Precongruence Formats for Decorated Trace Semantics*

Bard Bloom
IBM T.J. Watson Research Center
Hawthorne, NY 10532, USA
`bardb@us.ibm.com`

Wan Fokkink†
Dept. of Software Engineering
CWI
PO Box 94079
1090 GB Amsterdam, NL
`wan@cwi.nl`
`http://www.cwi.nl/~wan`

Rob van Glabbeek
Computer Science Department
Stanford University
Stanford, CA 94305-9045, USA
`rvg@cs.stanford.edu`
`http://theory.stanford.edu/~rvg`

This paper explores the connection between semantic equivalences and preorders for concrete sequential processes, represented by means of labelled transition systems, and formats of transition system specifications using Plotkin's structural approach. For several preorders in the linear time – branching time spectrum a format is given, as general as possible, such that this preorder is a precongruence for all operators specifiable in that format. The formats are derived using the modal characterizations of the corresponding preorders.

1 Introduction

Structural operational semantics [35] provides process algebras and specification languages with an interpretation. It generates a *(labelled) transition system* (LTS), in which states are the closed terms over a (single-sorted, first-order) signature, and transitions between states may be supplied with labels. The transitions between states are obtained from a *transition system specification* (TSS), this being a signature together with a set of proof rules called *transition rules*.

In case of a TSS with only positive premises, the associated LTS simply consists of the transitions derivable from the transition rules. In the presence of negative premises it is not always straightforward to associate an LTS to a TSS. One can for instance express that a transition holds if it does not hold. In van Glabbeek [21] the notion of derivability of transitions from a TSS is extended to negated transitions by incorporating a notion of *negation as failure* (cf. [12]): a *supported proof* enables one to derive the negation of a transition by disproving a premise in each substitution instance of each transition rule that carries this transition as its conclusion; a *well-supported proof* (inspired by the notion of a well-supported model [14]) features an even more powerful method to derive the negation of a transition by demonstrating that that transition can not be derived. A TSS is *complete* [21] if for each transition there is a well-supported proof from the TSS either for the transition itself or for its negation. The LTS associated to a complete TSS consists of the transitions for which there is a well-supported proof. An incomplete TSS arguably does not specify a LTS in a meaningful way at all. However, it specifies what one could call a *3-valued* LTS: an LTS in which potential transitions are either present, absent or unknown. This approach to the meaning of transition system specifications can be seen as a proof-theoretic characterization of the

*This paper subsumes most of [6] and part of [20]. An extended abstract of our work appeared as [8], except that in the current paper we do not include the full abstraction results mentioned there. Those results originated from [20] and are planned to appear in a full version of [20].

†Supported by a grant from The Nuffield Foundation.

work of VAN GELDER, ROSS & SCHLIPF [18] in logic programming. The notion of completeness of a TSS also coincides with the notion of *being positive after reduction* of BOL & GROOTE [10].

LTSs can be distinguished from each other by means of a wide range of semantic equivalences and preorders. These preorders are based on the branching structure of LTSs (*simulation* [33], *ready simulation* [9, 29], *bisimulation* [31, 33], *nested simulations* [24]), on execution sequences (*partial traces*, *completed traces*, *accepting traces*), or on decorated versions of execution sequences (*ready pairs* [5, 32, 37], *failure pairs* [5, 11, 13], *ready traces* [3, 36], *failure traces* [34]). In [19], VAN GLABBEEK classified most equivalences and preorders for concrete, sequential processes[1] that occur in the literature, and motivated them by means of testing scenarios, phrased in terms of 'button pushing experiments' on generative and reactive machines. This gave rise to *modal characterizations* of the preorders, i.e. characterizations in terms of the observations that an experimenter could make during a session with a process.

In general a semantic equivalence (or preorder) induced by a TSS is not a *congruence* (resp. *precongruence*), i.e. the equivalence class of a term $f(t_1, \ldots, t_n)$ need not be determined by the equivalence classes of its arguments $t_1, \ldots, t_n$. Being a (pre)congruence is an important property, for instance in order to fit the equivalence (or preorder) into an axiomatic framework. Syntactic formats for TSSs have been developed with respect to several semantic equivalences and preorders, to ensure that such an equivalence or preorder as induced by a TSS in the corresponding format is a (pre)congruence. These formats have helped to avoid repetitive (pre)congruence proofs, and to explore the limits of sensible TSS definitions. A first congruence format for bisimulation equivalence was put forward by DE SIMONE [38], which was extended to the *GSOS* format by BLOOM, ISTRAIL & MEYER [9] and to the *tyft/tyxt* format by GROOTE & VAANDRAGER [24]. The tyft/tyxt format was extended with negative premises [10, 23] to obtain the *ntyft/ntyxt* format for complete TSSs. The ntyft/ntyxt format also generalizes the GSOS format. The congruence results of [10, 23, 24] were originally only proved for TSSs satisfying a *well-foundedness* criterion; in [17] they were shown to hold for all complete TSSs in ntyft/ntyxt format. To mention some formats for other equivalences and preorders, VAANDRAGER [39] observed that de Simone's format is a precongruence format for the partial trace and the failure preorders, BLOOM [7] introduced a more general congruence format for partial trace equivalence, and FOKKINK [15] introduced a precongruence format for the accepting trace preorder. Finally, BLOOM [6] introduced a congruence format for readiness equivalence, and VAN GLABBEEK [20] one for ready simulation equivalence. This *ready simulation* format generalizes the GSOS format.

In this paper precongruence formats are proposed for several semantic preorders based on decorated traces, building on results reported in [6, 20]. We introduce precongruence formats for the ready trace preorder, the readiness preorder, the failure trace preorder and the failure preorder. The precongruence formats for the last two preorders coincide. Following [6, 15, 16], these three precongruence formats distinguish between *frozen* and *liquid* arguments of function symbols. This distinction is used in posing restrictions on occurrences of variables in transition rules. The ready simulation format of [20] is more liberal than the format for the ready trace preorder, which is more liberal than the format for the readiness preorder, which is more liberal than the format for the failure trace preorder, which in turn is more liberal than de Simone's format.

The precongruence formats introduced in this paper apply to incomplete TSSs as well. For this purpose the definitions of the corresponding preorders are extended to 3-valued LTSs.

[1] A process is *sequential* if it can do only one action at a time; *concrete* refers to the absence of internal actions or internal choice.

We also show that the tyft/tyxt format is adequate for the nested simulation preorders.

The precongruence formats put forward in this paper were obtained by a careful study of the modal characterizations of the preorders in question. The outline of the proof underlying each of our precongruence results is as follows. First, any TSS in our ready simulation format is transformed into an equivalent TSS—equivalent in the sense that is proves the same transitions and negated transitions—of a special form, in which the left-hand sides of positive premises are single variables. For such TSSs we show that the notions of supported and well-supported provability coincide. Next, any such TSS is extended with a number of transition rules with negative conclusions, in such a way that a (negated) transition has a supported proof from the original TSS if and only if it has a standard proof from the extended TSS. In the extended TSS, the left-hand sides of positive and negative premises can further be reduced to single variables. It is shown for each of the precongruence formats in this paper that its syntactic criteria are preserved under these transformations. Finally, the resulting transition rules are used to decompose modal formulas; that is, a modal formula φ for an open term t is decomposed into a choice of modal formulas $\psi(x)$ for variables x such that t satisfies φ if and only if for one of those ψs the variables x in t satisfy $\psi(x)$. The precongruence format for each preorder under consideration guarantees that if a modal formula is within the modal characterization of this preorder, then the same holds for the resulting decomposed modal formulas. This implies the desired precongruence result.

