fund, the projects StrongSoft (TIN2012-39391-C04–04) and TRACES (TIN2015-67522-C3-3-R) of the Spanish Ministerio de Economía y Competitividad, the project 001-ABEL-CM-2013 within the NILS Science and Sustainability Programme, and the grant GR3/14 of the Universidad Complutense de Madrid - Banco Santander.

[*]Corresponding author

Email addresses: `luca@ru.is` (L. Aceto), `fabregas@ucm.es` (I. Fábregas), `alvarog@ru.is` (A. García-Pérez), `annai@ru.is` (A. Ingólfsdóttir)

when they satisfy the same formulae in Hennessy-Milner logic [9, 23]. Languages whose semantics is given by LTSs satisfying some bounded nondeterminism property also tend to have lower "complexity" than those exhibiting unbounded nondeterminism. For instance, Chandra has shown in [12] that the halting problem for programs with unbounded nondeterminism is complete for Π^1_1 and is therefore of higher complexity than truth in the standard model of the natural numbers. Indeed, in the presence of infinite nondeterminism, quoting from [12, page 131]:

> total correctness (or even absence of divergence) is not expressible in first-order arithmetic. This may be contrasted with the results in [22] where the model does not allow infinite nondeterminism.

In the light of the above-mentioned work, it should come as no surprise that the study of rule formats for bounded nondeterminism is far from being new. Finite branching of the associated LTS is one of the sanity properties guaranteed by the GSOS format of [11]. Vaandrager [37] introduced a rule format for SOS, based on the de Simone format [14], that guarantees that the associated LTS is finite branching. Following Vaandrager, Bloom [10] introduced a rule format for his CHOCOLATE formalism that also guarantees a finite-branching LTS. Finally, Fokkink and Vu [16] introduced yet another, less restrictive rule format for SOS, which relies on an adaptation of the notion of strict stratification from [20], and showed that a TSS in this format induces an LTS that is finite branching. Recently, we adapted the work in [16] and presented rule formats also for initials finiteness and image finiteness [4].

Our work in this paper takes this programme further by advancing on three fronts:

(i) We extend the applicability of the rule formats for bounded nondeterminism to the case of higher-order processes by tackling one of the most prominent challenges in that setting, namely that of allowing arbitrary terms as labels in transitions [8, 27].

(ii) We consider a family of bounded-nondeterminism properties that are more elementary than finite branching, and which include image finiteness and initials finiteness [1]. We introduce a unified rule format that can be instantiated to each of twelve bounded-nondeterminism conditions.

(iii) For each rule in a TSS, our rule format uses *global* information in order to determine whether the rule is a so-called junk rule (i.e., a rule that is never involved in proving some transition). Our rule format filters out more junk rules than the one in [16].

Previous works on rule formats for bounded nondeterminism, such as that in [16], require the labels in transitions to be ground. However, SOS for higher-order processes [8, 27] needs to consider arbitrary terms as labels and the use of variables in labels must be suitably restricted to guarantee the finiteness property of interest. Indeed, the unconstrained occurrence of variables in a label could break various bounded nondeterminism properties because a variable stands for an arbitrary process of the LTS. As an example, an axiom $f(x) \xrightarrow{y} f(x)$, where x and y are variables, could be used to prove any transition of the form $f(p) \xrightarrow{q} f(p)$, where p and q are arbitrary closed terms. Therefore any term of the form $f(p)$ would be neither finite branching nor initials finite. On the other hand, the use of y in that axiom does not jeopardize image finiteness.

In order to deal with (i) and (ii) above in a uniform way for all the twelve bounded nondeterminism properties studied in the paper, we consider a family of representational transformations for transitions that, instead of *triadic* transition formulae $t \xrightarrow{l} t'$, consider *dyadic* transition formulae $s \longrightarrow r$ in which the label l occurs either as a part of the source s or of the target r. Building on [16], the dyadic transformations make it possible to define a general rule format (see Theorem 1), which can be instantiated with the particular dyadic transformation that corresponds to a bounded nondeterminism property of our choice.

The dyadic transformations unveil a space of bounded-nondeterminism properties that has a rich structure (see Section 4). The space contains some elementary properties that are minimal in the "information" order. We explore the equivalences between the conjunction of certain pairs of properties and other properties that are derived, bringing forth a principled approach to bounded nondeterminism in all its possible variants.

In order to deal with the detection of junk rules, point (iii) above, we refine the η-types of [16], which are at the basis of the rule formats for bounded nondeterminism from [3, 16]. Junk rules are those that can never take part in a proof tree. The set of TSSs for which a rule format is applicable can be safely enlarged by relaxing the restrictions of the rule format on junk rules, since these rules cannot give rise to transitions. We introduce the S-types (see Definition 19), which use global information (i.e., information from every rule in a TSS) in order to detect more junk rules than the η-types of [16], which only use local information. A key ingredient of the S-types is a notion of *partial* strict stratification (see Definition 17) that allows some processes that unify with sources of premises in a rule to have undefined stratification order.

The rest of the paper is organized as follows. Section 2 collects standard background on the SOS formalism and unifies notation. Section 3 introduces the dyadic transformations, which provide a uniform framework to deal with TSSs with terms and labels and their bounded nondeterminism properties. Section 4 studies the structure of the space of bounded-nondeterminism properties unveiled by the dyadic transformations. Section 5 explains how the set of finitely branching TSSs that meet the conditions of the rule format from [16] can be enlarged by considering the so-called junk rules. Section 6 shows how to refine the η-types of [16] in order to filter out more junk rules. Section 7 introduces the S-types, which consider the previously mentioned refinements on the η-types and adapt them to dyadic TSSs. Section 8 presents rule formats that are based on the S-types guaranteeing finite branching of the LTS associated to a triadic TSS with terms as labels. (A variety of bounded nondeterminism properties can be guaranteed by picking different dyadic transformations.) Section 9 shows examples of the applicability of our rule format. In particular, we consider a TSS with terms as labels that implements a subset of CHOCS [27, 35] and use our format to show various bounded nondeterminism properties of its transition relations. Section 10 discusses avenues for future work and concludes. Throughout the paper, we use a variety of simple (but admittedly artificial) examples to motivate our definitions and main observations.

2. Preliminaries

In this section we review and adapt standard background on the structural operational semantics formalism (SOS for short) and unify notation. We follow the presentation in [16].

For a set S, we write $\mathcal{P}(S)$ for the collection of all the subsets of S, and $\mathcal{P}_\omega(S)$ for the collection of all the finite subsets of S.

Definition 1 (Signature and Term)**.** We assume a countably infinite set of variables V, ranged over by x, y, z. A signature Σ is a set of function symbols, disjoint from V, together with an arity map that assigns a natural number to each function symbol. We use f and g to range over the set of functions symbols in Σ. Function symbols of arity zero, which are ranged over by c and d, are called constants. Function symbols of arity one and two are called unary and binary functions respectively.

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

(i) each variable is a term, and

(ii) if f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

The function $\mathrm{var} : \mathbb{T}(\Sigma) \to \mathcal{P}_\omega(V)$ delivers, for a term t, the set of variables that occur in t. A term t is closed iff $\mathrm{var}(t) = \emptyset$. The set of closed terms over Σ, ranged over by p, q, is denoted by $\mathbb{C}(\Sigma)$.

Intuitively, a signature Σ collects the term constructors offered by a programming or specification language. In this paper we consider terms also as transition labels, where symbols l, m, n range over label terms.

Definition 2 (Formula)**.** The set of positive formulae over signature Σ is the set of triples $(t, l, t') \in \mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma)$. We use the more suggestive notation $t \xrightarrow{l} t'$ in lieu of (t, l, t'). The set of negative formulae over signature Σ is the set of pairs $(t, l) \in \mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma)$. We use the more suggestive notation $t \overset{l}{\not\longrightarrow}$ in lieu of (t, l).

Definition 3 (Substitution)**.** A substitution is a partial map $\sigma : V \to \mathbb{T}(\Sigma)$. The substitutions are ranged over by σ, τ. A substitution is closed if it maps variables to closed terms. A substitution extends to a map from terms to terms in the usual way, i.e., the term $\sigma(t)$ is obtained by replacing the occurrences in t of each variable x in the domain of σ by $\sigma(x)$. When applying substitutions σ and τ successively, we may abbreviate $\tau(\sigma(t))$ to $\tau\sigma(t)$. We say term u is a substitution instance of t iff there exists a substitution σ such that $\sigma(t) = u$.

In what follows, we will sometimes use the notation $\{x_i \mapsto t_i \mid i \in I\}$, where I is an index set and the x_i's are pairwise distinct variables, to denote the substitution that maps each x_i to the term t_i, with $i \in I$.

A substitution σ extends to formulae $t \xrightarrow{l} t'$ and $u \not\xrightarrow{l}$ in the usual way, by applying the substitution to each term component of the formula, i.e., $\sigma(t) \xrightarrow{\sigma(l)} \sigma(t')$ and $\sigma(u) \not\xrightarrow{\sigma(l)}$ respectively. The notion of substitution instance extends similarly.

Labelled transition systems [24] are a fundamental model of computation and are often used to describe the operational semantics of programming and specification languages (see, for instance, [25, 31, 32, 34]). We will consider classic labelled transition systems [24] where transitions are labelled with processes instead of actions.

Definition 4 (Labelled transition system with processes as labels)**.** Let Σ be a signature. A labelled transition system with processes as labels (LTS for short) is a pair $(\mathbb{C}(\Sigma), \to)$ where $\mathbb{C}(\Sigma)$ is the set of processes, i.e., closed terms over Σ, and $\longrightarrow \subseteq \mathbb{C}(\Sigma) \times \mathbb{C}(\Sigma) \times \mathbb{C}(\Sigma)$ is the set of transitions, i.e., closed positive formulae. We say that $p \xrightarrow{l} p'$ is a transition of the LTS iff $(p, l, p') \in \longrightarrow$.

Transition system specifications are sets of inference rules that can be used to prove the valid transitions between terms in a language.

Definition 5 (Transition system specification with terms as labels)**.** Let Σ be a signature. A transition rule (a rule, for short) ρ is of the form

$$\frac{H}{t \xrightarrow{l} t'}$$

(abbreviated as $H/t \xrightarrow{l} t'$) where H is a set of positive premises of the form $u \xrightarrow{m} u'$ and negative premises of the form $v \not\xrightarrow{n}$, and $t \xrightarrow{l} t'$ is the conclusion of the rule (with $t, t', u, u', v, l, m, n \in \mathbb{T}(\Sigma)$). We say t is the source, l is the label, and t' is the target of ρ. We say ρ is an axiom iff ρ has an empty set of premises, i.e., $H = \emptyset$.

A transition system specification with terms as labels (TSS for short) is a set of transition rules.

A substitution map extends to a rule ρ by applying the substitution to the formulae in ρ. The notion of substitution instance extends similarly to rules.

Definition 6 (Unify with a rule)**.** Let R be a TSS. We say that transition $p \xrightarrow{l} p'$ unifies with rule $\rho \in R$ iff ρ has conclusion $t \xrightarrow{m} t'$ and $p \xrightarrow{l} p'$ is a substitution instance of $t \xrightarrow{m} t'$.

Definition 7 (Proof tree)**.** Let R be a TSS without negative premises. A proof tree in R of a transition $p \xrightarrow{l} p'$ is an upwardly branching tree without paths of infinite length whose nodes are labelled by transitions such that

(i) the root is labelled by $p \xrightarrow{l} p'$, and

(ii) if K is the set of labels of the nodes directly above a node with label $q \xrightarrow{n} q'$, then $K/q \xrightarrow{n} q'$ is a substitution instance of some rule $H/t \xrightarrow{m} t' \in R$.

We say that $p \xrightarrow{l} p'$ is provable in R iff $p \xrightarrow{l} p'$ has a proof tree in R.

The set of provable transitions in R is the least set of transitions that satisfies the rules in R. Notice that if $p \xrightarrow{l} p'$ unifies with an axiom (i.e., a rule of the form $\emptyset/t \xrightarrow{m} t'$) then, trivially, $p \xrightarrow{l} p'$ has a proof tree in R which consists of a root node labelled by $p \xrightarrow{l} p'$.

A TSS with terms as labels and without negative premises induces an LTS with processes as labels in a straightforward way.

Definition 8 (TSS induces LTS)**.** Let R be a TSS with terms as labels and without negative premises, and let T be an LTS with processes as labels. We say that R induces T (or T is associated with R) iff the set of transitions of T is the set of provable transitions in R.

The phrases

- $p \xrightarrow{l} p'$ is provable in R,
- $p \xrightarrow{l} p'$ is a transition of T, and
- p can perform an l-transition to p' in T

are synonyms. For brevity, we may omit the R and/or the T when they are clear from the context.

In [33], Przymusinski introduced *three-valued stable models*, which can be used to associate an LTS to a TSS with negative premises. Each TSS has a least three-valued stable model, which coincides with the well-founded semantics from [18]. We consider the set of *sentences that are certainly true* in the least three-valued stable model, which, for a TSS without negative premises, coincides with the set of provable transitions in Definition 8. As Fokkink and Vu noticed in [16], if R is a TSS and R' is obtained by removing all the negative premises from the rules in R, then the LTS associated with R is included in the LTS associated with R'. In particular, if the LTS associated with R' has any of the bounded-nondeterminism properties considered in this paper, then the LTS associated with R has the property too. We follow [16] and ignore the negative premises in the TSSs. (Note that ignoring negative premises makes the rule formats proposed in the paper more restrictive, as negative premises may render some rules junk. However, as in [16], none of the rule formats that we introduce here impose any restrictions on negative premises and the results we provide extend to a setting with negative premises at the price of working with three-valued stable models.) From now on we use 'TSS with terms as labels' as a synonym of 'TSS with terms as labels and without negative premises'.

Notation 9. Many of the examples in the paper consider the following signature Σ_0 of reference. Let L consist of countably many ground labels l_i with $i \in \mathbb{N}$ (i.e., the l_i are constants). The signature Σ_0 consists of unary function symbols f and g, and the constant symbols in L.

3. Dyadic transformations

As mentioned in the introduction, our goal in this paper is to present a unified rule format guaranteeing a variety of bounded nondeterminism properties for TSSs with terms as labels. In order to do so, we introduce a collection of dyadic transformations over TSSs that play a key role in our approach.

The dyadic transformations stem from what we call the branching flow in transitions (the notion of branching flow will become clear below and in Definition 15 in Section 4). Differently from the usual flow that stresses the states in a sequence of transitions, the branching flow emphasizes the one-to-many relation between the components of formulae in the set of transitions of an LTS.