Larsen [28] and Larsen & Xinxin [30] obtained compositionality results for the *Hennessy-Milner logic* [25] and the *μ-calculus* [27], respectively, with respect to *action transducers*, which constitute a reformulation of TSSs in de Simone's format. The technique for decomposing modal formulas that is employed in the current paper is firmly related to their approach, but applies to the richer ready simulation format.

This paper is set up as follows. Section 2 gives definitions of existing semantic preorders, both in terms of decorated traces or simulations and of observations. Section 3 presents the basics of structural operational semantics, and extends the definitions of most of the preorders to 3-valued LTSs. Section 4 recalls the ntyft/ntyxt format and formulates extra requirements to obtain new precongruence formats for a range of preorders. In Section 5 it is shown that the syntactic restrictions of the respective precongruence formats are preserved under *irredundant* provability. Section 6 explains how well-supported proofs are reduced to standard proofs. In Section 7 it is shown that left-hand sides of premises can be reduced to single variables. Section 8 contains the proofs of the precongruence results, based on material in the previous three sections, and using the definitions of preorders in terms of observations. In Section 9, counterexamples are given to show that all syntactic restrictions are essential for the obtained precongruence results. Section 10 presents some applications of the precongruence formats to TSSs from the literature. In Section 11 it is argued that our techniques apply to other preorders as well. As an example we show that the precongruence format for the failure trace preorder is a congruence format for partial trace equivalence, and that the positive variant of this format is a precongruence format for the partial trace preorder. In Section 12, a conservative extension result is established for incomplete TSSs in ready simulation format. Finally, Section 13 discusses possible extensions of the congruence formats derived in this paper.

2 Preorders and equivalences on labelled transition systems

Definition 1 A *labelled transition system (LTS)* is a pair $(\mathbb{P}, \rightarrow)$ with $\mathbb{P}$ a set (of *processes*) and $\rightarrow \subseteq \mathbb{P} \times A \times \mathbb{P}$ for A a set (of *actions*).

Notation: Write $p \xrightarrow{a} q$ for $(p,a,q) \in\rightarrow$ and $p \stackrel{a}{\not\longrightarrow}$ for $\neg\exists q \in \mathbb{P} : p \xrightarrow{a} q$.

The elements of $\mathbb{P}$ represent the processes we are interested in, and $p \xrightarrow{a} q$ means that process p can evolve into process q while performing the action a. We start with defining six preorders on LTSs, which are based on execution sequences of processes. Given an LTS, we write $p \xrightarrow{\varsigma} q$, where $\varsigma = a_1 \cdots a_n \in A^*$ for $n \in \mathbb{N}$, if there are processes $p_0, \ldots, p_n$ with $p = p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} p_n = q$. For a process p we define

$$\mathsf{initials}(p) \quad = \quad \{a \in A \mid \exists p' \in \mathbb{P} : p \xrightarrow{a} p'\}.$$

Definition 2 Assume an LTS.

- A sequence $\varsigma \in A^*$ is a *(partial) trace* of process p if $p \xrightarrow{\varsigma} q$ for some process q.
- $\varsigma \in A^*$ is a *completed trace* of process p if $p \xrightarrow{\varsigma} q$ for some process q with $\mathsf{initials}(q) = \emptyset$.
- A pair (ς, X) with $\varsigma \in A^*$ and $X \subseteq A$ is a *ready pair* of process p if $p \xrightarrow{\varsigma} q$ for some process q with $\mathsf{initials}(q) = X$.
- A pair (ς, X) with $\varsigma \in A^*$ and $X \subseteq A$ is a *failure pair* of process p if $p \xrightarrow{\varsigma} q$ for some process q with $\mathsf{initials}(q) \cap X = \emptyset$.
- A sequence $X_0 a_1 X_1 \cdots a_n X_n$ (with $n \in \mathbb{N}$), where $X_i \subseteq A$ and $a_i \in A$ for $i = 0, \ldots, n$, is a *ready trace* of process p_0 if $p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} p_n$ and $\mathsf{initials}(p_i) = X_i$ for $i = 0, \ldots, n$.
- A sequence $X_0 a_1 X_1 \cdots a_n X_n$ (with $n \in \mathbb{N}$), where $X_i \subseteq A$ and $a_i \in A$ for $i = 0, \ldots, n$, is a *failure trace* of process p_0 if $p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} p_n$ and $\mathsf{initials}(p_i) \cap X_i = \emptyset$ for $i = 0, \ldots, n$.

We write $p \sqsubseteq_T q$, $p \sqsubseteq_C q$, $p \sqsubseteq_R q$, $p \sqsubseteq_F q$, $p \sqsubseteq_{RT} q$ or $p \sqsubseteq_{FT} q$ if the set of (partial) traces, completed traces, ready pairs, failure pairs, ready traces or failure traces of p is included in that of q, respectively. Write $p \sqsubseteq_{CT} q$ for $p \sqsubseteq_T q \wedge p \sqsubseteq_C q$; the preorder $\sqsubseteq_C$ is used only to define $\sqsubseteq_{CT}$.

We proceed to define preorders based on simulation of one process by another.

Definition 3 Assume an LTS. For a binary relation R, let $pR^{-1}q$ iff qRp.

- A binary relation R on processes is a *(1-nested) simulation* if whenever pRq and $p \xrightarrow{a} p'$, then there is a transition $q \xrightarrow{a} q'$ such that $p'Rq'$.
- A simulation R is a *ready simulation* if whenever pRq and $p \stackrel{a}{\not\longrightarrow}$, then $q \stackrel{a}{\not\longrightarrow}$.
- A simulation R is an n-nested simulation (for $n \geq 2$) if R^{-1} is contained in an $(n-1)$-nested simulation.
- A *bisimulation* is a simulation R such that also R^{-1} is a simulation.

We write $p \sqsubseteq_{nS} q$ (for $n \geq 1$), $p \sqsubseteq_{RS} q$ or $p \sqsubseteq_B q$ if there is an n-nested simulation, ready simulation or bisimulation R with pRq, respectively.

Note that $\sqsubseteq_B$ is symmetric, so that it constitutes an equivalence relation. In the literature this notion of *bisimulation equivalence* is often denoted by $\leftrightarroweq$.

In [19], VAN GLABBEEK observed that if one restricts to the domain of finitely branching, concrete, sequential processes, then most semantic preorders found in the literature "that can be defined uniformly in terms of action relations" coincide with one of the preorders defined above. He motivated these preorders by means of testing scenarios, phrased in terms of 'button pushing experiments' on generative and reactive machines. This gave rise to *modal* characterizations of the preorders, characterizations in terms of the *observations* that an experimenter could make during a session with a process.

Definition 4 Assume an action set A. The set $\mathbb{O}$ of *potential observations* or *modal formulas* is defined inductively by:

$\top \in \mathbb{O}$. The trivial observation, obtained by terminating the session.

$a\varphi \in \mathbb{O}$ if $\varphi \in \mathbb{O}$ and $a \in A$. The observation of an action a, followed by the observation φ.