We introduce a family of transformations that turn the *triadic* formula representation $t \xrightarrow{l} t'$ into a *dyadic* representation $s \mathcal{R} r$ where s is the *source*, r is the *target* and $\mathcal{R}$ is a binary relation symbol in the set $\{\longrightarrow, \longleftarrow, \uparrow, \downarrow\}$. The relation symbol $\mathcal{R}$ does not carry any label, and the arbitrary term l labelling the triadic formula $t \xrightarrow{l} t'$ is part either of the source s or the target r in the dyadic formula $s \mathcal{R} r$. Definition 10 below presents six dyadic transformations. Our dyadic representation is reminiscent of the *commitments* of [26], the *residual data types* in Definition 13.9 of [7] and the *residuals* in Definition 1 of [29]. In those three works, a transition consisting of an agent, an action and a derivative agent is represented as an agent and a commitment (resp. residual), which contains both the action and the derivative agent.

Definition 10 (Dyadic transformation)**.** Let $t \xrightarrow{l} t'$ be a triadic formula of a TSS with terms as labels. The dyadic transformations $\mathcal{D}_k$ (with $1 \leq k \leq 6$) are given by:

- $\mathcal{D}_1(t \xrightarrow{l} t') = t \longrightarrow (l, t')$.
- $\mathcal{D}_2(t \xrightarrow{l} t') = t' \longleftarrow (l, t)$.
- $\mathcal{D}_3(t \xrightarrow{l} t') = l \uparrow (t, t')$.
- $\mathcal{D}_4(t \xrightarrow{l} t') = (t, l) \longrightarrow t'$.
- $\mathcal{D}_5(t \xrightarrow{l} t') = (t', l) \longleftarrow t$.
- $\mathcal{D}_6(t \xrightarrow{l} t') = (t, t') \downarrow l$.

A transformation $\mathcal{D}_k$ (with $1 \leq k \leq 6$) is extended to a triadic TSS with terms as labels in the trivial way. Each rule

$$\frac{\{u_i \xrightarrow{n_i} u_i' \mid i \in I\}}{t \xrightarrow{m} t'} \qquad \text{is transformed into} \qquad \frac{\{\mathcal{D}_k(u_i \xrightarrow{n_i} u_i') \mid i \in I\}}{\mathcal{D}_k(t \xrightarrow{m} t')} .$$

The sets $\mathbb{S}_k(\Sigma)$ and $\mathbb{R}_k(\Sigma)$ contain respectively the sources and the targets over a signature Σ in a $\mathcal{D}_k$-transformed TSS. For closed formulae $o \,\mathcal{R}\, d$ we use the terminology *origin* for the o and *destination* for the d. The sets $\mathbb{O}_k(\Sigma)$ and $\mathbb{D}_k(\Sigma)$ contain respectively the origins (i.e., closed sources) and the destinations (i.e., closed targets) over a signature Σ in a $\mathcal{D}_k$-transformed TSS.

Example 1. Applying the transformation $\mathcal{D}_2$ to the rule

$$\frac{x \xrightarrow{y} x'}{f(x) \xrightarrow{f(y)} f(x')}$$

yields the rule

$$\frac{x' \longleftarrow (y, x)}{f(x') \longleftarrow (f(y), f(x))} .$$

The term $f(x')$ is the source of the transition formula $f(x') \longleftarrow (f(y), f(x))$ and $(f(y), f(x))$ is its target. Therefore, for all closed terms p and q, the closed term $f(p)$ is an origin and $(f(p), f(q))$ is a destination.

The dyadic transformations are duly extended to proof trees and LTSs. Notice that the transformations $\mathcal{D}_k$, with $1 \leq k \leq 6$, are only representational. Both a dyadic formula and its triadic analogue have type $\mathbb{S}_k(\Sigma) \times \mathbb{R}_k(\Sigma) = \mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma)$, i.e., both formulae are structurally identical, and a triadic TSS and its dyadic counterpart prove exactly the same set of transitions, in the sense of the following lemma.

Lemma 1. *Let R be a triadic TSS with terms as labels and let R' be the dyadic TSS obtained by applying to R some dyadic transformation $\mathcal{D}_k$ with $1 \leq k \leq 6$, i.e., $R' = \mathcal{D}_k(R)$. A transition $p \xrightarrow{l} p'$ is provable in R iff $\mathcal{D}_k(p \xrightarrow{l} p')$ is provable in R'.*

As we sketched before, the $\mathcal{R}$ in a dyadic TSS is a directed binary relation symbol that reflects the branching flow. By way of example, consider the dyadic transformation $\mathcal{D}_1$. An origin p that branches into a finite number of destinations (l, p') corresponds to a finitely branching process p in the (triadic) LTS. An origin p that branches into a possibly infinite number of destinations (l, p') where there are only finitely many distinct l corresponds to initials finiteness. On the other hand, in the setting of transformation $\mathcal{D}_4$, an origin (p, l) that branches into a finite number of destinations p' corresponds to image finiteness. (We defer a discussion of the correspondence of the other dyadic transformations to other bounded-nondeterminism properties of interest to Section 4.)

The states in a sequence of transitions can be retrieved from the relation symbol $\mathcal{R}$, by noticing the direction of the arrow that $\mathcal{R}$ depicts. (For $\uparrow$ and $\downarrow$, turn the arrow clockwise and anti-clockwise respectively over the (t, t').) For example, the $\mathcal{D}_6$-transformed transitions $(p_1, p_2) \downarrow l_1$ and $(p_2, p_3) \downarrow l_2$, correspond to the sequence of triadic transitions $p_1 \xrightarrow{l_1} p_2 \xrightarrow{l_2} p_3$.

For the first three dyadic transformations, i.e., $\mathcal{D}_k$ where $1 \leq k \leq 3$, the targets range over the Cartesian product $\mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma)$. The following notation will be adopted in what follows.

Definition 11. Let $t \xrightarrow{l} t'$ be a formula of a triadic TSS and let $o\,\mathcal{R}\,(d_1, d_2)$ be the dyadic formula obtained by applying to $t \xrightarrow{l} t'$ one of the first three dyadic transformations $\mathcal{D}_k$, i.e., $o\,\mathcal{R}\,(d_1, d_2) = \mathcal{D}_k(t \xrightarrow{l} t')$ with $1 \leq k \leq 3$. We write $Tr_k(o_1, d_1, d_2)$ for the triadic transition obtained by assigning the roles of source, label and target to each of the o, d_1 and d_2 according to $\mathcal{D}_k$, in such a way that Tr_k is the inverse of $\mathcal{D}_k$. More precisely, $Tr_1(o, d_1, d_2) = o \xrightarrow{d_1} d_2$, $Tr_2(o, d_1, d_2) = d_2 \xrightarrow{d_1} o$, and $Tr_3(o, d_1, d_2) = d_1 \xrightarrow{o} d_2$.

We consider the dyadic projections that are defined over the first three dyadic transformations by projecting either on the first or on the second component in the targets of transitions.

Definition 12 (Dyadic projection)**.** Let $t \xrightarrow{l} t'$ be a formula of a triadic TSS. The dyadic projections $\mathcal{D}_k^{prj}$ (with $1 \leq k \leq 3$ and prj either one of the projection functions π_1 and π_2) are given by:

- $\mathcal{D}_1^{\pi_1}(t \xrightarrow{l} t') = t \longrightarrow l$.
- $\mathcal{D}_2^{\pi_1}(t \xrightarrow{l} t') = t' \longleftarrow l$.
- $\mathcal{D}_3^{\pi_1}(t \xrightarrow{l} t') = l \uparrow t$.
- $\mathcal{D}_3^{\pi_2}(t \xrightarrow{l} t') = l \uparrow t'$.
- $\mathcal{D}_2^{\pi_2}(t \xrightarrow{l} t') = t' \longleftarrow t$.
- $\mathcal{D}_1^{\pi_2}(t \xrightarrow{l} t') = t \longrightarrow t'$.

A dyadic projection $\mathcal{D}_k^{prj}$ (with $1 \leq k \leq 3$ and $prj \in \{\pi_1, \pi_2\}$) is extended to a triadic TSS with terms as labels in the trivial way. Each rule

$$\frac{\{u_i \xrightarrow{n_i} u_i' \mid i \in I\}}{t \xrightarrow{m} t'} \qquad \text{is transformed into} \qquad \frac{\{\mathcal{D}_k^{prj}(u_i \xrightarrow{n_i} u_i') \mid i \in I\}}{\mathcal{D}_k^{prj}(t \xrightarrow{m} t')}\ .$$

Notice that the dyadic projections $\mathcal{D}_k^{prj}$ are not only representational transformations. Now, a projected formula has type $\mathbb{S}_k(\Sigma) \times prj(\mathbb{R}_k(\Sigma)) = \mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma)$ (with $1 \leq k \leq 3$ and $prj \in \{\pi_1, \pi_2\}$), and its triadic analogue has type $\mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma) \times \mathbb{T}(\Sigma)$, which means that the two formulae are not structurally identical. The dyadic projection entails a many-to-one relation among the transitions of a triadic LTS and the transitions of its dyadic-projected counterpart. That is, if $o\,\mathcal{R}\,d$ is a dyadic-projected transition, then there exists a closed term p such that o, d and p can be assigned the roles of source, label and targets as to constitute a transition in the original triadic LTS.

The rules of a triadic TSS with terms as labels are in a similar many-to-one relation with the rules of the $\mathcal{D}_k^{prj}$-projected TSS. For illustration, for each rule $\rho = \{v_i \longrightarrow w_i \mid i \in I\}/s \longrightarrow r$ in the $\mathcal{D}_1^{\pi_1}$-projected TSS there exist terms t and u_i with $i \in I$ such that $\{v_i \xrightarrow{w_i} u_i \mid i \in I\}/s \xrightarrow{r} t$ is a rule in the original triadic TSS.

Lemma 2. *Let R be a triadic TSS with terms as labels and let R' be the dyadic TSS obtained by applying to R some dyadic projection $\mathcal{D}_k^{prj}$ (with $1 \leq k \leq 3$ and $prj \in \{\pi_1, \pi_2\}$), i.e., $R' = \mathcal{D}_k^{prj}(R)$. Then, for each triadic transition $p \xrightarrow{l} p'$ that is provable in R, the transition $\mathcal{D}_k^{prj}(p \xrightarrow{l} p')$ is provable in R'.*

Remark 1. Let R be a triadic TSS and let R' be the dyadic TSS obtained by applying to R some dyadic projection $\mathcal{D}_k^{prj}$ (with $1 \leq k \leq 3$ and $prj \in \{\pi_1, \pi_2\}$). Since dyadic projections abstract from information that is present in the original triadic TSS, there might be some transition $o\,\mathcal{R}\,d$ that is provable in R', but for which there exists no closed term q such that either $Tr_k(o, d, q)$ (if $prj = \pi_1$) or $Tr_k(o, q, d)$ (if $prj = \pi_2$) is provable in R. By way of example, consider the triadic TSS R_1 with rules

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

The dyadic TSS R'_1 obtained from R_1 using $\mathcal{D}^{\pi_1}_1$ is

$$\frac{}{c \longrightarrow a} \qquad\qquad \frac{x \longrightarrow a}{f(x) \longrightarrow a}\ .$$

The transition $f(c) \longrightarrow a$ is provable using the rules in R'_1. On the other hand, there is no transition from $f(c)$ that is provable in R_1.

For what concerns the dyadic projection $\mathcal{D}^{\pi_2}_1$, consider the triadic TSS R_2 with rules

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

The dyadic TSS R'_2 obtained from R_2 using $\mathcal{D}^{\pi_2}_1$ is

$$\frac{}{c \longrightarrow c} \qquad\qquad \frac{x \longrightarrow y}{f(x) \longrightarrow f(y)}\ .$$

The transition $f(c) \longrightarrow f(c)$ is provable using the rules in R'_2. On the other hand, there is no transition from $f(c)$ that is provable in R_2.

Similar examples can be constructed for each of the other four dyadic projections.

Notation 13. In order to unify the notation, in what follows, we let $prj \in \{\pi_1, \pi_2, id\}$, where id is the identity function, and say 'dyadic transformation $\mathcal{D}^{prj}_k$' when referring both to the dyadic transformations $\mathcal{D}^{prj}_k$ with $1 \leq k \leq 6$ and $prj = id$, and to the dyadic projections $\mathcal{D}^{prj}_k$ with $1 \leq k \leq 3$ and $prj \in \{\pi_1, \pi_2\}$. We write $\mathbb{S}^{prj}_k(\Sigma)$, $\mathbb{R}^{prj}_k(\Sigma)$, $\mathbb{O}^{prj}_k(\Sigma)$ and $\mathbb{D}^{prj}_k(\Sigma)$ for the sets of sources, targets, origins, and destinations respectively in a $\mathcal{D}^{prj}_k$-dyadic LTS.

From now on, we replace the binary relation symbol '$\mathcal{R}$' with the more suggestive '$\longrightarrow$' and write '$s \longrightarrow r$' instead of '$s \,\mathcal{R}\, r$' for any $\mathcal{D}^{prj}_k$-dyadic formula

Definition 14. Let R be a triadic TSS with terms as labels and let R' be the dyadic TSS obtained by applying to R some dyadic transformation $\mathcal{D}^{prj}_k$, i.e., $R' = \mathcal{D}^{prj}_k(R)$.

(i) We say R' is $\mathcal{D}^{prj}_k$-dyadic.

(ii) We say R is $\mathcal{D}^{prj}_k$-finite iff R' is finitely branching, that is, iff for every origin o in R' the set $\{d \mid o \longrightarrow d\}$ is finite.

(When the transformation $\mathcal{D}^{prj}_k$ is clear from the context, or when we refer to any of the dyadic transformations indistinctly, we will omit the tag '$\mathcal{D}^{prj}_k$-' and simply say 'dyadic'.)

For example, let R be a triadic TSS with terms as labels. R is finitely branching iff R is $\mathcal{D}^{id}_1$-finite. R is image finite iff R is $\mathcal{D}^{id}_4$-finite. R is initials finite iff R is $\mathcal{D}^{\pi_1}_1$-finite.

The dyadic transformations allow one to deal with SOS with higher-order processes by using the framework for plain SOS. Notice that one can start with a triadic TSS with terms as labels, apply a dyadic transformation, and then trivially inject the obtained dyadic TSS into a triadic TSS with ground labels by assuming that all transitions are labelled with the same constant term.

Next we explore the space of bounded nondeterminism properties that results from Definition 14.

4. Space of bounded-nondeterminism properties

Definition 4 introduces a variety of bounded-nondeterminism properties. Most of these properties do not hold for reasonably expressive LTSs/process calculi. However, they have some theoretical interest in the classification of the bounded-nondeterminism properties.