$\widetilde{a} \in \mathbb{O}$ for $a \in A$. The observation that the process cannot perform the action a.

$\bigwedge_{i\in I} \varphi_i \in \mathbb{O}$ if $\varphi_i \in \mathbb{O}$ for all $i \in I$. The process admits each of the observations φ_i.

$\neg\varphi \in \mathbb{O}$ if $\varphi \in \mathbb{O}$. (It can be observed that) φ cannot be observed.

Definition 5 Let $(\mathbb{P}, \rightarrow)$ be a LTS, labelled over A. The *satisfaction relation* $\models \subseteq \mathbb{P} \times \mathbb{O}$ telling which observations are possible for which process is inductively defined by the clauses below.

$$\begin{array}{ll} p \models \top & \\ p \models a\varphi & \text{if } \exists q : p \xrightarrow{a} q \wedge q \models \varphi \\ p \models \widetilde{a} & \text{if } p \stackrel{a}{\not\longrightarrow} \\ p \models \bigwedge_{i\in I} \varphi_i & \text{if } p \models \varphi_i \text{ for all } i \in I \\ p \models \neg\varphi & \text{if } p \not\models \varphi \end{array}$$

We will use the binary conjunction $\varphi_1 \wedge \varphi_2$ as an abbreviation of $\bigwedge_{i\in\{1,2\}} \varphi_i$, whereas $\top$ is identified with the empty conjunction. We identify formulas that are logically equivalent using the laws for conjunction $\top \wedge \varphi \cong \varphi$ and $\bigwedge_{i\in I}(\bigwedge_{j\in J_i} a_{ij}) \cong \bigwedge_{k\in K} a_k$ where $K = \{ij \mid i \in I \wedge j \in J_i\}$. This is justified because $\varphi \cong \psi$ implies $p \models \varphi \Leftrightarrow p \models \psi$.

Definition 6 Below, several sublanguages of the set $\mathbb{O}$ of observations are defined.

$\mathbb{O}_T$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_T)$ *(partial) trace* observations

$\mathbb{O}_{CT}$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_{CT}) \mid \bigwedge_{a\in A} \widetilde{a}$ *completed trace* observations

$\mathbb{O}_F$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_F) \mid \bigwedge_{i\in I} \widetilde{a_i}$ *failure* observations

$\mathbb{O}_R$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_R) \mid \bigwedge_{i\in I} \widetilde{a_i} \wedge \bigwedge_{j\in J} b_j\top$ *readiness* observations

$\mathbb{O}_{FT}$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_{FT}) \mid \bigwedge_{i\in I} \widetilde{a_i} \wedge \varphi' \ (\varphi' \in \mathbb{O}_{FT})$ *failure trace* observations

$\mathbb{O}_{RT}$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_{RT}) \mid \bigwedge_{i\in I} \widetilde{a_i} \wedge \bigwedge_{j\in J} b_j\top \wedge \varphi' \ (\varphi' \in \mathbb{O}_{RT})$ *ready trace* observations

$\mathbb{O}_{1S}$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_{1S}) \mid \bigwedge_{i\in I} \varphi_i \ (\varphi_i \in \mathbb{O}_{1S})$ *(1-nested) simulation* observations

$\mathbb{O}_{RS}$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_{RS}) \mid \widetilde{a} \mid \bigwedge_{i\in I} \varphi_i \ (\varphi_i \in \mathbb{O}_{RS})$ *ready simulation* observations

$\mathbb{O}_{nS}$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_{nS}) \mid \bigwedge_{i\in I} \varphi_i \ (\varphi_i \in \mathbb{O}_{nS}) \mid \neg\varphi' \ (\varphi' \in \mathbb{O}_{(n-1)S})$
n-nested simulation observations (for $n \geq 2$)

$\mathbb{O}_B$ $\quad \varphi ::= \top \mid a\varphi' \ (\varphi' \in \mathbb{O}_B) \mid \bigwedge_{i\in I} \varphi_i \ (\varphi_i \in \mathbb{O}_B) \mid \neg\varphi' \ (\varphi' \in \mathbb{O}_B)$ *bisimulation* observations

For each of these notions N, the set of *N-observations of p* is $\mathcal{O}_N(p) := \{\varphi \in \mathbb{O}_N \mid p \models \varphi\}$.

Theorem 1 For $N \in \{T, CT, F, R, FT, RT, RS, nS \ (n \geq 1), B\}$, $p \sqsubseteq_N q$ iff $\mathcal{O}_N(p) \subseteq \mathcal{O}_N(q)$.

In fact a slight variation of this result will be needed in this paper.

Definition 7 For N as above, let $\mathbb{O}_N^\wedge$ consist of all formulas $\bigwedge_{i \in I} \varphi_i$ with $\varphi_i \in \mathbb{O}_N$. Let $\mathcal{O}_N^\wedge(p) := \{\varphi \in \mathbb{O}_N^\wedge \mid p \models \varphi\}$.

Then clearly $\mathcal{O}_N(p) \subseteq \mathcal{O}_N(q) \Leftrightarrow \mathcal{O}_N^\wedge(p) \subseteq \mathcal{O}_N^\wedge(q)$. In case $N \in \{RS, nS\ (n \geq 1), B\}$ there is, up to logical equivalence, no difference between $\mathbb{O}_N$ and $\mathbb{O}_N^\wedge$, or between $\mathcal{O}_N(p)$ and $\mathcal{O}_N^\wedge(p)$.

Corollary 1 For $N \in \{T, CT, F, R, FT, RT, RS, nS\ (n \geq 1), B\}$, $p \sqsubseteq_N q$ iff $\mathcal{O}_N^\wedge(p) \subseteq \mathcal{O}_N^\wedge(q)$.

The following hierarchy follows immediately from the definitions of the respective preorders in terms of observations; see [19]. In any LTS,

$$
\begin{tikzcd}
 & & & & \sqsubseteq_R \arrow[dr] & & \\
\sqsubseteq_B \arrow[r] & \sqsubseteq_{2S} \arrow[r] & \sqsubseteq_{RS} \arrow[r] \arrow[ddrr] & \sqsubseteq_{RT} \arrow[ur] \arrow[dr] & & \sqsubseteq_F \arrow[r] & \sqsubseteq_{CT} \arrow[dd] \\
 & & & & \sqsubseteq_{FT} \arrow[ur] & & \\
 & & & & \sqsubseteq_{1S} \arrow[rr] & & \sqsubseteq_T
\end{tikzcd}
$$

where a directed edge from one preorder to another means that the source of the edge is included in (i.e. is a finer preorder than) the target.

For every preorder $\sqsubseteq_N$ defined above there exists an associated equivalence $=_N$ (the *kernel* of $\sqsubseteq_N$) given by $p =_N q$ iff $p \sqsubseteq_N q \wedge q \sqsubseteq_N p$. As $\sqsubseteq_B$ is symmetric, one has $p =_B q \Leftrightarrow q \sqsubseteq_B p$. Obviously $p =_N q$ iff $\mathcal{O}_N(p) = \mathcal{O}_N(p)$ iff $\mathcal{O}_N^\wedge(p) = \mathcal{O}_N^\wedge(p)$. The same inclusion hierarchy given for the preorders above applies to their kernels.