Definition 15. We define the following bounded-nondeterminism properties of a triadic LTS:

(i) Finite branching: $\forall p.\ \{(l, p') \mid p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma) \times \mathbb{C}(\Sigma))$.

(ii) Finite folding: $\forall p'.\ \{(p, l) \mid p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma) \times \mathbb{C}(\Sigma))$.

(iii) Finite bundling: $\forall l.\ \{(p, p') \mid p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma) \times \mathbb{C}(\Sigma))$.

(iv) Image finiteness: $\forall p.\forall l.\ \{p' \mid p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(v) Source finiteness: $\forall l.\forall p'.\ \{p \mid p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(vi) Label finiteness: $\forall p.\forall p'.\ \{l \mid p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(vii) Initials finiteness: $\forall p.\ \{l \mid \exists p'.\ p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(viii) Finals finiteness: $\forall p'.\ \{l \mid \exists p.\ p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(ix) Heads finiteness: $\forall l.\ \{p \mid \exists p'.\ p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(x) Tails finiteness: $\forall l.\ \{p' \mid \exists p.\ p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(xi) Antecedents finiteness: $\forall p'.\ \{p \mid \exists l.\ p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

(xii) Consequents finiteness: $\forall p.\ \{p' \mid \exists l.\ p \xrightarrow{l} p'\} \in \mathcal{P}_\omega(\mathbb{C}(\Sigma))$.

The properties listed above involve all the possible combinations of the three components in a triadic formula (source, target and label) such that either one component branches finitely into the other two, two components branch finitely into the other one, or one component branches finitely into a second component and possibly infinitely into the third component. Figure 1 depicts these combinations schematically. Finite branching (i), image finiteness (iv), and initials finiteness (vii) are well-known properties. We have contrived the names of the rest of the properties, which, to the best of our knowledge, have not been considered in the literature before.

Properties (i) to (xii) coincide with finite branching of the dyadic LTSs obtained by applying each of the six dyadic transformations in Definition 10 and the six dyadic projections in Definition 12, respectively. In the nomenclature of Definition 14:

(i) Finite branching $\stackrel{\text{def}}{=} \mathcal{D}_1^{id}$-finite.	(vii) Initials finite $\stackrel{\text{def}}{=} \mathcal{D}_1^{\pi_1}$-finite.
(ii) Finite folding $\stackrel{\text{def}}{=} \mathcal{D}_2^{id}$-finite.	$(viii)$ Finals finite $\stackrel{\text{def}}{=} \mathcal{D}_2^{\pi_1}$-finite.
(iii) Finite bundling $\stackrel{\text{def}}{=} \mathcal{D}_3^{id}$-finite.	(ix) Heads finite $\stackrel{\text{def}}{=} \mathcal{D}_3^{\pi_1}$-finite.
(iv) Image finite $\stackrel{\text{def}}{=} \mathcal{D}_4^{id}$-finite.	(x) Tails finite $\stackrel{\text{def}}{=} \mathcal{D}_3^{\pi_2}$-finite.
(v) Source finite $\stackrel{\text{def}}{=} \mathcal{D}_5^{id}$-finite.	(xi) Antecedents finite $\stackrel{\text{def}}{=} \mathcal{D}_2^{\pi_2}$-finite.
(vi) Label finite $\stackrel{\text{def}}{=} \mathcal{D}_6^{id}$-finite.	(xii) Consequents finite $\stackrel{\text{def}}{=} \mathcal{D}_1^{\pi_2}$-finite.

Properties $(i) - (xii)$ can be arranged in the Hasse diagram of Figure 2, which depicts the information order of the properties, that is, the order relation is reverse implication. We write $q \leq p$ if q occurs below p in the Hasse diagram. Properties (iv), (vi) and (v) at the bottom are elementary (i.e., most general) and properties (i), (iii) and (ii) at the top are derived (i.e., most specific). Properties (vii) to (xii) are complementary (i.e., the difference between a derived and an elementary, more on this below).

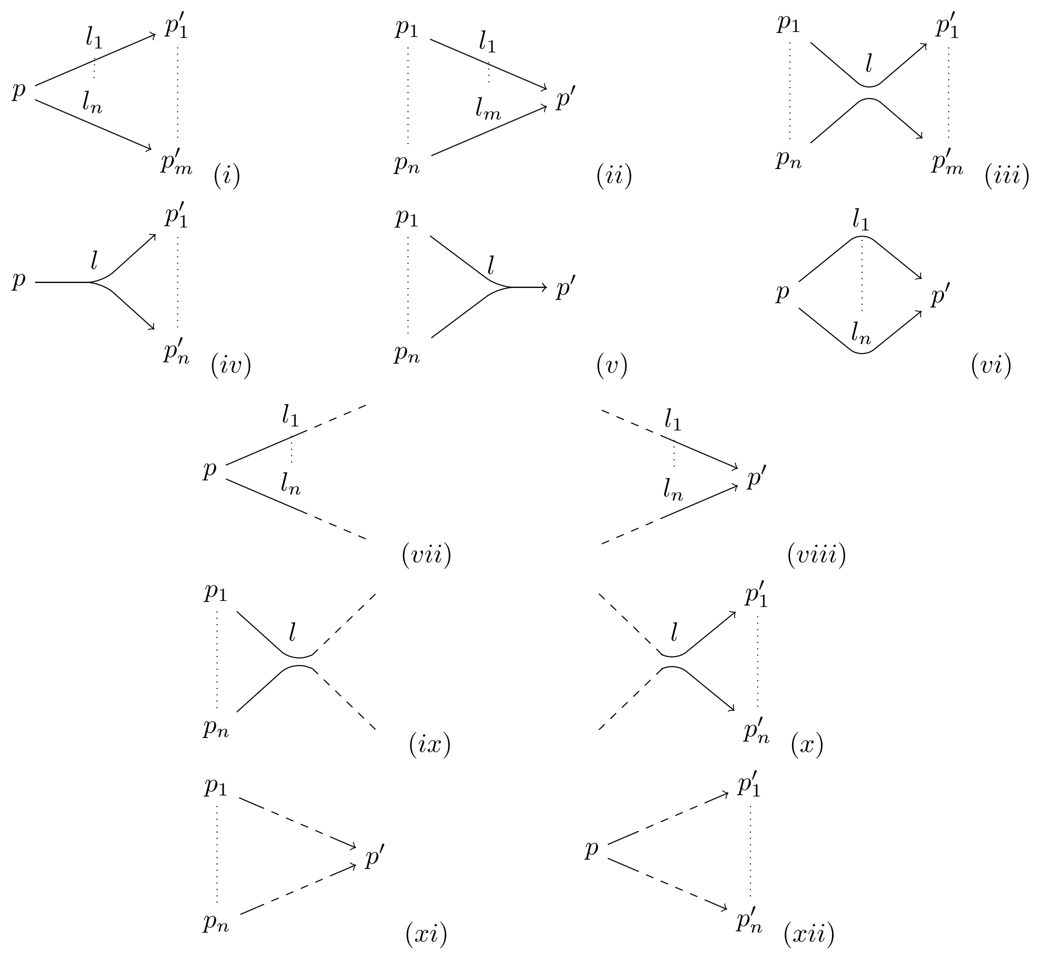

Figure 1: Schematic representation of the properties in Definition 15. The dots indicate that the components branch finitely (i.e., the indices n and the m are natural numbers), and the dashes indicate that the missing components can branch infinitely.

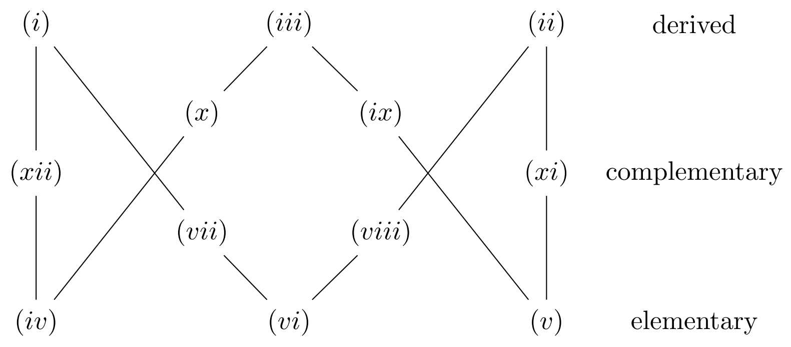

Figure 2: Hasse diagram of the properties in Definition 15.

The properties are displayed in three layers: the bottom layer contains three elementary properties; the top layer contains three derived properties; the middle layer contains six complementary properties, each of them in the middle of a path from an elementary to a derived property.

Proposition 1. *For every two bounded-nondeterminism properties p and q of Definition 15, p implies q iff $q \leq p$ in the Hasse diagram of Figure 2.*

Proof. Recall notation Tr_k from Definition 11. For the ($\Longleftarrow$) direction, a derived property d that corresponds to a dyadic transformation $\mathcal{D}_k^{id}$ (with $1 \leq k \leq 3$) states that a component t_1 branches finitely into the other components t_2 and t_3, i.e., for each t_1, the set $\{(t_2, t_3) \mid Tr_k(t_1, t_2, t_3)\}$ is finite. Trivially, this implies the two complementary properties c_1 and c_2 that state that the t_1 is respectively allowed to branch infinitely into either the t_2 or the t_3, i.e., for each t_1 both the sets $\{t_2 \mid \exists t_3. Tr_k(t_1, t_2, t_3)\}$ and $\{t_3 \mid \exists t_2. Tr_k(t_1, t_2, t_3)\}$ are finite. In turn, the complementary properties c_1 and c_2 respectively imply the two elementary properties e_1 and e_2 that state that, given a t_1, the t_2 and t_3 branch finitely into each other, i.e., for each t_1 and t_2 the set $\{t_3 \mid Tr_k(t_1, t_2, t_3)\}$ is finite, and for each t_1 and t_3 the set $\{t_2 \mid Tr_k(t_1, t_2, t_3)\}$ is finite.

For the ($\Longrightarrow$) direction, no implications other than the ones given by the Hasse diagram of Figure 2 exist. This can be shown straightforwardly by picking counter-examples for each case, which we omit here. $\square$

For instance, properties (xii) and (vii) are below property (i) in the diagram, and, in turn, properties (iv) and (vi) are below properties (xii) and (vii) respectively. Therefore, (i) implies each of (xii), (vii), (iv) and (vi).

Some equivalences between combinations of the properties depicted in the Hasse diagram of Figure 2 can be established.

Proposition 2. *A bounded-nondeterminism derived property d is equivalent to the conjunction of the elementary property e from one of the chains that has d as maximum in the Hasse diagram of Figure 2, and the complementary property c from the adjacent chain that also has d as maximum, i.e., $d \Longleftrightarrow e \wedge c$.*

Proof. By Definition 15 the derived property d corresponds to a dyadic transformation $\mathcal{D}_k^{id}$ with $1 \leq k \leq 3$. We only detail the proof when the complementary property c corresponds to $\mathcal{D}_k^{\pi_1}$, as the case that corresponds to $\mathcal{D}_k^{\pi_2}$ is similar. The ($\Longrightarrow$) direction is trivial by Proposition 1. For the ($\Longleftarrow$) direction, notice that the elementary property e states that two components t_1 and t_2 branch finitely into the other component t_3, i.e., for each t_1 and t_2 the set $\{t_3 \mid Tr_k(t_1, t_2, t_3)\}$ is finite. The complementary property c from the other chain states that component t_1 branches finitely into component t_2, but possibly branches infinitely into component t_3. By applying property e to each of the finitely many pairs (t_1, t_2), component t_1 is shown to branch finitely into component t_3, and thus the derived property d holds, i.e., for each t_1 the set $\{(t_2, t_3) \mid Tr_k(t_1, t_2, t_3)\}$ is finite. $\square$

For instance, properties (iv) and (vii) together are equivalent to (i), and so are properties (vi) and (xii) together.

Proposition 3. *A bounded-nondeterminism derived property d is equivalent to the conjunction of the two complementary properties c_1 and c_2 from the two adjacent chains that have d as maximum.*

Proof. The ($\Longrightarrow$) direction is trivial by Proposition 1. For the ($\Longleftarrow$) direction, by Proposition 1 the complementary property c_i (with $i \in \{1,2\}$) implies the elementary property e_i such that $e_i \leq c_i$, and by Proposition 2, $e_i \wedge c_j$ with $i, j \in \{1,2\}$ and $i \neq j$ implies d. □

A complementary property in a chain that has a derived property d as maximum is the difference between d and the elementary property from the adjacent chain that also has d as maximum. Notice that in order to imply a derived property, at least one complementary property from either of the adjacent chains that have the derived one as maximum must take part in the conjunction. The following proposition illustrates this.

Proposition 4. *A bounded-nondeterminism derived property d is not implied by the conjunction of the two elementary properties e_1 and e_2 that are below d in the Hasse diagram of Figure 2.*

Proof. By Definition 15 the derived property d corresponds to a dyadic transformation $\mathcal{D}^{id}_k$ with $1 \leq k \leq 3$. The derived property d requires that one component t_1 branches finitely into the components t_2 and t_3, i.e., for every t_1, the set $\{(t_2, t_3) \mid Tr_k(t_1, t_2, t_3)\}$ is finite. The elementary properties e_1 and e_2 that are below d respectively state that, given a t_1, the t_2 and t_3 branch finitely into each other, i.e., for every t_1 and t_2 the set $\{t_3 \mid Tr_k(t_1, t_2, t_3)\}$ is finite, and for every t_1 and t_3 the set $\{t_2 \mid Tr_k(t_1, t_2, t_3)\}$ is finite. But for each t_1, the t_2 and the t_3 could be infinitely many and still branch finitely into each other.

This idea can be exploited to construct counter-examples to the implication $e_1 \wedge e_2 \implies d$. By way of example, consider the triadic LTS L given by $\{p_0 \xrightarrow{l_i} p_i \mid i \in \mathbb{N}\}$. L is image finite (property (iv) in Figure 2) because for each p_i and l_j there is at most one p_ℓ such that $p_i \xrightarrow{l_j} p_\ell$ (i.e., $i = 0$, $j = \ell$), and L is labels finite (property (vi) in Figure 2) because for each p_i and p_ℓ there is at most one l_j such that $p_i \xrightarrow{l_j} p_\ell$ (i.e., $i = 0$, $j = \ell$). However, L is not finitely branching because for p_0 there are infinitely many pairs (l_i, p_i) such that $p_0 \xrightarrow{l_i} p_i$. □

For further illustration, the equivalence class of property (i) up to logical equivalence is

$$\{(i), (xii) \wedge (vi), (vii) \wedge (iv), (xii) \wedge (vii)\},$$

and property (i) is implied by any of the properties in

$$\{(xii) \wedge (viii), (xii) \wedge (ii), (vii) \wedge (x), (vii) \wedge (iii)\}.$$

In the light of the dyadic transformations, the rule format in [16] for finite branching is enough to guarantee $\mathcal{D}^{prj}_k$-finiteness of the LTS associated to a triadic TSS with terms as labels. That is, by applying each of the dyadic transformations as a pre-processing step, that rule format ensures all the derived, elementary and complementary bounded-nondeterminism properties in this section.

Next we show how to improve on the rule format of [16] to cover more TSSs than the ones covered there. Sections 5 and 6 explain how the set of finitely branching TSSs that meet the conditions of the rule format can be enlarged by considering junk rules (i.e., rules that cannot give rise to transitions), and by refining the η-types of [16] such that some junk rules do not have η-type. Section 7 introduces the S-types, which adapt these refinements on the η-types to dyadic TSSs. Finally, Section 8 presents rule formats that are based on the S-types and guarantee $\mathcal{D}^{prj}_k$-finiteness of the LTS associated with a triadic TSS with terms as labels.

5. Junk rules and the η-types

A rule format is a set of conditions on the rules of a TSS that ensure that the associated LTS enjoys some property of interest (e.g., being finitely branching). A rule format imposes *sufficient conditions* on the TSS for the associated LTS to enjoy the property. In devising a rule format there is a trade-off between generality and ease of use: the syntactic conditions typically do not cover all the TSSs whose associated

LTSs have the property of interest, but only a reasonably large class of those that do so. For instance, a TSS might contain rules that do not give rise to transitions (the so-called junk rules) and do not fit the rule format. In general, the conditions of a rule format could be weakened in order to enlarge the class of TSSs that are covered by it and enjoy the property of interest. However, this usually leads to more complex conditions.

One of our goals in this paper is to improve the rule format for bounded nondeterminism in [16] so that it can filter out more junk rules. To do so, in this section we will explain the notion of η-type from [16] and how it can be used to filter out some junk rules. Next, in Section 6, we will refine the notion of η-type from [16] in order to increase the number of TSSs that our rule format (presented in Section 8) covers.

Note to the reader. The following considerations about junk rules are orthogonal to the dyadic transformations that we have presented before. Therefore, for the sake of clarity in the presentation, in the remainder of the section and in Section 6 we consider triadic TSSs with ground labels.

Familiarity with the rule format for finite branching from [16] will help in appreciating this section. However, we have striven to make the presentation understandable even with no prior knowledge of that paper.

Intuitively, the rule format for bounded nondeterminism in [16] places restrictions on the allowed rules ensuring that, for each closed term p,

1. the rules in the TSS do not allow one to simulate 'unguarded recursion' for p,
2. only finitely many rules can be employed to derive transitions from p, and
3. each rule can only be used to infer finitely many transitions from p.

The third property is checkable for each rule in isolation and is embodied in the requirement that the TSS be in bounded nondeterminism format (see Definition 23 to follow). This format essentially ensures that the transitions of p that can be derived from a rule depend only on the transitions of the closed terms that are substituted for the variables that occur in the source of the rule. For example, rules of the form

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

are *not* in bounded nondeterminism format because the instantiation of the variable y can be chosen arbitrarily and thus its behaviour need not have any connection with that of the closed term that is substituted for the variable x. It is easy to see how rules of that form can be used to derive infinitely many transitions from a term of the form $f(p)$.

On the other hand, the first and the second properties are 'global' and need to be checked for sets of rules. The existence of a strict stratification (see Definition 17) enforces the first property, while the second is guaranteed by the requirement that the TSS be bounded (see Definition 22 for our version of that notion). In order to define the notion of bounded TSS, Fokkink and Vu classify the transition rules in a TSS according to their so-called η-types.

The intuition behind the use of the η-types, presented in Definition 11 of [16], is that they enforce that the number of rules that give rise to transitions from a given process is finite. The η-types are parametric on a map $\eta : \mathbb{T}(\Sigma) \to \mathcal{P}_\omega(\mathbb{T}(\Sigma))$ that specifies a predefined, finite set of terms $\eta(t)$ for each source t. For the purpose of this discussion, our readers can think of $\eta(t)$ as being the set of the sources of premises in rules having t as the source of the conclusion; intuitively, those are the terms on which initial transitions from instantiations of t may depend. By way of example, let A be a countably infinite set of actions, and consider the (infix) binary operator $+$ from CCS in [25] with rules

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

The initial behaviour of an instantiation of $x+y$ depends only on that of the terms (substituted for) x and y. Thus, based on what we said earlier, it is natural to define $\eta(x+y) = \{x, y\}$. We say that η is the *support map* and call $\eta(t)$ the *support for source* t.

Since A is infinite, the set of rules having $x+y$ as source is also infinite. However, when the closed terms p and q are finitely branching, only finitely many rules can be used to derive transitions from $p+q$. The technical notion of η-type is meant to capture this fact. Two rules with the same source t have different η-type if, and only if, for some term u in the support for t, the rules differ in the respective sets of labels that appear in their premises with source u. An η-type is written $\langle t, \psi\rangle$, where t is the source of the rule and $\psi : \eta(t) \to \mathcal{P}_\omega(L)$ is the map that delivers for each term u in the support for t, a finite set of labels. Intuitively, $\psi(u)$ is the set of labels in the premises with source u.

Continuing with our example, the η-type of rule L_a is $\langle x+y, \{x \mapsto \{a\}, y \mapsto \emptyset\}\rangle$, and the η-type of rule R_a is $\langle x+y, \{x \mapsto \emptyset, y \mapsto \{a\}\}\rangle$. Note that different rules have different η-types. The η-types of the rules in a TSS that meet the conditions of the rule format of [16] are required to be finitely inhabited. That is, for each η-type $\langle t, \psi\rangle$ there are only finitely many rules having $\langle t, \psi\rangle$ as their η-type. This constraint is met by the η-types of the rules for the $+$ operation.

The rule format from [16] requires that the images in the codomain of both maps η and ψ be finite. The requirement that, for each term t, the support $\eta(t)$ is finite is a necessary condition for the associated LTS to be finitely branching. The following example illustrates why.

Example 2. Consider the following TSS over the signature Σ_0 of Notation 9, where $L = \{l_i \mid i \in \mathbb{N}\}$:

$$\frac{}{l_i \xrightarrow{l_i} l_i}\,, \qquad i \in \mathbb{N} \qquad\qquad \frac{l_i \xrightarrow{l_i} l_i}{f(x) \xrightarrow{l_i} l_i}\,, \qquad i \in \mathbb{N}.$$

Notice that a source $f(p)$ (with $p \in \mathbb{C}(\Sigma)$) affords infinitely many transitions $f(p) \xrightarrow{l_i} l_i$ (with $i \in \mathbb{N}$) and the associated LTS is infinitely branching. However, if we allowed the sets in the codomain of η to be infinite, then we could have $\eta(f(x)) = L$ and the ith instantiation of the rule template on the right would have η-type $\langle f(x), \psi_i\rangle$, where ψ_i is such that $\psi_i(l_i) = \{l_i\}$, and $\psi_i(l_j) = \emptyset$ if $j \neq i$. The η-types would be finitely inhabited and the rule format would not guarantee finite branching of the associated LTS.

The requirement that $\eta(f(x))$ be finite enforces that the infinitely many instantiations of the rule template on the right, which all share the same source $f(x)$, must be distinguished by focusing only on finitely many sources of their premises. Since the set of possible sources of their premises is infinite, i.e., $\{l_i \mid i \in \mathbb{N}\}$, it is actually impossible to distinguish all of them, and the instantiations whose premises do not have their source in the finite support set $\eta(f(x))$ have one and the same η-type $\langle f(x), \{t \mapsto \emptyset \mid t \in \eta(f(x))\}\rangle$, which is infinitely inhabited. Suitably, the TSS does not meet the conditions of the rule format from [16].

On the contrary, the requirement that, for each rule with source t and for each term $u \in \eta(t)$, the set of labels in the premises with source u is finite is not a necessary condition for the associated LTS to be finitely branching. However, the rule format of [16] would report more false negatives if this requirement were omitted. That is, more TSSs would be considered as not satisfying the conditions for finite branching although their associated LTSs are finitely branching. The reason is the presence of a certain class of junk rules. Again, let us illustrate this point with an example.

Example 3. Consider the following TSS over the signature Σ_0 of Notation 9, where $L = \{l_i \mid i \in \mathbb{N}\}$:

$$\frac{}{l_1 \xrightarrow{l_1} l_1} \qquad\qquad \frac{\{l_1 \xrightarrow{l_j} l_j \mid j \in \mathbb{N}\}}{f(l_1) \xrightarrow{l_i} l_i}\,, \qquad i \in \mathbb{N}.$$

The associated LTS is finitely branching because the only provable transition is $l_1 \xrightarrow{l_1} l_1$. The instantiations of the rule template on the right have infinitely many premises with source l_1 and label and target l_j where $j \in \mathbb{N}$. However, none of the instantiations can take part in a proof tree because the infinitely many distinct premises share source l_1 and the transition $l_1 \xrightarrow{l_1} l_1$ cannot prove all of them (i.e., all the instantiations are junk rules). Assume $\eta(f(l_1)) = \{l_1\}$. Happily, none of the instantiations of the rule template on the right has a valid η-type, because no ψ exists such that the finite set $\psi(l_1)$ contains every label in the premises with source l_1 (i.e., $\psi(l_1) \neq \{l_j \mid j \in \mathbb{N}\}$). Therefore the η-types are finitely inhabited and the TSS meets the conditions of the rule format from [16]. On the other hand, if $\psi(l_1)$ were allowed to be equal to $\{l_j \mid j \in \mathbb{N}\}$, then all the instantiations of the rule template on the right would have the same η-type $\langle f(x), \{l_1 \mapsto L\}\rangle$, which would be infinitely inhabited.

6. Refining the η-types

Although the rule format of [16] covers TSSs such as the one in Example 3, it falls short of covering TSSs with other kinds of junk rules. In the following subsections we analyze some of these cases and provide successive refinements to the η-types of [16] that we will incorporate in the rule format we present in Section 8.

6.1. Restricted support

A rule of a TSS whose premises do not unify with conclusions of other rules in the TSS is certainly a junk rule, since its premises will never be true. But this fact is not always ascertained by not having a valid η-type, since the η-types rely only on local information. The following example illustrates this observation.

Example 4. Consider the following TSS over the signature Σ_0 of Notation 9:

$$\frac{}{g(l_1) \xrightarrow{l_1} l_1} \qquad\qquad \frac{g^i(x) \xrightarrow{l_i} x}{f(x) \xrightarrow{l_i} x}\,, \qquad i \in \mathbb{N}.$$

Here g^i stands for applying the unary function symbol g to its argument i times. Notice that the only provable transitions are $g(l_1) \xrightarrow{l_1} l_1$ and $f(l_1) \xrightarrow{l_1} l_1$, and therefore the TSS induces an LTS that is finitely branching. Assume that the TSS has a support map such that

$$\begin{array}{rcll} \eta(g(l_1)) & = & \emptyset, & \\ \eta(f(x)) & = & \{g(x)\} & \text{and} \\ \eta(t) & = & \emptyset & \text{otherwise.} \end{array}$$

Clearly, all the instantiations of the rule template on the right with $i \neq 1$ are junk rules because their premises do not unify with the conclusion of any rule in the TSS. However, all these instantiations have $\langle f(x), \emptyset\rangle$ as their η-type, which is infinitely inhabited. (Notice that no η exists such that the η-types would be finitely inhabited.) Thus the TSS does not meet the conditions of the rule format of [16], even though the associated LTS is finitely branching.

Observation 1. *The η-types have to be refined so that the support map η be restricted to deliver only sets of terms that unify with sources of rules in the TSS. According to this, a rule with source t should only have a valid η-type if the set of sources of its premises is a subset of the restricted support $\eta(t)$.*

Take the support map η of Example 4 above. That function η meets the condition of the restricted support because every term in a set delivered by η (i.e., $g(x) \in \eta(f(x))$) unifies with some rule in the TSS (i.e., $\sigma(g(x)) = g(l_1)$, with $\sigma = \{x \mapsto l_1\}$). The instantiations of the rule template on the right with $i \neq 1$ should not have a valid η-type because $g^i(x) \notin \eta(f(x))$ when $i \neq 1$.

6.2. Targets of premises are important too

Two rules with the same source t have different η-types if there is some term in $\eta(t)$ for which the two rules have premises with different actions. For instance, rules L_a and R_a on page 13 have different η-types because the only premise in L_a is $y \xrightarrow{a} y'$, whereas the only premise in R_a is $x \xrightarrow{a} x'$.

Sometimes it is convenient to distinguish rules with the same source not only by focusing on the actions of their premises but also by looking at the targets of their premises. This is shown in the following example.

Example 5. Consider the following TSS over the signature Σ_0 of Notation 9:

$$1\ \frac{}{g(l_1)\xrightarrow{l_1} l_1} \qquad\qquad 2i\ \frac{g(x)\xrightarrow{l_1} l_i}{f(x)\xrightarrow{l_1} l_i}\ , \qquad i\in\mathbb{N}.$$

The TSS generates an LTS that is finitely branching since the provable transitions are just $g(l_1)\xrightarrow{l_1} l_1$ and $f(l_1)\xrightarrow{l_1} l_1$. Clearly, all the instantiations of the rule template on the right with $i\neq 1$ are junk rules, because the axiom on the left only allows one to prove the transition $g(l_1)\xrightarrow{l_1} l_1$, which can only make true the premise of the instantiation with $i=1$. Pick some η. The η-type of rule L is

$$\langle g(l_1), \{t\mapsto\emptyset \mid t\in\eta(g(l_1))\}\rangle.$$

The η-type of rule $2i$ is $\langle f(x),\psi_i\rangle$ where $\psi_i:\eta(f(x))\to\mathcal{P}_\omega(L)$ is such that

$$\psi_i(t) \quad = \quad \begin{cases}\emptyset & \text{if } t\neq g(x)\\ \{l_1\} & \text{if } t=g(x)\end{cases}$$

Since the premises of rules $2i$ with $i\in\mathbb{N}$ only differ in their targets, all the ψ_i are equal. Therefore, all the rules $2i$ with $i\in\mathbb{N}$ have the same η-type $\langle f(x),\psi_i\rangle$, which is infinitely inhabited, and thus the TSS is not in the rule format of [16]. In words, the rule format does not cover the TSS above, although the associated LTS is finitely branching. Notice that a rule format with the previously mentioned refinement of the restricted support in Observation 1 would not cover the TSS either, because the support map η defined as $\langle f(x),\{g(x)\mapsto\{l_1\}\}\rangle$ meets the conditions of the restricted support (i.e., $\{g(x)\}\subseteq\eta(f(x))$), and every instantiation would have valid (refined) η-type.