3 Structural operational semantics

In this paper V and A are two sets of *variables* and *actions*. V should be infinite and at least at large as A (i.e. $|V| \geq |A|$). Many concepts that will appear later on are parameterized by the choice of V and A, but as in this paper this choice is fixed, a corresponding index is suppressed. A syntactic object is called *closed* if it does not contain any variables from V.

Definition 8 A *signature* is a collection Σ of *function symbols* $f \notin V$, with $|\Sigma| \leq |V|$, equipped with a function $ar : \Sigma \to \mathbb{N}$. The set $\mathbb{T}(\Sigma)$ of *terms* over a signature Σ is defined recursively by:

- $V \subseteq \mathbb{T}(\Sigma)$,
- if $f \in \Sigma$ and $t_1, \ldots, t_{ar(f)} \in \mathbb{T}(\Sigma)$ then $f(t_1, \ldots, t_{ar(f)}) \in \mathbb{T}(\Sigma)$.

A term $c()$ is abbreviated as c. For $t \in \mathbb{T}(\Sigma)$, $var(t)$ denotes the set of variables that occur in t. $T(\Sigma)$ is the set of closed terms over Σ, i.e. the terms $t \in \mathbb{T}(\Sigma)$ with $var(t) = \emptyset$. A Σ-*substitution* σ is a partial function from V to $\mathbb{T}(\Sigma)$. If σ is a substitution and S is any syntactic object, then $\sigma(S)$ denotes the object obtained from S by replacing, for x in the domain of σ, every occurrence of x in S by $\sigma(x)$. In that case $\sigma(S)$ is called a *substitution instance* of S. A Σ-substitution is *closed* if it is a total function from V to $T(\Sigma)$.

Definition 9 Let Σ be a signature. A *positive Σ-literal* is an expression $t \xrightarrow{a} t'$ and a *negative Σ-literal* an expression $t \stackrel{a}{\not\longrightarrow}$ with $t, t' \in \mathbb{T}(\Sigma)$ and $a \in A$. For $t, t' \in \mathbb{T}(\Sigma)$ the literals $t \xrightarrow{a} t'$ and $t \stackrel{a}{\not\longrightarrow}$ are said to *deny* each other. A *transition rule* over Σ is an expression of the form $\frac{H}{\alpha}$ with H a set of Σ-literals (the *premises* of the rule) and α a Σ-literal (the *conclusion*). The left- and right-hand side (if any) of α are called the *source* and the *target* of the rule, respectively. A rule $\frac{H}{\alpha}$ with $H = \emptyset$ is also written α. A *transition system specification (TSS)*, written (Σ, R), consists of a signature Σ and a collection R of transition rules over Σ. A TSS is *standard* if all its rules have positive conclusions, and *positive* if moreover all premises of its rules are positive.

The concept of a positive TSS was introduced by GROOTE & VAANDRAGER [24]; negative premises were added by GROOTE [23]. The resulting notion constitutes the first formalisation of PLOTKIN's *structural operational semantics* [35] that is sufficiently general to cover most of its applications. TSSs with negative conclusions are introduced here, because they are needed as intermediate steps in our proofs for standard TSSs.

The following definition tells when a literal is provable from a TSS. It generalises the standard definition (see e.g. [24]) by allowing the derivation of transition rules. The derivation of a literal α corresponds to the derivation of the transition rule $\frac{H}{\alpha}$ with $H = \emptyset$. The case $H \neq \emptyset$ corresponds to the derivation of α under the assumptions H.

Definition 10 Let $P = (\Sigma, R)$ be a TSS. An *irredundant proof* of a transition rule $\frac{H}{\alpha}$ from P is a well-founded, upwardly branching tree of which the nodes are labelled by Σ-literals, and some of the leaves are marked "hypothesis", such that:

- the root is labelled by α,
- H is the set of labels of the hypotheses, and
- if β is the label of a node q which is not a hypothesis and K is the set of labels of the nodes directly above q, then $\frac{K}{\beta}$ is a substitution instance of a transition rule in R.

A *proof* of $\frac{K}{\alpha}$ from P is an irredundant proof of $\frac{H}{\alpha}$ from P with $H \subseteq K$. If an (irredundant) proof of $\frac{K}{\alpha}$ from P exists, then $\frac{K}{\alpha}$ is *(irredundantly) provable* from P, notation $P \vdash \frac{K}{\alpha}$ (resp. $P \vdash_{\mathrm{irr}} \frac{K}{\alpha}$). We write $P \vdash \frac{K}{H}$ (resp. $P \vdash H$) for sets of literals K and H if $P \vdash \frac{K}{\alpha}$ (resp. $P \vdash \alpha$) for all $\alpha \in H$.

The main purpose of a TSS (Σ, R) is to specify a *transition relation* over Σ, this being a set of closed positive Σ-literals (*transitions*). A positive TSS specifies a transition relation in a straightforward way as the set of all provable transitions. But as pointed out by GROOTE [23], it is much less trivial to associate a transition relation to a TSS with negative premises; in particular there are TSSs that appear not to specify a transition relation in a meaningful way at all. In VAN GLABBEEK [21] eleven answers to the questions "*Which TSSs are meaningful, and which transition relations do they specify?*" are reviewed. The "most general solution without undesirable properties" is due to VAN GELDER, ROSS & SCHLIPF [18] in the setting of logic programming, and has been adapted to TSSs by BOL & GROOTE [10]. In [21] it has been reformulated in terms of completeness with respect to a notion of provability of closed literals that incorporates a form of *negation as failure*.

Definition 11 Let $P = (\Sigma, R)$ be a standard TSS. A *well-supported proof* of a closed literal α from P is a well-founded, upwardly branching tree of which the nodes are labelled by closed Σ-literals, such that:

- the root is labelled by α, and

- if β is the label of a node q and K is the set of labels of the nodes directly above q, then
  1. either $\frac{K}{\beta}$ is a closed substitution instance of a transition rule in R
  2. or β is negative and for every set N of negative closed literals such that $P \vdash \frac{N}{\gamma}$ for γ a closed literal denying β, a literal in K denies one in N.

α is *ws-provable* from P, notation $P \vdash_{ws} \alpha$, if a well-supported proof of α from P exists.

Note that the proof-steps 1 and 2 establish the validity of β when K is the set of literals established earlier. Step 2 allows to infer $t \stackrel{a}{\not\longrightarrow}$ whenever it is manifestly impossible to infer $t \xrightarrow{a} t'$ for some term t' (because every conceivable proof of $t \xrightarrow{a} t'$ involves a premise that has already been refuted). This practice is sometimes referred to as *negation as failure* [12].

Definition 12 A standard TSS P is *complete* if for any closed literal $t \stackrel{a}{\not\longrightarrow}$ either $P \vdash_{ws} t \xrightarrow{a} t'$ for some closed term t' or $P \vdash_{ws} t \stackrel{a}{\not\longrightarrow}$.

Now a standard TSS is meaningful, in the sense that it specifies a transition relation, iff it is complete. The specified transition relation is then the set of all ws-provable transitions.

In the present paper this solution is extended by considering all standard TSSs to be meaningful. However, following VAN GELDER, ROSS & SCHLIPF [18], the meaning of an incomplete TSS is now not given by a two-valued transition relation as defined above, but by a three-valued transition relation, in which a potential transition can be *true*, *false* or *unknown*. In fact, a slight abstraction of this notion will suffice, in which a transition relation is simply defined as a set of closed, positive and negative, literals.