Observation 2. *The η-types have to be refined in order to consider the targets of premises. Following this idea, for a rule with η-type $\langle t,\psi\rangle$, the map ψ should deliver, for each source u in the support $\eta(t)$, the set of pairs of labels and targets in the premises of the rule with source u.*

Take the rules $2i$ with $i\in\mathbb{N}$ of Example 5 and consider the map η given there. Each premise has a distinct target l_i. Each rule $2i$ has η-type $\langle f(x),\psi_i\rangle$ where

$$\psi_i(t) \quad = \quad \begin{cases}\emptyset & \text{if } t\neq g(x)\\ \{(l_1,l_i)\} & \text{if } t=g(x)\end{cases}$$

and thus the η-types are finitely inhabited.

6.3. Uniformity in the targets of premises

Recall that the rule format in [16] requires the TSS to be uniform. We paraphrase the notion of uniformity here as the requirement that if t and t' are sources of any two rules in a TSS that differ only in the names of the variables that occur in them, then t and t' are exactly the same term. Uniformity prevents that several rules use distinct names in their sources for variables that morally should have the same name. A non-uniform use of the variables in the sources of rules may result in an infinitely branching LTS even for a TSS whose η-types are all finitely inhabited. Since the η-types of [16] are only sensitive to the source of the rule (notice that the labels are ground there), uniformity needed be enforced only in the sources of rules. A non-uniform use of variables in other positions would not jeopardize the finitely-inhabited discipline of the η-types of [16], and the rule format there needed not care about those uses.

However, the refinement of the η-types in Observation 2 that considers the targets of premises makes the ψ map sensitive to the targets of premises in rules. The following example shows that, additionally to the variables in the source of the rules, the variables in the targets of premises have to be used uniformly.

Example 6. Assume the existence of infinitely many different variable names y_i with $i \in \mathbb{N}$ that are distinct from variable name x. Consider the following TSS over the signature Σ_0 of Notation 9:

$$\frac{}{l_i \xrightarrow{l_1} l_i}\,, \qquad i \in \mathbb{N} \qquad\qquad \frac{x \xrightarrow{l_1} y_i}{f(x) \xrightarrow{l_1} g^i(y_i)}\,, \qquad i \in \mathbb{N}.$$

Assume that the TSS has a support map given by

$$\begin{array}{rcll} \eta(l_i) & = & \emptyset & \\ \eta(f(x)) & = & \{x\} & \\ \eta(t) & = & \emptyset & \quad \text{otherwise} \end{array}$$

The premises of the different instantiations of the rule template on the right only differ in the variable name y_i they use as a target of their premises. Each instantiation has a distinct target $g^i(y_i)$. For each i the instantiation of the rule template would have a different η-type $\langle f(x), \{x \mapsto \{(l_1, y_i)\}\}\rangle$ (considering the refinement in Section 6.2 that takes the targets of premises into account) and the refined η-types would be finitely inhabited. In words, the non-uniform use of variable names y_i yields finitely inhabited η-types and the TSS would meet the conditions of the refined rule format. However, the associated LTS is not finitely branching because, for instance, $f(l_1) \xrightarrow{l_1} g^i(l_1)$ for each $i \in \mathbb{N}$. Notice that this fact is also true if, additionally, we consider the refinement of the restricted support in Observation 1, because the set of sources of premises for each instantiation of the rule template on the right is a subset of the support for $f(x)$ (i.e., $\{x\} \subseteq \eta(f(x))$).

Observation 3. *If the η-types are refined so that the map ψ considers the sets of pairs of labels and targets in the premises whose sources are in the support, then the variables in the targets of premises have to be used uniformly.*

If the TSS in Example 6 were uniform in the targets of premises, then all the variables y_i would be renamed to y. The TSS thus obtained induces the same LTS as the one above. For every η, all the instantiations of the rule template on the right have η-type $\langle f(x), \psi\rangle$ where $\psi : \eta(f(x)) \to \mathcal{P}_\omega(L)$ is such that

$$\psi(t) \;=\; \begin{cases} \emptyset & \text{if } t \neq x \\ \{(l_1, y)\} & \text{if } t = x \end{cases}$$

and thus the η-types with the previously mentioned refinements would be infinitely inhabited.

7. Introducing the S-types

We introduce a variation of the η-types (we have called them S-types) that incorporate the refinements in Observations 1, 2 and 3, and that provide some of the conditions of a rule format for finite branching that covers Examples 4, 5 and 6 among others. As far as we are aware, our refinements are compatible with the TSSs that are covered already by the rule format on [16] (see Section 4 of [16] for examples). We assume triadic TSS with terms as labels and consider the dyadic transformations of Section 3. The conditions of the rule format are given for the dyadic TSSs.

Recall the strict stratification of [16]. A strict stratification is defined for all closed terms. We notice that the strict stratification and the information relative to the restricted support map of Section 6.1 can be neatly combined into a variation of the strict stratification that, differently from the one in [16], allows certain terms that do not unify with sources of premises in rules to have undefined order. We introduce the partial strict stratification of a dyadic TSS.

Notation 16. Let S be a partial map, we write $S(p) \neq \bot$ to indicate that $S(p)$ is defined.

Definition 17 (Partial strict stratification)**.** Let R be a dyadic TSS and S be a partial map from origins to ordinal numbers. S is a partial strict stratification of R iff the following conditions hold:

(i) $S(\sigma(s)) \neq \bot$, for every source s of some rule in R and for every substitution σ that closes s.

(ii) For every rule $H/s \longrightarrow d$ in R and for every $v \longrightarrow w \in H$, it holds that $S(\sigma(v)) < S(\sigma(s))$ for each substitution σ that closes s and v such that $S(\sigma(v)) \neq \bot$.

We say an origin o has order $S(o)$.

Example 7. Consider the following dyadic TSS:

$$\text{L}\ \frac{}{g(l_1) \longrightarrow (l_1, l_1)} \qquad\qquad \text{R}i\ \frac{g^i(x) \longrightarrow (l_i, x)}{f(x) \longrightarrow (l_1, x)}\ , \qquad i \in \mathbb{N}.$$

Define

$$\begin{array}{rcll} S(g(l_1)) & = & 0 & \\ S(f(p)) & = & 1 & \\ S(q) & = & \bot & \text{otherwise} \end{array}$$

Then S is a partial strict stratification in the sense of Definition 17.

Consider the rule R2 in the example above. The partial strict stratification S given in that example is not defined for terms of the form $g^2(p)$, which are the possible instantiations of the source of the premise of that rule. This witnesses the fact that R2 is a junk rule. In general, a rule for which the sources of its premises do not unify with origins that have a defined order is a junk rule. This is formalized by the following lemma.

Lemma 3 (Junk rules)**.** *Let R be a dyadic TSS and S be a partial strict stratification of R. Let o be an origin. Assume that $\rho = H/s \longrightarrow r \in R$ and $\sigma(s) = o$. If for every substitution τ such that $\tau\sigma$ closes all the terms in H there exists a premise $v \longrightarrow w \in H$ such that $S(\tau\sigma(v)) = \bot$, then ρ cannot unify with the root of a proof tree.*

Proof. The term $\tau\sigma(v)$ does not unify with any rule $\rho' \in R$ because for all the origins o' that unify with rules it must be the case that $S(o') \neq \bot$ by condition (i) in Definition 17. □

The connection between a partial strict stratification and the restricted support map is formalized as follows.

Definition 18 (S-restricted support)**.** Let R be a $\mathcal{D}_k^{prj}$-dyadic TSS and S be a partial strict stratification of R. The S-restricted support map is the map $\eta : \mathbb{S}_k^{prj}(\Sigma) \to \mathcal{P}(\mathbb{S}_k^{prj}(\Sigma))$ given by

$$\eta(s) = \{v \mid \exists\sigma.\ S(\sigma(v)) \neq \bot \text{ and } v \longrightarrow w \text{ is a premise in a rule with source } s\}.$$

For every v in an S-restricted support $\eta(s)$, there is a premise of the form $v \longrightarrow w$ in some rule with source s and v unifies with an origin o such that $S(o) \neq \bot$. By Definition 17, for each σ that closes both v and s, $S(\sigma(v)) < S(\sigma(s))$.

For illustration, take the TSS in Example 7 and the partial strict stratification S defined there. The S-restricted support map is such that

$$\begin{array}{rcll} \eta(g(l_1)) & = & \emptyset, & \\ \eta(f(x)) & = & \{g(x)\} & \text{and} \\ \eta(t) & = & \emptyset & \text{otherwise.} \end{array}$$

Note that only rule R1 has a set of sources of premises that is included in $\eta(f(x))$.

The S-types that we introduce below rely on the notion of a partial strict stratification. The S-types are no longer parametric on a support map η (as was the case in [16]), but on the partial strict stratification S, which uniquely determines the S-restricted support map.

Definition 19 (S-types)**.** Let R be a $\mathcal{D}_k^{prj}$-dyadic TSS, S be a partial strict stratification of R, $\eta : \mathbb{S}_k^{prj}(\Sigma) \to \mathcal{P}(\mathbb{S}_k^{prj}(\Sigma))$ be the associated S-restricted support map, and $\rho = H/s \longrightarrow r$ be a rule of R with source s and premises $H = \{v_i \longrightarrow w_i \mid i \in I\}$.

We say that ρ has S-type $\langle s, \psi\rangle$ iff $\{v_i \mid i \in I\} \subseteq \eta(s)$, and the map $\psi(v) = \{w \mid v \longrightarrow w$ is a premise of $\rho\}$ is such that $\psi : \eta(s) \to \mathcal{P}_\omega(\mathbb{R}_k^{prj}(\Sigma))$, that is, $\psi(v)$ is finite for each $v \in \eta(s)$.

Notice that, for a rule $\rho = H/s \longrightarrow r$ to have a valid S-type $\langle s, \psi\rangle$, the set of sources of premises in H has to be a subset of the support $\eta(s)$ and the elements in the codomain of ψ have to be finite sets.

Consider again the TSS in Example 7, the partial strict stratification S defined there and the S-restricted support η given before. The S-type of the rule L is $\langle g(l_1), \emptyset\rangle$, the S-type of the rule R1 is $\langle f(x), \{g(x) \mapsto \{(l_1, x)\}\}\rangle$, and the rules R$i$ where $i \neq 1$ do not have a valid S-type.

We consider uniformity both in the sources of rules and, as the refinement of Section 6.3 requires, in the targets of premises. We have adapted the notion of uniformity for TSSs from Definition 12 in [16] to dyadic TSSs. (In our previous work [4] uniformity in the targets of premises is already considered when discussing a rule format for image finiteness in plain SOS.)

Definition 20 (Uniform in the sources)**.** Let R be a dyadic TSS. R is uniform in the sources iff for s and s' sources of any two rules in R, either $s = s'$, or otherwise s and s' cannot differ only in the names of their variables.

Definition 21 (Uniform in the targets of premises)**.** Let R be a dyadic TSS. R is uniform in the targets of premises iff for $v \longrightarrow w$ and $v \longrightarrow w'$ premises of any two (not necessarily different) rules in R, either $w = w'$, or otherwise w and w' cannot differ only in the names of their variables.

In a TSS that is uniform in the sources (respectively, targets of premises), each origin (respectively, destination) is a substitution instance of at most finitely many sources (respectively, targets of positive premises) of transition rules.

Proposition 5. *Let R be a $\mathcal{D}_k^{prj}$-dyadic TSS. The following statements hold:*

(i) If R is uniform in the sources, then for each closed term p the set of pairs (t, σ) with $\sigma : \mathrm{var}(t) \to \mathbb{C}(\Sigma)$ such that $\sigma(t) = p$ and t is the source of some rule in R is finite.

(ii) If R is uniform in the targets of positive premises, then for each transition $p \longrightarrow p'$, and for each term t and substitution $\sigma : \mathrm{var}(t) \to \mathbb{C}(\Sigma)$ such that $\sigma(t) = p$, the set of pairs (t', τ) with $\tau : (\mathrm{var}(t') \setminus \mathrm{var}(t)) \to \mathbb{C}(\Sigma)$ such that $\sigma(t) \longrightarrow \tau\sigma(t') = p \longrightarrow p'$ and $t \longrightarrow t'$ is a positive premise of some rule in R is finite.

We omit the proof of Proposition 5, which can be adapted straightforwardly from analogous results for triadic TSSs (see Propositions 2 and 3 in our previous work presented in [4]).

We follow [16] and combine Definitions 19, 20 and 21 into a single condition that will be used later in the rule format of Section 8.

Definition 22 ($\mathcal{D}_k^{prj}$-bounded)**.** Let R be a $\mathcal{D}_k^{prj}$-dyadic TSS. We say that R is $\mathcal{D}_k^{prj}$-bounded iff R is uniform both in the sources of rules and in the target of premises, and there exists a partial strict stratification S of R such that the elements in the codomain of the S-restricted support map are finite sets (i.e., $\eta : \mathbb{S}_k^{prj}(\Sigma) \to \mathcal{P}_\omega(\mathbb{S}_k^{prj}(\Sigma))$) and for every rule $\rho \in R$ with S-type $\langle s, \psi\rangle$, the S-type $\langle s, \psi\rangle$ is finitely inhabited.

We say that R is $\mathcal{D}_k^{prj}$-bounded by S.

The main result in Theorem 1 to follow holds for every partial strict stratification. However, different choices for the map S may filter out different sets of junk rules.

8. Rule format for $\mathcal{D}_k^{prj}$-finiteness

Consider SOS rules of the form $c \xrightarrow{y} c$ or $c \xrightarrow{c} y$, where c is a constant and y is a variable. Axioms of that form can be instantiated to derive transitions $c \xrightarrow{p} c$ and $c \xrightarrow{c} p$ respectively, for each closed term p, and constant c would have infinitely many transitions. The bounded nondeterminism format from [16], which enforces that all the variables in the rules of a TSS are source dependent (see Definition B.5.2 in [15]), prevents the use of axioms of the form above. That is, a rule in bounded nondeterminism format cannot introduce variables spuriously that could break bounded nondeterminism. We adapt the bounded nondeterminism format to dyadic TSSs and we rename it as $\mathcal{D}_k^{prj}$-bounded nondeterminism format.

Definition 23 ($\mathcal{D}_k^{prj}$-bounded nondeterminism format)**.** Let R be a $\mathcal{D}_k^{prj}$-dyadic TSS. A rule in R is in $\mathcal{D}_k^{prj}$-bounded nondeterminism format iff

(i) all the variables in the sources of premises occur also in the source of the rule, and

(ii) all the variables in the target of the rule occur also in the source of the rule, or in the targets of its premises.

A TSS is in $\mathcal{D}_k^{prj}$-bounded nondeterminism format if all its rules are in $\mathcal{D}_k^{prj}$-bounded nondeterminism format.

Consider the above mentioned axioms $c \xrightarrow{y} c$ and $c \xrightarrow{c} y$. Their dyadic counterparts generated using the transformation $\mathcal{D}_1^{id}$ are $c \longrightarrow (y, c)$ and $c \longrightarrow (c, y)$, neither of which is in bounded nondeterminism format. Applying the $\mathcal{D}_4^{id}$ transformation to $c \xrightarrow{y} c$ and $c \xrightarrow{c} y$ yields the dyadic rules $(c, y) \longrightarrow c$ and $(c, c) \longrightarrow y$, the former of which is in bounded nondeterminism format. Applying the $\mathcal{D}_1^{\pi_1}$ transformation to $c \xrightarrow{y} c$ and $c \xrightarrow{c} y$ yields the dyadic axioms $c \longrightarrow y$ and $c \longrightarrow c$, of which the latter is in bounded nondeterminism format.

We are now ready to present our rule format for $\mathcal{D}_k^{prj}$-finiteness.

Theorem 1. *Let R be a $\mathcal{D}_k^{prj}$-bounded TSS in $\mathcal{D}_k^{prj}$-bounded nondeterminism format. The LTS associated with R is finitely branching.*

Proof. We prove that each origin o in the LTS associated with R is finitely branching. Since R is $\mathcal{D}_k^{prj}$-bounded, it is uniform in the sources and there are only finitely many pairs (s, σ) such that $\sigma(s) = o$ and s is the source of some rule of R. We focus on the set $\{(s_i, \sigma_i) \mid i \in I\}$, with I a finite index set such that o unifies with a rule that has some s_i as source. Since R is $\mathcal{D}_k^{prj}$-bounded, there exists a partial strict stratification S such that R is $\mathcal{D}_k^{prj}$-bounded by S. We proceed by induction on $S(o)$. (Recall from Definition 17 that $S(o) \neq \perp$.)

The base case is when $S(o) = 0$. The origin o unifies with rules that have source s_i and that may have valid S-type or not. By Lemma 3, the rules that do not have valid S-type cannot give rise to transitions and can be safely ignored. By Definition 17, the rules that have valid S-type are of the form

$$\frac{}{s_i \longrightarrow r_j}\,, \qquad i \in I,\ j \in J_i$$

where the J_i are taken to be disjoint to avoid proliferation of indices. Since R is $\mathcal{D}_k^{prj}$-bounded, for each $i \in I$ and each $j \in J_i$ the instantiation of the rule template above has S-type $\langle s_i, \psi_j \rangle$, and $\langle s_i, \psi_j \rangle$ is finitely inhabited. By Definitions 18 and 19, $\psi_j = \{v \mapsto \emptyset \mid v \in \eta(s_i)\}$ for each $j \in J_i$. Since R is in $\mathcal{D}_k^{prj}$-bounded nondeterminism format, $\mathrm{var}(r_j) \subseteq \mathrm{var}(s_i)$ and thus the $\sigma_i(r_j)$ are closed. Since for all $j \in J_i$ the S-types $\langle s_i, \psi_j \rangle$ are equal and they are finitely inhabited, the J_i are finite. Therefore, for each $i \in I$ the set $\{\sigma_i(r_j) \mid \sigma_i(s_i) \longrightarrow \sigma_i(r_j) \text{ with } j \in J_i\}$ is finite. By the finiteness of I it follows that the set $\{d \mid o \longrightarrow d\}$ is finite and the theorem holds.

The general case is when $S(o) > 0$. The rules with source s_i are of the form

$$\frac{\{v_\ell \longrightarrow w_\ell \mid \ell \in L_j\}}{s_i \longrightarrow r_j}\,, \qquad i \in I,\ j \in J_i \tag{1}$$

where the J_i and the L_j are taken to be disjoint to avoid proliferation of indices. (Note that L_j may be empty for some rules.) It is safe to ignore all the rules that do not give rise to transitions and therefore, in the remainder of the proof, we assume that each rule ρ of the form in (1) gives rise to transitions. We first show that ρ has a valid S-type.

Since R is in $\mathcal{D}_k^{prj}$-bounded nondeterminism format, $\mathrm{var}(v_\ell) \subseteq \mathrm{var}(s_i)$ for each $\ell \in L_j$, and therefore the $\sigma_i(v_\ell)$ are closed terms. As ρ gives rise to transitions, by Lemma 3 we have that $S(\sigma_i(v_\ell)) \neq \bot$. By Definitions 17 and 18, $\{v_\ell \mid \ell \in L_j\} \subseteq \eta(s_i)$. For every $v \in \eta(s_i)$, if v is not a source v_ℓ of some premise, then by Definition 19 $\psi(v) = \emptyset$, which is a finite set. If v is a source v_ℓ of some premise, then $S(\sigma_i(v)) < S(\sigma_i(s_i))$ and by the induction hypothesis $\sigma_i(v)$ is finitely branching. Since R is uniform in the targets of premises and by Proposition 5, the set $\{w \mid v \longrightarrow w \text{ is a premise of } \rho\}$ is a finite set. By Definition 19, ρ has a valid S-type. We let $\langle s_i, \psi_j\rangle$ be the S-type of ρ.

Our goal is now to prove that there are only finitely many outgoing transitions that can be proved using rules of the form (1). To this end, first of all we show that there are only finitely many distinct ψ_j with $j \in J_i$ such that rules with S-type $\langle s_i, \psi_j\rangle$ give rise to transitions from $\sigma_i(s_i)$. By Definition 19, each rule of S-type $\langle s_i, \psi_j\rangle$ contains a premise of the form $v \longrightarrow w$ for each $v \in \eta(s_i)$ and each $w \in \psi_j(v)$. Since R is in $\mathcal{D}_k^{prj}$-bounded nondeterminism format, $\mathrm{var}(v) \subseteq \mathrm{var}(s_i)$, and thus the $\sigma_i(v)$ are closed. By Definition 7, for each transition in the node of a proof tree, if the transition unifies with a rule of S-type $\langle s_i, \psi_j\rangle$ then for each $v \in \eta(s_i)$ the process $\sigma_i(v)$ can perform, at least, a transition for each $w \in \psi_j(v)$. The ψ_j in the S-types of rules that give rise to transitions from $\sigma_i(s_i)$ are dependent functions of type $\Pi_{v\in\eta(s_i)}\{\sigma_i(v) \longrightarrow \tau\sigma_i(w)\}$ with substitutions $\tau : (\mathrm{var}(w) \setminus \mathrm{var}(v)) \to \mathbb{C}(\Sigma)$. For each i and each $j \in J_i$ the refined type of the ψ_j is finitely inhabited, since the codomain of a dependent function depends on the inputs of the function. Each image of ψ_j cannot be an arbitrary subset of $\mathbb{R}_k^{prj}(\Sigma)$, but only the subset that is determined by the input v. That is, the only elements in the codomain of ψ_j are the sets $\{\tau\sigma_i(w) \mid \sigma_i(v) \longrightarrow \tau\sigma_i(w)\}$ where $v \in \eta(s_i)$. Since the $\eta(s_i)$ are finite sets, both the domain and the codomain of ψ_j are finite. Therefore, for each $i \in I$ there are only finitely many ψ_j with $j \in J_i$ such that the rules with S-type $\langle s_i, \psi_j\rangle$ give rise to transitions from $\sigma_i(s_i)$.

Finally, we consider a rule ρ in the form of (1). Since R is in $\mathcal{D}_k^{prj}$-bounded nondeterminism format, $\mathrm{var}(v_\ell) \subseteq \mathrm{var}(s_i)$ and therefore the $\sigma_i(v_\ell)$ are closed terms. Since $S(\sigma_i(v_\ell)) < S(o)$, by the induction hypothesis, the $\sigma_i(v_\ell)$ are finitely branching, and therefore for each $i \in I$ the set $\{p' \mid \sigma_i(v_\ell) \longrightarrow p'\}$ is finite. Since R is uniform in the targets of premises and by Proposition 5, for each $i \in I$ the set $\{\tau_m\sigma_i(w_\ell) \mid \sigma_i(v_\ell) \longrightarrow \tau_m\sigma_i(w_\ell)\}$ is finite, with

$$\tau_m : ((\bigcup_{\ell\in L_j} \mathrm{var}(w_\ell)) \setminus \mathrm{var}(s_i)) \to \mathbb{C}(\Sigma)$$

closed substitutions where m ranges over finite index sets M_j with $j \in J_i$. Since R is in $\mathcal{D}_k^{prj}$-bounded nondeterminism format, $\mathrm{var}(r_j) \subseteq (\mathrm{var}(s_i) \cup (\bigcup_{\ell\in L_j} \mathrm{var}(w_\ell)))$ and therefore the $\tau_m\sigma_i(r_j)$ are closed targets. Since for each i there are finitely many distinct ψ_j with $j \in J_i$ such that the rules with S-type $\langle s_i, \psi_j\rangle$ give rise to transitions, and since the $\langle s_i, \psi_j\rangle$ are finitely inhabited and the M_j are finite, then for each $i \in I$ the set $\{\tau_m\sigma_i(r_j) \mid \sigma_i(s_i) \longrightarrow \tau_m\sigma_i(r_j)\}$ is finite. By the finiteness of I it follows that the set $\{d \mid o \longrightarrow d\}$ is finite and we are done. □

Remark 2. The use of dependent function types when showing that, for a given i, there are finitely many distinct ψ_j with $j \in J_i$ such that the rules with S-type $\langle s_i, \psi_j\rangle$ give rise to transitions from $\sigma_i(s_i)$, follows from our previous work presented in [4]. However, an alternative argument that does not involve dependent function types stems from [16]. The alternative argument assumes that there exists some $m \in I$ for which there are infinitely many ψ_n with $n \in J_m$ such that rules with S-type $\langle s_m, \psi_n\rangle$ give rise to transitions from $\sigma_m(t_m)$, and then shows that this assumption contradicts the induction hypothesis.

Corollary 1. *Let R be a triadic TSS with terms as labels and let $R' = \mathcal{D}_k^{prj}(R)$. If R' is $\mathcal{D}_k^{prj}$-bounded and in $\mathcal{D}_k^{prj}$-bounded nondeterminism format then R is $\mathcal{D}_k^{prj}$-finite.*

Proof. By Definition 14. □

$$\frac{}{a \xrightarrow{z}_{/a} z} \qquad \frac{a \neq b}{b \xrightarrow{z}_{/a} b} \qquad \frac{x_0 \xrightarrow{z}_{/a} y_0 \quad x_1 \xrightarrow{z}_{/a} y_1}{c!x_0.x_1 \xrightarrow{z}_{/a} c!y_0.y_1} \qquad \frac{x \xrightarrow{z}_{/a} y \quad a \neq b}{c?b.x \xrightarrow{z}_{/a} c?b.y}$$

$$\frac{x_0 \xrightarrow{z}_{/a} y_0 \quad x_1 \xrightarrow{z}_{/a} y_1}{x_0 + x_1 \xrightarrow{z}_{/a} y_0 + y_1} \qquad \frac{x_0 \xrightarrow{z}_{/a} y_0 \quad x_1 \xrightarrow{z}_{/a} y_1}{x_0 \mid x_1 \xrightarrow{z}_{/a} y_0 \mid y_1}$$

$$\frac{x_0 \xrightarrow{z}_{c?} y_0 \quad x_1 \xrightarrow{z}_{c!} y_1}{x_0 \mid x_1 \longrightarrow_{\tau} y_0 \mid y_1} \qquad \frac{x_0 \xrightarrow{z}_{c!} y_0 \quad x_1 \xrightarrow{z}_{c?} y_1}{x_0 \mid x_1 \longrightarrow_{\tau} y_0 \mid y_1}$$

$$\frac{}{c!x_0.x_1 \xrightarrow{x_0}_{c!} x_1} \qquad \frac{x_1 \xrightarrow{z}_{/a} y_1}{c?a.x_1 \xrightarrow{z}_{c?} y_1} \qquad \frac{}{\tau.x \longrightarrow_{\tau} x}$$

$$\frac{x_0 \xrightarrow{z}_{c!} y_0}{x_0 + x_1 \xrightarrow{z}_{c!} y_0} \qquad \frac{x_0 \xrightarrow{z}_{c?} y_0}{x_0 + x_1 \xrightarrow{z}_{c?} y_0} \qquad \frac{x_0 \longrightarrow_{\tau} y_0}{x_0 + x_1 \longrightarrow_{\tau} y_0}$$

$$\frac{x_1 \xrightarrow{z}_{c!} y_1}{x_0 + x_1 \xrightarrow{z}_{c!} y_1} \qquad \frac{x_1 \xrightarrow{z}_{c?} y_1}{x_0 + x_1 \xrightarrow{z}_{c?} y_1} \qquad \frac{x_1 \longrightarrow_{\tau} y_1}{x_0 + x_1 \longrightarrow_{\tau} y_1}$$

$$\frac{x_0 \xrightarrow{z}_{c!} y_0}{x_0 \mid x_1 \xrightarrow{z}_{c!} y_0 \mid x_1} \qquad \frac{x_0 \xrightarrow{z}_{c?} y_0}{x_0 \mid x_1 \xrightarrow{z}_{c?} y_0 \mid x_1} \qquad \frac{x_0 \longrightarrow_{\tau} y_0}{x_0 \mid x_1 \longrightarrow_{\tau} y_0 \mid x_1}$$

$$\frac{x_1 \xrightarrow{z}_{c!} y_1}{x_0 \mid x_1 \xrightarrow{z}_{c!} x_0 \mid y_1} \qquad \frac{x_1 \xrightarrow{z}_{c?} y_1}{x_0 \mid x_1 \xrightarrow{z}_{c?} x_0 \mid y_1} \qquad \frac{x_1 \longrightarrow_{\tau} y_1}{x_0 \mid x_1 \longrightarrow_{\tau} x_0 \mid y_1} ,$$

where a and b are atoms and c is a channel name.

Figure 3: Inference rules for MicroCHOCS.

The conditions of Theorem 1 define our rule format for $\mathcal{D}_k^{prj}$-finiteness for triadic TSS with terms as labels.