Definition 13 Let Σ be a signature. A *(3-valued) transition relation* over Σ is a set of closed Σ-literals, not containing literals that deny each other. A transition relation $\rightarrow$ is *2-valued* if it satisfies $(t \stackrel{a}{\not\longrightarrow}) \in \rightarrow \quad \Leftrightarrow \quad \neg\exists t' \in T(\Sigma) : (t \xrightarrow{a} t') \in \rightarrow$. The transition relation *associated* to a standard TSS (Σ, R) is the set of closed Σ-literals that are ws-provable from that TSS.

In [21] is has been shown that $\vdash_{ws}$ is consistent, in the sense that no standard TSS admits well-supported proofs of two literals that deny each other. Thus the transition relation associated to a standard TSS is indeed a transition relation as defined above. Note that if a standard TSS is complete, its associated transition relation is 2-valued. This means that the negative literals in its associated transition relation are completely determined by the positive ones; hence the transition relation can be simply given by its positive part.

In Section 2 several preorders have been defined on labelled transition systems, and provided with modal characterisations. These definitions and results apply immediately to the 2-valued transition relation $\rightarrow$ associated to a complete TSS (Σ, R), for such a transition relation gives rise to the LTS $(T(\Sigma), \rightarrow)$. In the case of $N \in \{T, CT, F, R, FT, RT, 1S, RS\}$, the definition of the *N-preorder induced by a TSS* even extends to incomplete TSSs, namely as follows.

Definition 14 Let $P = (\Sigma, R)$ be a standard TSS. The *satisfaction relation* $\models_P \subseteq T(\Sigma) \times \mathbb{O}_{RS}$ is inductively defined by the clauses below (in which $p, q \in T(\Sigma)$).

$$\begin{array}{ll}
p \models_P \top & \\
p \models_P a\varphi & \text{if } \exists q : P \vdash_{ws} p \xrightarrow{a} q \land q \models_P \varphi \\
p \models_P \widetilde{a} & \text{if } P \vdash_{ws} p \stackrel{a}{\not\longrightarrow} \\
p \models_P \bigwedge_{i \in I} \varphi_i & \text{if } p \models_P \varphi_i \text{ for all } i \in I
\end{array}$$

For $N \in \{T, CT, F, R, FT, RT, 1S, RS\}$, the set of *$N$-observations of* $p \in T(\Sigma)$ is given by $\mathcal{O}_N^P(p) := \{\varphi \in \mathbb{O}_N \mid p \models_P \varphi\}$. Likewise, $\mathcal{O}_N^{\wedge P}(p) := \{\varphi \in \mathbb{O}_N^{\wedge} \mid p \models_P \varphi\}$. The *$N$-preorder induced by P* is defined by $p \sqsubseteq_N^P q$ if $\mathcal{O}_N^P(p) \subseteq \mathcal{O}_N^P(q)$. When clear from the context, sub- or superscripts P will be omitted.

Note that in case P is complete, $\sqsubseteq_N^P$ is indeed the N-preorder as defined in Section 2 on the LTS $(T(\Sigma), \rightarrow)$, where $\rightarrow$ is the 2-valued transition relation associated to P. In the general case, the 3-valued transition relation associated to P gives rise to a structure $(T(\Sigma), \rightarrow, \not\rightarrow)$ with $\rightarrow \subseteq \mathbb{P} \times A \times \mathbb{P}$ and $\not\rightarrow \subseteq \mathbb{P} \times A$. Such a structure could be called a *3-valued LTS*. Now $p \stackrel{a}{\not\rightarrow}$ stands for $(p, a) \in \not\rightarrow$, and Theorem 1 can be interpreted as the *definition* of $\sqsubseteq_N$ (agreeing with Definition 14 above). Note that for $N \in \{T, CT, F, R, FT, RT, 1S, RS\}$ Corollary 1 extends to the preorders induced by incomplete TSSs.

Definition 14 does not capture the modality $\neg$ of Definition 4; hence the bisimulation equivalence and n-nested simulation preorders (for $n \geq 2$) induced by a TSS are defined for complete TSSs only. (Extending Definition 14 in a simple-minded way would invalidate the hierarchy of Section 2.)

Definition 15 Let Σ be a signature. A preorder $\sqsubseteq$ on $T(\Sigma)$ is a *precongruence* if for all $f \in \Sigma$

$$p_i \sqsubseteq q_i \text{ for } i = 1, \ldots, ar(f) \quad \Rightarrow \quad f(p_1, \ldots, p_{ar(f)}) \sqsubseteq f(q_1, \ldots, q_{ar(f)}).$$

This is equivalent to the requirement that for all $t \in \mathbb{T}(\Sigma)$ and closed substitutions $\sigma, \sigma' : V \rightarrow T(\Sigma)$

$$\sigma(x) \sqsubseteq \sigma'(x) \text{ for } x \in var(t) \quad \Rightarrow \quad \sigma(t) \sqsubseteq \sigma'(t).$$

In case $\sqsubseteq$ is an equivalence relation as well as a precongruence, it is called a *congruence*. Note that if $\sqsubseteq_N$ is a precongruence, its kernel is a congruence. Thus by establishing precongruence results for the preorders $\sqsubseteq_N$, we also obtain congruence results for the associated equivalences $=_N$.

4 Precongruence formats

In this section we define the formats for TSSs that play a rôle in this paper and state the precongruence results that we are going to establish.

Definition 16 An *ntytt rule* is a transition rule in which the right-hand sides of positive premises are variables that are all distinct, and that do not occur in the source. An ntytt rule is an *ntyxt rule* if its source is a variable, and an *ntyft rule* if its source contains exactly one function symbol and no multiple occurrences of variables. An ntytt rule (resp. ntyft rule) is an *nxytt rule* (resp. *nxyft rule*) if the left-hand sides of its premises are variables. An ntytt rule (resp. ntyft rule) is an *xyntt rule* (resp. *xynft rule*) if the left-hand sides of its positive premises are variables.

Definition 17 A transition rule has *no lookahead* if the variables occurring in the right-hand sides of its positive premises do not occur in the left-hand sides of its premises. A variable occurring in a transition rule is *free* if it does not occur in the source nor in the right-hand sides of the positive premises of this rule. We say that a transition rule is *decent* if it has no lookahead and does not contain free variables.

Each combination of syntactic restrictions on transition rules induces a corresponding syntactic format for TSSs of the same name. For instance, a TSS is in *decent ntyft* format if it contains decent ntyft rules only. We proceed to define further syntactic formats for TSSs.

Definition 18 A TSS is in *ntyft/ntyxt format* if it contains only ntyft and ntyxt rules. A TSS is in *tyft/tyxt format* if it is positive and in ntyft/ntyxt format. A TSS is in *ready simulation format* if it is in ntyft/ntyxt format and its transition rules have no lookahead.

Theorem 2 [17] If a complete standard TSS is in ntyft/ntyxt format, then the bisimulation equivalence that it induces is a congruence.