9. Application

The rule format of Section 8 can be applied to TSSs with terms as labels from the literature. As an example, we adapt the inference rules for CHOCS from Figure 1 in [27]. Consider the subset of the CHOCS language in Figure 3, which we call MicroCHOCS, which includes operators for send $c!x.p$, receive $c?a.p$, choice $p_1 + p_2$ and parallel composition $p_1 \mid p_2$. The following classes of transitions are defined: $\longrightarrow_{/a}$ where a is an atom, $\longrightarrow_{c!}$ and $\longrightarrow_{c?}$ where c is a channel name, and $\longrightarrow_{\tau}$ where τ is the internal action. These transitions are labelled by arbitrary terms. Note that the x, z, x_0, x_1, y_0 and y_1 are term variables. In words, transition $p \xrightarrow{q}_{/a} p'$ means that process p' is obtained by substituting the free occurrences of atom a in process p by process q. Transition $c!p.q \xrightarrow{p}_{c!} q$ means that process $c!p.q$ sends process p through channel c and continues with process q. Transition $c?a.p \xrightarrow{q}_{c?} p'$ means that process $c?a.p$ receives process q through channel c and continues with process p', which is the result of substituting the free occurrences of atom a in p by q. Transition $p \longrightarrow_{\tau} p'$ means that process p continues with process p' after performing the silent action.

We will now use the theory we developed in the main body of the paper to study the bounded non-determinism properties of the transition relations in MicroCHOCS. We partition the system into the different sub-systems defining each of the classes of transitions. For each atom a, we will show that sub-system $\longrightarrow_{/a}$ is image finite. For each channel name c, we will show that sub-system $\longrightarrow_{c!}$ is finitely branching. For each channel name c, we will show that sub-system $\longrightarrow_{c?}$ is image finite by considering a TSS with the same cardinality that gets rid of transitions $\longrightarrow_{/a}$ and using the fact that, for each atom a, the sub-system $\longrightarrow_{/a}$ is image finite. Finally, we will show that subsystem $\longrightarrow_{\tau}$ is finitely branching by considering a TSS with the same cardinality that gets rid of transitions $\longrightarrow_{c?}$ and $\longrightarrow_{c!}$ and by using the fact that, for each channel name, the sub-system $\longrightarrow_{c?}$ is image finite and the subsystem $\longrightarrow_{c!}$ is finitely branching.

Example 8 (Sub-system $\longrightarrow_{/a}$)**.** The first six rule templates of MicroCHOCS implement capture avoiding substitution, i.e., $t \xrightarrow{z}_{/a} t'$ where term t' results from substituting subject z for the free occurrences of atom a in term t. Since the terms are finite, once the subject and the atom are fixed, there is only one resulting term and the LTS associated with these six rules is image finite. (Note, however, that the system is not finitely branching. Indeed, $a \xrightarrow{p}_{/a} p$ for each closed term p.) This can be checked by applying the $\mathcal{D}_4^{id}$ transformation and the rule format defined by the conditions of Theorem 1. We notice that the atom-inequality conditions in the second and in the fourth rules can only decrease the number of transitions in the associated LTS. Therefore it is safe to strip away the atom inequality conditions, which we omit from now on. We fix an atom a and we let R be the TSS consisting of the instances of the first six rules above that define the class of transitions $\longrightarrow_{/a}$. Below we give the $\mathcal{D}_4^{id}$-dyadic TSS $R' = \mathcal{D}_4^{id}(R)$:

$$\frac{}{(a,z) \longrightarrow z} \qquad \frac{}{(b,z) \longrightarrow b} \qquad \frac{(x_0,z) \longrightarrow y_0 \qquad (x_1,z) \longrightarrow y_1}{(c!x_0.x_1,z) \longrightarrow c!y_0.y_1}$$

$$\frac{(x,z) \longrightarrow y}{(c?b.x,z) \longrightarrow c?b.y} \qquad \frac{(x_0,z) \longrightarrow y_0 \qquad (x_1,z) \longrightarrow y_1}{(x_0+x_1,z) \longrightarrow y_0+y_1}$$

$$\frac{(x_0,z) \longrightarrow y_0 \qquad (x_1,z) \longrightarrow y_1}{(x_0 \mid x_1,z) \longrightarrow y_0 \mid y_1}\ .$$

R' has a partial strict stratification S given by

$$\begin{array}{rcll}
S(b,q) & = & 1 & b \text{ an atom} \\
S(c!p_0.p_1,q) & = & 1 + S(p_0,q) + S(p_1,q) & \\
S(c?b.p,q) & = & 1 + S(p,q) & \\
S(p_0+p_1,q) & = & 1 + S(p_0,q) + S(p_1,q) & \\
S(p_0 \mid p_1,q) & = & 1 + S(p_0,q) + S(p_1,q) & \\
S(p,q) & = & \bot & \text{otherwise}
\end{array}$$

where p, p_0, p_1 and q are closed terms. The S-restricted support map of R' is given by

$$\begin{array}{rcll}
\eta(b,z) & = & \emptyset & b \text{ an atom} \\
\eta(c!x_0.x_1,z) & = & \{(x_0,z),(x_1,z)\} & \\
\eta(c?b.x,z) & = & \{(x,z)\} & \\
\eta(x_0+x_1,z) & = & \{(x_0,z),(x_1,z)\} & \\
\eta(x_0 \mid x_1,z) & = & \{(x_0,z),(x_1,z)\}. &
\end{array}$$

The elements in the codomain of η are finite sets. The rules above have respectively (in the order in which they are read) the S-types

$$\begin{array}{rl}
\langle(a, z), & \emptyset\rangle \\
\langle(b, z), & \emptyset\rangle \\
\langle(c!x_0.x_1, z), & \{(x_0, z) \mapsto \{y_0\}, (x_1, z) \mapsto \{y_1\}\}\rangle \\
\langle(c?a.x, z), & \{(x, z) \mapsto \{y\}\}\rangle \\
\langle(x_0 + x_1, z), & \{(x_0, z) \mapsto \{y_0\}, (x_1, z) \mapsto \{y_1\}\}\rangle \\
\langle(x_0 \mid x_1, z), & \{(x_0, z) \mapsto \{y_0\}, (x_1, z) \mapsto \{y_1\}\}\rangle
\end{array}$$

which are finitely inhabited. Since R' is in $\mathcal{D}_4^{id}$-bounded nondeterminism format, R' meets the conditions of Theorem 1. Therefore R' is finitely branching and by Corollary 1 R is $\mathcal{D}_4^{id}$-finite, that is, R is image finite. Since the atom-inequality conditions that we removed from the second and fourth rules can only decrease the number of transitions, for each atom a, the LTS associated with the rules of MicroCHOCS that define the class of transitions $\longrightarrow_{/a}$ is image finite.

Example 9 (Sub-system $\longrightarrow_{c!}$)**.** The last five rule templates of the first column of the rules for MicroCHOCS in Figure 3 describe the effect of sending a process on a channel. They consist of an axiom and four compatibility rule templates for the choice and parallel composition operator. We check that this sub-system is finitely branching. We fix a channel c and we let R be the TSS consisting of the instances of the rules that define the class of transitions $\longrightarrow_{c!}$. Below we give the $\mathcal{D}_1^{id}$-dyadic TSS $R' = \mathcal{D}_1^{id}(R)$:

$$\frac{}{c!x_0.x_1 \longrightarrow (x_0, x_1)} \qquad \frac{x_0 \longrightarrow (z, y_0)}{x_0 + x_1 \longrightarrow (z, y_0)} \qquad \frac{x_1 \longrightarrow (z, y_1)}{x_0 + x_1 \longrightarrow (z, y_1)}$$

$$\frac{x_0 \longrightarrow (z, y_0)}{x_0 \mid x_1 \longrightarrow (z, y_0 \mid x_1)} \qquad \frac{x_1 \longrightarrow (z, y_1)}{x_0 \mid x_1 \longrightarrow (z, x_0 \mid y_1)}\ .$$

R' has a partial strict stratification S given by

$$\begin{array}{rcll}
S(c!p_0.p_1) & = & 1 & \\
S(p_0 + p_1) & = & 1 + S(p_0) + S(p_1) & \\
S(p_0 \mid p_1) & = & 1 + S(p_0) + S(p_1) & \\
S(p) & = & \perp & \text{otherwise}
\end{array}$$

where p, p_0, p_1 and q are closed terms. The S-restricted support map of R' is given by

$$\begin{array}{rcl}
\eta(c!x_0.x_1) & = & \emptyset \\
\eta(x_0 + x_1, z) & = & \{x_0, x_1\} \\
\eta(x_0 \mid x_1, z) & = & \{x_0, x_1\}.
\end{array}$$

The elements in the codomain of η are finite sets. The rules above have respectively (in the order in which they are read) the S-types

$$\begin{array}{rl}
\langle c!x_0.x_1, & \emptyset\rangle \\
\langle x_0 + x_1, & \{x_0 \mapsto \{(z, y_0)\}, x_1 \mapsto \emptyset\}\rangle \\
\langle x_0 + x_1, & \{x_0 \mapsto \emptyset, x_1 \mapsto \{(z, y_1)\}\}\rangle \\
\langle x_0 \mid x_1, & \{x_0 \mapsto \{(z, y_0)\}, x_1 \mapsto \emptyset\}\rangle \\
\langle x_0 \mid x_1, & \{x_0 \mapsto \emptyset, x_1 \mapsto \{(z, y_1)\}\}\rangle
\end{array}$$

which are finitely inhabited. Since R' is in $\mathcal{D}_1^{id}$-bounded nondeterminism format, R' meets the conditions of Theorem 1. Therefore R' is finitely branching and, by Corollary 1, R is $\mathcal{D}_1^{id}$-finite, that is, R is finitely branching. For each channel name c, the LTS associated with the rules of MicroCHOCS that define the class of transitions $\longrightarrow_{c!}$ is finitely branching.

Example 10 (Sub-system $\longrightarrow_{c?}$)**.** The last five rule templates of the second column of MicroCHOCS (together with the sub-system $\longrightarrow_{/a}$) describes the effect of receiving a process on a channel. They consist of a rule template that uses sub-system $\longrightarrow_{/a}$ in its premise and four compatibility rule templates for the choice and parallel composition operator. This sub-system is essentially the same as the one in Example 9, except for the first rule template

$$\frac{x_1 \xrightarrow{z}_{/a} y_1}{c?a.x_1 \xrightarrow{z}_{c?} y_1}\ .$$

Since for each atom a the class of transitions $\longrightarrow_{/a}$ is image finite (Example 8), in order to test the sub-system $\longrightarrow_{c?}$ for image finiteness it is enough to omit the rules for $\longrightarrow_{/a}$ and to replace the rule template above with the axiom template

$$\frac{}{c?a.x_1 \xrightarrow{z}_{c?} x_1}\ .$$

We check that the LTS associated with the resulting TSS is image finite. We fix a channel c and we let R be the TSS that consists of the instances of the axiom above and of the compatibility rules that define the class of transitions $\longrightarrow_{c?}$. Let $R' = \mathcal{D}_4^{id}(R)$. Below we give the $\mathcal{D}_4^{id}$-dyadic TSS $R' = \mathcal{D}_4^{id}(R)$:

$$\frac{}{(c?a.x_1, z) \longrightarrow x_1} \qquad \frac{(x_0, z) \longrightarrow y_0}{(x_0 + x_1, z) \longrightarrow y_0} \qquad \frac{(x_1, z) \longrightarrow y_1}{(x_0 + x_1, z) \longrightarrow y_1}$$

$$\frac{(x_0, z) \longrightarrow y_0}{(x_0 \mid x_1, z) \longrightarrow y_0 \mid x_1} \qquad \frac{(x_1, z) \longrightarrow y_1}{(x_0 \mid x_1, z) \longrightarrow x_0 \mid y_1}\ .$$

R' has a partial strict stratification S given by

$$\begin{array}{rcll} S(c?a.p, q) &=& 1 & \\ S(p_0 + p_1, q) &=& 1 + S(p_0, q) + S(p_1, q) & \\ S(p_0 \mid p_1, q) &=& 1 + S(p_0, q) + S(p_1, q) & \\ S(p, q) &=& \bot & \text{otherwise} \end{array}$$

where p, p_0, p_1 and q are closed terms. R' has an S-restricted support map given by

$$\begin{array}{rcl} \eta(c?a.x_1, z) &=& \emptyset \\ \eta(x_0 + x_1, z) &=& \{(x_0, z), (x_1, z)\} \\ \eta(x_0 \mid x_1, z) &=& \{(x_0, z), (x_1, z)\}. \end{array}$$

The elements in the codomain of η are finite sets. The rules above have respectively (in the order in which the rules are read) the S-types

$$\begin{array}{rl} \langle (c?a.x_1, z), & \emptyset \rangle \\ \langle (x_0 + x_1, z), & \{(x_0, z) \mapsto \{y_0\}, (x_1, z) \mapsto \emptyset\} \rangle \\ \langle (x_0 + x_1, z), & \{(x_0, z) \mapsto \emptyset, (x_1, z) \mapsto \{y_1\}\} \rangle \\ \langle (x_0 \mid x_1, z), & \{(x_0, z) \mapsto \{y_0\}, (x_1, z) \mapsto \emptyset\} \rangle \\ \langle (x_0 \mid x_1, z), & \{(x_0, z) \mapsto \emptyset, (x_1, z) \mapsto \{y_1\}\} \rangle \end{array}$$

which are finitely inhabited. Since R' is in $\mathcal{D}_4^{id}$-bounded nondeterminism format, R' meets the conditions of Theorem 1. Therefore R' is finitely branching and, by Corollary 1, R is $\mathcal{D}_4^{id}$-finite, that is, R is image finite. Since the LTS for $\longrightarrow_{/a}$ is image finite for each atom a and R is image finite for each channel name c, then the LTS associated with the rules of MicroCHOCS that define the class of transitions $\longrightarrow_{c?}$ is image finite.

Example 11 (Sub-system $\longrightarrow_\tau$)**.** The third row and the last five rules of the third column of MicroCHOCS (together with sub-systems $\longrightarrow_{c!}$ and $\longrightarrow_{c?}$) describe the behaviour of the silent action τ. This sub-system consist of the rules

$$\frac{x_0 \xrightarrow{z}_{c?} y_0 \qquad x_1 \xrightarrow{z}_{c!} y_1}{x_0 \mid x_1 \longrightarrow_\tau y_0 \mid y_1} \qquad \frac{x_0 \xrightarrow{z}_{c!} y_0 \qquad x_1 \xrightarrow{z}_{c?} y_1}{x_0 \mid x_1 \longrightarrow_\tau y_0 \mid y_1} \qquad \frac{}{\tau.x \longrightarrow_\tau x}$$

and four compatibility rules for the choice and parallel composition operator that resemble the ones in Examples 9 and 10 except that they omit the label z that appears there. For each channel name c the class of transitions $\longrightarrow_{c!}$ is finitely branching. Therefore, for the transitions of the form $p \mid q \xrightarrow{\tau} p' \mid q'$ that are provable by the rules in the third row of MicroCHOCS, there are only finitely many process r that unify with the z that labels the premises of such rules. Once r is fixed, since the class of transitions $\longrightarrow_{c?}$ is image finite, there are only finitely many processes p' (notice that they unify with the y_0) and the process $p \mid q$, which unify with $x_0 \mid y_0$, is finitely branching. In order to test the sub-system $\longrightarrow_\tau$ for finite branching it is therefore enough to consider the TSS R that consists of the axioms

$$\frac{}{x_0 \mid x_1 \longrightarrow_\tau x_0 \mid x_1} \qquad \frac{}{\tau.x \longrightarrow_\tau x} .$$

and of the four compatibility rules. We check that the LTS associated with R is finitely branching. Let $R' = \mathcal{D}_1^{id}(R)$ with the rules

$$\frac{}{x_0 \mid x_1 \longrightarrow (0, x_0 \mid x_1)} \qquad \frac{}{\tau.x \longrightarrow (0, x)} \qquad \frac{x_0 \longrightarrow (0, y_0)}{x_0 + x_1 \longrightarrow (0, y_0)}$$

$$\frac{x_1 \longrightarrow (0, y_1)}{x_0 + x_1 \longrightarrow (0, y_1)} \qquad \frac{x_0 \longrightarrow (0, y_0)}{x_0 \mid x_1 \longrightarrow (0, y_0 \mid x_1)} \qquad \frac{x_1 \longrightarrow (0, y_1)}{x_0 \mid x_1 \longrightarrow (0, x_0 \mid y_1)}$$

where 0 is the only constant process of MicroCHOCS. R' has a partial strict stratification S given by

$$\begin{array}{rcll} S(\tau.p) & = & 1 & \\ S(p_0 + p_1) & = & 1 + S(p_0) + S(p_1) & \\ S(p_0 \mid p_1) & = & 1 + S(p_0) + S(p_1) & \\ S(p) & = & \bot & \text{otherwise} \end{array}$$

where p, p_0, p_1 and q are closed terms. The S-restricted support map of R' is given by

$$\begin{array}{rcl} \eta(\tau.x) & = & \emptyset \\ \eta(x_0 + x_1, z) & = & \{x_0, x_1\} \\ \eta(x_0 \mid x_1, z) & = & \{x_0, x_1\}. \end{array}$$

The elements in the codomain of η are finite sets. The rules above have respectively (in the order in which they are read) the S-types

$$\begin{array}{rl} \langle x_0 \mid x_1, & \{x_0 \mapsto \emptyset, x_1 \mapsto \emptyset\}\rangle \\ \langle \tau.x, & \emptyset\rangle \\ \langle x_0 + x_1, & \{x_0 \mapsto \{(z, y_0)\}, x_1 \mapsto \emptyset\}\rangle \\ \langle x_0 + x_1, & \{x_0 \mapsto \emptyset, x_1 \mapsto \{(z, y_1)\}\}\rangle \\ \langle x_0 \mid x_1, & \{x_0 \mapsto \{(z, y_0)\}, x_1 \mapsto \emptyset\}\rangle \\ \langle x_0 \mid x_1, & \{x_0 \mapsto \emptyset, x_1 \mapsto \{(z, y_1)\}\}\rangle \end{array}$$

which are finitely inhabited. Since R' is in $\mathcal{D}_1^{id}$-bounded nondeterminism format, R' meets the conditions of Theorem 1. Therefore R' is finitely branching and, by Corollary 1, R is $\mathcal{D}_1^{id}$-finite, that is, R is finitely branching. Since for each channel name c the LTSs for $\longrightarrow_{c!}$ is finitely branching and $\longrightarrow_{c?}$ is image finite, the LTS associated with the rules of MicroCHOCS that define the class of transitions $\longrightarrow_\tau$ is finitely branching.

10. Future work

The first condition in the $\mathcal{D}_k^{prj}$-bounded nondeterminism format from Definition 23 requires that all the variables in the sources of premises occur also in the source of the rule. This restriction disallows any sort of look-ahead in the rules, and it is easy to construct TSSs that do not meet it and yield infinitely branching LTSs. It is an interesting, and probably challenging, avenue for future work to find ways of relaxing that constraint while preserving bounded nondeterminism.

Another avenue for future work that we find worth pursuing is to investigate whether the kind of dyadic TSSs we have considered in this paper may be used fruitfully to study congruence rule formats for established notions of bisimilarity for CHOCS. Earlier work on notions of bisimulation equivalence based on presentations using *commitments* [26], *residual data types* [7] and *residuals* [29] indicates that such a programme might be successfully carried out using dyadic TSSs. However, much work remains to be done in order to develop this line of work and to assess fully the potential usefulness of dyadic formalisms in the study of congruence properties of notions of bisimilarity for CHOCS-like languages.

In the remainder of this section we present TSSs that we are aware are not covered by our rule format.

A partial strict stratification is not enough to deal with the general case of a TSS with junk rules. The following example illustrates why.

Example 12. Consider the TSS over the signature Σ_0 in Notation 9:

$$\frac{g^{i+1}(x) \xrightarrow{y} z}{g^i(x) \xrightarrow{y} z}\,, \qquad i \in \mathbb{N} \qquad\qquad \frac{g(x) \xrightarrow{y} z}{f(x) \xrightarrow{y} z}\,.$$

The TSS above does not contain any axiom, and therefore all the rules are junk rules (i.e., an infinite collection of premises with sources $g^i(x)$ and increasing i could be stacked over a process $f(p)$ without ever constituting a proof tree). However, the TSS does not have a partial strict stratification because for each $i \in \mathbb{N}$ the premise of the instantiation of the rule template on the left coincides with the source of the instantiation with $i+1$, and then the order of $g^i(p)$ has to be greater than the order of $g^{i+1}(p)$. This is not possible because we would need a partial strict stratification S such that $S(g^1(p)) > S(g^2(p)) > \ldots$ contradicting the well foundedness of the ordinals. In words, the TSS is finitely branching but does not meet the conditions of the rule format. For the rule format to cover this case, the partial strict stratification would have to allow processes unifying with the source of a rule to have undefined order, e.g., $S(f(p)) = \bot$ and $S(g^i(p)) = \bot$ with $i \in \mathbb{N}$ and $p \in \mathbb{C}(\Sigma)$. But then proving that a junk rule cannot give rise to transitions poses a non-trivial challenge.

The bounded nondeterminism format (the $\mathcal{D}_k^{prj}$-bounded nondeterminism in the dyadic case) is sometimes too restrictive, as shown by the following example.

Example 13. Consider the TSS over the signature Σ_0 in Notation 9:

$$\frac{}{f(x) \xrightarrow{l_1} l_1}\,, \qquad i \in \mathbb{N} \qquad\qquad \frac{f(x) \xrightarrow{y} z}{l_1 \xrightarrow{y} z}\,.$$

For each process $f(p)$, the only provable transition is $f(p) \xrightarrow{l_1} l_1$. Therefore, the only transition from l_1 is $l_1 \xrightarrow{l_1} l_1$. However, the rule on the right is not in bounded nondeterminism format because its premise introduces variable x spuriously. In words, the TSS is finitely branching but does not meet the conditions of the rule format. The rule format should accept the rule on the right since the sub-process that unifies with x (i.e., the p in a process $f(p)$) is discarded by the next step in the transition sequence (i.e., $f(p) \xrightarrow{l_1} l_1$). We have not tackled this refinement yet.

Finally, Nominal Structural Operational semantics [13] enriches the SOS formalism by adopting the nominal techniques from [17, 36] to deal with names and many-sorted variable-binding operations within the SOS framework. In a nominal TSS arbitrary terms can label transitions and the rules may include a set of freshness assertions that restrict provability of the rule. In order to apply our results to this setting we will need to extend our rule format to many-sorted signatures and study the impact of the freshness assertions. This is work in progress.

Acknowledgements. We thank the three anonymous referees for their careful reading of our original submission and their insightful comments.

References

[1] S. Abramsky. *Domain theory and the logic of observable properties.* PhD thesis, Department of Computer Science, Queen Mary College, University of London, 1987.

[2] S. Abramsky. A domain equation for bisimulation. *Information and Computation*, 92(2):161–218, 1991.

[3] L. Aceto, W. Fokkink, and C. Verhoef. Structural operational semantics. In A. P. J.A. Bergstra and S. Smolka, editors, *Handbook of Process Algebra*, chapter 3, pages 197–292. Elsevier, 2001.

[4] L. Aceto, Á. García-Pérez, and A. Ingólfsdóttir. Rule formats for bounded nondeterminism in structural operational semantics. In C. W. Probst, C. Hankin, and R. R. Hansen, editors, *Semantics, Logics, and Calculi — Essays Dedicated to Hanne Riis Nielson and Flemming Nielson on the Occasion of Their 60th Birthdays*, volume 9560 of *Lecture Notes in Computer Science*, pages 313–343. Springer, 2016.

[5] L. Aceto and A. Ingólfsdóttir. A characterization of finitary bisimulation. *Information Processing Letters*, 64(3):127–134, 1997.

[6] K. R. Apt and G. D. Plotkin. Countable nondeterminism and random assignment. *Journal of the ACM*, 33(4):724–767, 1986.

[7] J. Bengtson. *Formalising process calculi.* PhD thesis, Faculty of Science and Technology, Uppsala Universitet, 2010.

[8] K. L. Bernstein. A congruence theorem for structured operational semantics of higher-order languages. In J. Mitchell, editor, *13th Symposium on Logic in Computer Science*, pages 153–163. IEEE, 1998.

[9] P. Blackburn, M. de Rijke, and Y. Venema. *Modal logic*, volume 53 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, Cambridge, 2001.

[10] B. Bloom. CHOCOLATE: Calculi of Higher Order COmmunication and LAmbda TErms (preliminary report). In H.-J. Boehm, B. Lang, and D. M. Yellin, editors, *Conference Record of the 21st ACM Symposium on Principles of Programming Languages, Portland, Oregon*, pages 339–347. ACM Press, 1994.

[11] B. Bloom, S. Istrail, and A. R. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268, 1995.

[12] A. K. Chandra. Computable nondeterministic functions. In *19th Annual Symposium on Foundations of Computer Science, Ann Arbor, Michigan, USA, 16–18 October 1978*, pages 127–131. IEEE Computer Society, 1978.

[13] M. Cimini, M. R. Mousavi, M. A. Reniers, and M. J. Gabbay. Nominal SOS. *Electronic Notes in Theoretical Computer Science*, 286:103–116, 2012.

[14] R. de Simone. Higher-level synchronising devices in MEIJE–SCCS. *Theoretical Computer Science*, 37(3):245–267, 1985.

[15] W. Fokkink. *Introduction to Process Algebra.* Springer, 2000.

[16] W. Fokkink and T. D. Vu. Structural operational semantics and bounded nondeterminism. *Acta Informatica*, 39(6-7):501–516, 2003.

[17] M. J. Gabbay and A. Pitts. A new approach to abstract syntax involving binders. In G. Longo, editor, *14th Symposium on Logic in Computer Science, Trento, Italy*, pages 214–224. IEEE Computer Society Press, 1999.

[18] A. van Gelder and K. A. Ross and J. S. Schilpf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–662, 1991.

[19] 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, *4th Annual Symposium on Theoretical Aspects of Computer Science, Passau, Germany*, volume 247 of *Lecture Notes in Computer Science*, pages 336–347. Springer, 1987.

[20] J. F. Groote. Transition system specifications with negative premises. *Theoretical Computer Science*, 118(2):263–299, 1993.

[21] J. F. Groote and F. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, 1992.

[22] D. Harel, A. R. Meyer, and V. R. Pratt. Computability and completeness in logics of programs (preliminary report). In J. E. Hopcroft, E. P. Friedman, and M. A. Harrison, editors, *Proceedings of the 9th Annual ACM Symposium on Theory of Computing, May 4–6, 1977, Boulder, Colorado, USA*, pages 261–268. ACM, 1977.

[23] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *J. ACM*, 32(1):137–161, 1985.

[24] R. M. Keller. Formal verification of parallel programs. *Communications of the ACM*, 19(7):371–384, 1976.

[25] R. Milner. *Communication and concurrency.* PHI Series in computer science. Prentice Hall, 1989.

[26] R. Milner. The polyadic π-calculus: A tutorial. In F. L. Bauer, W. Brauer, and H. Schwichtenberg, editors, *Logic and Algebra of Specification*, pages 203–246. Springer-Verlag, 1993.

[27] M. R. Mousavi, M. Gabbay, and M. A. Reniers. SOS for higher order processes. In M. Abadi and L. de Alfaro, editors, *16th International Conference in Concurrency Theory*, volume 3653 of *Lecture Notes in Computer Science (LNCS)*, pages 308–322. Springer-Verlag, 2005.

[28] M. R. Mousavi, M. A. Reniers, and J. F. Groote. SOS formats and meta-theory: 20 years after. *Theoretical Computer Science*, 373(3):238–272, 2007.

[29] J. Parrow, J. Borgström, L.-H. Eriksson, R. Gutkovas, and T. Weber. Modal Logics for Nominal Transition Systems. In *26th International Conference on Concurrency Theory (CONCUR 2015)*, volume 42 of *LIPIcs*, pages 198–211. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

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

[31] G. D. Plotkin. An operational semantics for CSP. In D. Bjørner, editor, *IFIP TC2 Working Conference on Formal Description of Programming Concepts – II*, pages 199–225. North-Holland, 1983.

[32] G. D. Plotkin. A structural approach to operational semantics. *Journal of Logic and Algebraic Programming*, 60-61:17–139, 2004.

[33] T. C. Przymusinski. The well-founded semantics coincides with the three-valued stable semantics. *Fundamenta Informaticae*, 13(4):445–463, 1990.

[34] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

[35] B. Thomsen. A calculus of higher order communicating systems. In *16th Annual ACM Symposium on Principles of Programming Languages*, pages 143–154. ACM Press, 1989.

[36] C. Urban, A. Pitts, and M. J. Gabbay. Nominal unification. *Theoretical Computer Science*, 323(1-3):473–497, 2004.

[37] F. Vaandrager. Expressiveness results for process algebras. In J. W. de Bakker, W. P. de Roever, and G. Rozenberg, editors, *REX Workshop on Semantics: Foundations and Applications, Beekbergen, The Netherlands*, volume 666 of *Lecture Notes in Computer Science*, pages 609–638. Springer, 1993.