Proof: In Bol & Groote [10] it is established that for any complete standard TSS in ntyft/ntyxt format that satisfies a condition called *well-foundedness*, the bisimulation equivalence that it induces is a congruence. This extends an earlier result of Groote & Vaandrager [24] for well-founded TSSs in tyft/tyxt format. In [17] it was shown that for every complete standard TSS in ntyft/ntyxt format there exists a complete standard TSS in xynft format that is well-founded, and that induces the same transition relation. From this the theorem follows immediately. □

Theorem 3 If a TSS is in tyft/tyxt format, then the n-nested simulation preorders that it induces for $n \geq 1$ are precongruences.

Proof: Groote & Vaandrager [24] stated that for any well-founded TSS in tyft/tyxt format, the n-nested simulation equivalences that it induces are congruences. They remarked that this result can be established in exactly the same way as the fact that the bisimulation equivalence that a well-founded TSS in tyft/tyxt format induces is a congruence.

We remark that in exactly the same way one can show that the n-nested simulation *preorders* that a well-founded TSS in tyft/tyxt format induces are *pre*congruences. In [17] it was shown that for every TSS in tyft/tyxt format there exists a positive TSS in xynft format that is well-founded, and that induces the same transition relation. From this the theorem follows immediately. □

Theorem 4 If a standard TSS is in ready simulation format, then the ready simulation preorder that it induces is a precongruence.

Definition 19 An occurrence of a variable in an ntytt rule is *propagated* if the occurrence is either in the target, or in the left-hand side of a positive premise whose right-hand side occurs in the target. An occurrence of a variable in an ntytt rule is *polled* if the occurrence is in the left-hand side of a premise that does not have a right-hand side occurring in the target.

Consider for instance the transition rules of Example 7 in Section 9. In the second rule both occurrences of x in the premisses are propagated, i.e. the variable x is propagated twice. In the third rule the variables x_1 and x_2 are polled once each. We can think of a process, represented by a variable in a transition rule, as being *copied* if the variable is propagated more than once. The process is *tested* if the variable is either propagated or polled.

Our precongruence formats for decorated trace preorders operate by keeping track of which variables represent running processes, and which do not. For example, it is semantically reasonable to copy a process before it starts, effectively getting information about all the conjuncts in a $\phi \in \mathbb{O}^{\wedge}$. However, copying a running process would give information about the branching structure of the process, which is incompatible with any form of decorated trace semantics. We introduce a predicate Λ as the basis for determining the Λ-*floating* variables, which represent processes that may be running.

Definition 20 Let Σ be a signature, and Λ a unary predicate on $\{(f,i) \mid 1 \leq i \leq ar(f),\ f \in \Sigma\}$. If $\Lambda(f,i)$, then we say that argument i of f is *liquid*; otherwise it is *frozen*. An occurrence of a variable x in a term $t \in \mathbb{T}(\Sigma)$ is *at a* Λ-*liquid position* if either $t = x$, or $t = f(t_1, \ldots, t_{ar(f)})$ and the occurrence of x is Λ-liquid in t_i for some liquid argument i of f. A variable in an ntytt rule over Σ is Λ-*floating* if either it occurs as the right-hand side of a positive premise, or it occurs exactly once in the source, at a Λ-liquid position.

Note that an occurrence of a variable x in a term $t \in \mathbb{T}(\Sigma)$ is Λ-liquid iff t does not contain a subterm $f(t_1, \ldots, t_{ar(f)})$ such that the occurrence of x is in t_i for a frozen argument i of f.

Definition 21 Let Λ be a unary predicate on arguments of function symbols. A standard ntytt rule is Λ-*ready trace safe* if

- it has no lookahead, and
- each Λ-floating variable has at most one propagated occurrence, which must be at a Λ-liquid position.

The rule is Λ-*readiness safe* if

- it is Λ-ready trace safe, and
- no Λ-floating variable has both propagated and polled occurrences.

The rule is Λ-*failure trace safe* if

- it is Λ-readiness safe, and
- each Λ-floating variable has at most one polled occurrence, which must be at a Λ-liquid position in a positive premise.

The second restriction on "Λ-ready trace safe" guarantees that a running process is never copied, and continued to be marked as running after it has executed. The "Λ-readiness safe" restriction ensures that only at the end of its execution a running process is tested multiple times. The "Λ-failure trace safe" restriction further limits to a positive test on a single action.

Definition 22 A standard TSS is in *ready trace format* if it is in ntyft/ntyxt format and its rules are Λ-ready trace safe with respect to some Λ. A standard TSS is in *readiness format* if it is in ntyft/ntyxt format and its rules are Λ-readiness safe with respect to some Λ. A standard TSS is in *failure trace format* if it is in ntyft/ntyxt format and its rules are Λ-failure trace safe with respect to some Λ.

Note that if a standard TSS P is in ready trace format (resp. readiness format or failure trace format), then there is a smallest predicate Λ_0 such that the rules of P are Λ_0-ready trace safe (resp. Λ_0-readiness safe or Λ_0-failure trace safe). In the context of the ready trace format, for instance, Λ_0 can be defined as the smallest predicate Λ such that for all rules of P each Λ-floating variable is propagated at Λ-liquid positions only. Now P is in ready trace format iff it has no lookahead and in all of its rules each Λ_0-floating variable is propagated at most once. Therefore, in the context of a given standard TSS and a given format, positions can be called *liquid* and variables *floating* without mentioning a specific predicate Λ; in such a case Λ_0 may be assumed.

Theorem 5 If a standard TSS is in ready trace format, then the ready trace preorder that it induces is a precongruence.

Theorem 6 If a standard TSS is in readiness format, then the readiness preorder that it induces is a precongruence.

Theorem 7 If a standard TSS is in failure trace format, then the failure trace and failure preorders that it induces are precongruences.

Sections 5–8 are devoted to the proofs of Theorems 4–7. Section 9 presents a series of counterexamples showing that the syntactic restrictions formulated above are essential for the claimed precongruence results. These counterexamples also help in motivating the definitions above.

For comparison with the literature we point out that a standard TSS is in GSOS format [9] iff it is finite and in decent nxyft format, and each rule has finitely many premises. A standard TSS is in de Simone's format iff it is positive, in decent nxyft format, and its rules are Λ-failure trace safe with Λ the universal predicate (making all arguments of function symbols liquid).

5 Preservation of syntactic restrictions

Later on, in the proofs of the precongruence theorems, we will use transition rules with negative conclusions. For this reason, the ready trace, readiness and failure trace formats need to be extended to non-standard TSSs.

Definition 23 Let Λ be a unary predicate on arguments of function symbols. An ntytt rule with a negative conclusion is Λ-*ready trace safe* or Λ-*readiness safe* if it has no lookahead. The rule is Λ-*failure trace safe* if

- it is Λ-readiness safe, and
- Λ-floating variables are polled only at Λ-liquid positions and only in negative premises.

Now Definition 22 applies to non-standard TSSs as well. Note that for Λ-ready trace and Λ-readiness safety the requirements are the same as in the standard case, for in a rule with a negative conclusion no variable is propagated. In the definition of Λ-failure trace safety, however, rules with positive and negative conclusions are treated differently.

In the remainder of this section we show that the syntactic restrictions of the precongruence formats for TSSs in decent ntyft format are inherited by the ntytt rules irredundantly provable from such TSSs. The restriction to TSSs in decent ntyft format is justified because Propositions 2 and 3 in Section 6.1 will imply that in the proofs of Theorems 4–7 we may, without limitation of generality, restrict attention to TSSs in decent ntyft format. The results of this section will be of use in the forthcoming proofs of the precongruence theorems.

For H a set of literals let $lvar(H)$ denote the set of variables occurring in left-hand sides of literals in H, and $rhs(H)$ the set of right-hand sides of positive literals in H. Note that an ntytt rule $\frac{H}{t \stackrel{a}{\not\rightarrow}}$ [resp. $\frac{H}{t \stackrel{a}{\longrightarrow} t'}$] is decent iff $lvar(H) \subseteq var(t)$ [and $var(t') \subseteq var(t) \cup rhs(H)$].

Lemma 1 Let $P = (\Sigma, R)$ be a TSS in decent ntytt format. Then any ntytt rule irredundantly provable from P is decent.

Proof: We prove a slightly stronger statement, namely that *any* transition rule $\frac{H}{t\stackrel{a}{\nrightarrow}}$ [resp. $\frac{H}{t\stackrel{a}{\longrightarrow}t'}$] irredundantly provable from P satisfies $lvar(H)\subseteq var(t)$ [and $var(t')\subseteq var(t)\cup rhs(H)$]. We apply structural induction with respect to the irredundant proof π of such a rule from P. Let $P\vdash_{\rm irr}\frac{H}{t\stackrel{a}{\longrightarrow}t'}$; the case of a rule with a negative conclusion goes similarly.

Induction basis: Suppose π has only one node. Then $H=\{t\stackrel{a}{\longrightarrow}t'\}$, which implies that t' is a variable. Clearly $\frac{t\stackrel{a}{\longrightarrow}t'}{t\stackrel{a}{\longrightarrow}t'}$ has the required properties.

Induction step: Let $r\in R$ be the decent ntytt rule and ρ be the substitution used at the bottom of π, where r is of the form $\frac{\{v_k\stackrel{c_k}{\longrightarrow}y_k\mid k\in K\}\cup\{w_\ell\stackrel{d_\ell}{\nrightarrow}\mid \ell\in L\}}{u\stackrel{a}{\longrightarrow}u'}$. Then $\rho(u)=t$ and $\rho(u')=t'$. Moreover, transition rules $\frac{H_k}{\rho(v_k)\stackrel{c_k}{\longrightarrow}\rho(y_k)}$ for $k\in K$ and $\frac{H_\ell}{\rho(w_\ell)\stackrel{d_\ell}{\nrightarrow}}$ for $\ell\in L$ are irredundantly provable from P by means of strict subproofs of π, where $H=\bigcup_{k\in K}H_k\cup\bigcup_{\ell\in L}H_\ell$. By induction $lvar(H_k)\subseteq var(\rho(v_k))$ for $k\in K$. As r is decent we have $var(v_k)\subseteq var(u)$, so $lvar(H_k)\subseteq var(\rho(v_k))\subseteq var(\rho(u))=var(t)$ for $k\in K$. Similarly, $lvar(H_\ell)\subseteq var(t)$ for $\ell\in L$, thus $lvar(H)\subseteq var(t)$.

As r is decent we have $var(u')\subseteq var(u)\cup\{y_k\mid k\in K\}$, so

$$var(t')=var(\rho(u'))\subseteq var(t)\cup\bigcup_{k\in K}var(\rho(y_k)).$$

By induction we have $var(\rho(y_k))\subseteq var(\rho(v_k))\cup rhs(H_k)\subseteq var(t)\cup rhs(H_k)$ for $k\in K$. It follows that $var(t')\subseteq var(t)\cup rhs(H)$. $\square$

Lemma 2 Let $P=(\Sigma,R)$ be a TSS in decent ntyft format of which the transition rules are Λ-ready trace safe. Then any ntytt rule $\frac{H}{t\stackrel{a}{\longrightarrow}u}$ irredundantly provable from P is Λ-ready trace safe.

Proof: We apply structural induction with respect to the irredundant proof π of $\frac{H}{t\stackrel{a}{\longrightarrow}u}$ from P.

Induction basis: Suppose π has only one node. Then it must be the case that $H=\{t\stackrel{a}{\longrightarrow}u\}$, which implies that u is a variable. The Λ-floating variables in $\frac{t\stackrel{a}{\longrightarrow}u}{t\stackrel{a}{\longrightarrow}u}$ are u and the variables that occur in Λ-liquid positions in t. Both kinds are propagated only once, in the target and in the left-hand side of the positive premise, respectively. These propagations are at Λ-liquid positions.

Induction step: Let $r\in R$ be the decent ntyft rule and ρ be the substitution used at the bottom of π, where r is of the form $\frac{\{v_k\stackrel{c_k}{\longrightarrow}y_k\mid k\in K\}\cup\{w_\ell\stackrel{d_\ell}{\nrightarrow}\mid \ell\in L\}}{f(x_1,\ldots,x_{ar(f)})\stackrel{a}{\longrightarrow}v}$. Then $f(\rho(x_1),\ldots,\rho(x_{ar(f)}))=t$ and $\rho(v)=u$. Moreover, transition rules $r_k=\frac{H_k}{\rho(v_k)\stackrel{c_k}{\longrightarrow}\rho(y_k)}$ for $k\in K$ and $r'_\ell=\frac{H_\ell}{\rho(w_\ell)\stackrel{d_\ell}{\nrightarrow}}$ for $\ell\in L$ are irredundantly provable from P by means of strict subproofs of π, where $H=\bigcup_{k\in K}H_k\cup\bigcup_{\ell\in L}H_\ell$. As r is decent we have $var(v_k)\subseteq\{x_1,\ldots,x_{ar(f)}\}$, so $var(\rho(v_k))\subseteq\bigcup_{i=1}^{ar(f)}var(\rho(x_i))=var(t)$ for $k\in K$. Since $\frac{H}{t\stackrel{a}{\longrightarrow}u}$ is an ntytt rule, $rhs(H)\cap var(t)=\emptyset$. Hence $rhs(H_k)\cap var(\rho(v_k))=\emptyset$ for $k\in K$. It follows that the rules r_k for $k\in K$ are ntytt rules. The same holds for the rules r'_ℓ for $\ell\in L$. By Lemma 1, the rules r_k and r'_ℓ are decent.

Let K_0 denote $\{k\in K\mid y_k\in var(v)\}$. We make the following observation.

(A) If the right-hand side y of a positive premise in H occurs in u, then there is a $k\in K_0$ such that the premise is in H_k and $y\in var(\rho(y_k))$.

Namely, y does not occur in $\rho(x_i)$ for $i=1,\ldots,ar(f)$. Since $y\in var(\rho(v))$ and, by the decency of r, $var(v)\subseteq\{x_1,\ldots,x_{ar(f)}\}\cup\{y_k\mid k\in K_0\}$, y must occur in $\rho(y_k)$ for some $k\in K_0$. Furthermore,

$var(v_k) \subseteq \{x_1, \ldots, x_{ar(f)}\}$ implies $y \notin var(\rho(v_k))$. So by the decency of $\frac{H_k}{\rho(v_k) \xrightarrow{c_k} \rho(y_k)}$, the positive premise in H with right-hand side y is in H_k.

Let the variable z be Λ-floating in $\frac{H}{t \xrightarrow{a} u}$. We need to prove that z is propagated at most once in this rule, and at a Λ-liquid position. Since z is Λ-floating in $\frac{H}{t \xrightarrow{a} u}$, we can distinguish two cases.

1. Let z occur at a Λ-liquid position in $\rho(x_i)$, for a liquid argument i of f.

   By assumption z occurs only once in t, so z occurs only once in $\rho(x_i)$, and $z \notin var(\rho(x_j))$ for $j \neq i$. We make a second observation.

   (B) If $x_i \notin var(v_k)$ for some $k \in K$, then $z \notin var(H_k)$ and $z \notin var(\rho(y_k))$.

   Namely, as $\frac{H}{t \xrightarrow{a} u}$ is a ntytt rule, $var(t) \cap rhs(H_k) \subseteq var(t) \cap rhs(H) = \emptyset$, hence $z \notin rhs(H_k)$. Furthermore, $var(v_k) \subseteq \{x_1, \ldots, x_{ar(f)}\}$, $x_i \notin var(v_k)$ and z does not occur in $\rho(x_j)$ for $j \neq i$, so $z \notin var(\rho(v_k))$. Hence, by the decency of $\frac{H_k}{\rho(v_k) \xrightarrow{c_k} \rho(y_k)}$, z occurs neither in H_k nor in $\rho(y_k)$. Once more we distinguish two cases.

1.1 Let $x_i \notin var(v_k)$ for all $k \in K_0$.

   By (B), $z \notin var(\rho(y_k))$ for $k \in K_0$. Furthermore, z does not occur in $\rho(x_j)$ for $j \neq i$, and z occurs only once in $\rho(x_i)$, at a Λ-liquid position. Finally, since r is Λ-ready trace safe, and i is a liquid argument of f, x_i occurs at most once in v, and at a Λ-liquid position. As $var(v) \subseteq \{x_1, \ldots, x_{ar(f)}\} \cup \{y_k \mid k \in K_0\}$, it follows that z occurs at most once in $\rho(v) = u$, and at a Λ-liquid position.

   By (B), $z \notin var(H_k)$ for $k \in K_0$. Moreover, by (A), right-hand sides of positive premises in H_k for $k \in K \backslash K_0$ and H_ℓ for $\ell \in L$ do not occur in u. Hence, there are no propagated occurrences of z in left-hand sides of positive premises in $\frac{H}{t \xrightarrow{a} u}$.

1.2 Let $x_i \in var(v_{k'})$ for some $k' \in K_0$.

   r is Λ-ready trace safe, and i is a liquid argument of f, so $x_i \notin var(v)$ and $x_i \notin var(v_k)$ for $k \in K_0 \backslash \{k'\}$. Moreover, x_i occurs only once in $v_{k'}$, at a Λ-liquid position. Since z occurs only once in $\rho(x_i)$, at a Λ-liquid position, and z does not occur in $\rho(x_j)$ for $j \neq i$, it follows that z occurs only once in $\rho(v_{k'})$, at a Λ-liquid position. Thus, by induction z is propagated at most once in $\frac{H_{k'}}{\rho(v_{k'}) \xrightarrow{c_{k'}} \rho(y_{k'})}$, and at a Λ-liquid position. In particular, z occurs at most once in $\rho(y_{k'})$, and at a Λ-liquid position. Moreover, as r is Λ-ready trace safe, $y_{k'}$ occurs at most once in v, and at a Λ-liquid position. By (B), $x_i \notin var(v_k)$ implies that $z \notin var(\rho(y_k))$ for $k \in K_0 \backslash \{k'\}$. Furthermore, $x_i \notin var(v)$ and $z \notin var(\rho(x_j))$ for $j \neq i$. As $var(v) \subseteq \{x_1, \ldots, x_{ar(f)}\} \cup \{y_k \mid k \in K_0\}$, it follows that z occurs at most once in $\rho(v) = u$, and at a Λ-liquid position.

   If there are no propagated occurrences of z in left-hand sides of positive premises of $\frac{H}{t \xrightarrow{a} u}$, then we are done. So suppose there is a positive premise $s \xrightarrow{c} y$ in H where z occurs in s and $y \in var(u)$. We proceed to prove that the occurrence of z in s is Λ-liquid and that this is the only propagated occurrence of z in $\frac{H}{t \xrightarrow{a} u}$.

   By (B), $x_i \notin var(v_k)$ implies $z \notin var(H_k)$ for $k \in K_0 \backslash \{k'\}$. On the other hand, since $y \in var(u)$, (A) yields that $s \xrightarrow{c} y \in H_k$ and $y \in var(\rho(y_k))$ for some $k \in K_0$. Hence,

$s \xrightarrow{c} y \in H_{k'}$ and $y \in var(\rho(y_{k'}))$. By induction, z is propagated at most once in $\frac{H_{k'}}{\rho(v_{k'}) \xrightarrow{c_{k'}} \rho(y_{k'})}$, and at a Λ-liquid position. So the occurrence of z in s is Λ-liquid. Moreover, it is the only propagated occurrence of z in the left-hand sides of the positive premises in $H_{k'}$, and $z \notin var(\rho(y_{k'}))$.

$x_i \notin var(v)$, z occurs neither in $\rho(x_j)$ for $j \neq i$ nor in $\rho(y_k)$ for $k \in K_0 \backslash \{k'\}$, and $z \notin var(\rho(y_{k'}))$. As $var(v) \subseteq \{x_1, \ldots, x_{ar(f)}\} \cup \{y_k \mid k \in K_0\}$, it follows that z does not occur in $\rho(v) = u$.

According to (A), right-hand sides of positive premises in H_k for $k \in K \backslash K_0$ and in H_ℓ for $\ell \in L$ do not occur in u. Furthermore, by (B), $x_i \notin var(v_k)$ implies $z \notin var(H_k)$ for $k \in K_0 \backslash \{k'\}$. Finally, for positive premises $s' \xrightarrow{d} y'$ in $H_{k'}$ with $z \in var(s')$ and $y' \neq y$ we have $y' \notin var(\rho(y_{k'}))$, and so by (A) $y' \notin var(u)$. Hence, the occurrence of z in the left-hand side of $s \xrightarrow{c} y$ is the only propagated occurrence of z in $\frac{H}{t \xrightarrow{a} u}$.

2. Let z be the right-hand side of a positive premise in H.

   Since $\frac{H}{t \xrightarrow{a} u}$ is decent, z does not occur in left-hand sides of premises in H. It remains to prove that z occurs at most once in u, and at a Λ-liquid position.

   Since $\frac{H}{t \xrightarrow{a} u}$ is an ntytt rule, z does not occur in $\rho(x_i)$ for $i = 1, \ldots, ar(f)$. As r is decent, $var(v_k) \subseteq \{x_1, \ldots, x_{ar(f)}\}$ for $k \in K$. Hence z does not occur in $\rho(v_k)$ for $k \in K$.

   According to (A), right-hand sides of positive premises in H_k for $k \in K \backslash K_0$ and in H_ℓ for $\ell \in L$ do not occur in u. So we may assume that z is the right-hand side of a positive premise in $H_{k'}$ for some $k' \in K_0$. Then clearly z does not occur in H_k for $k \in K_0 \backslash \{k'\}$. So by the decency of $\frac{H_k}{\rho(v_k) \xrightarrow{c_k} \rho(y_k)}$, $z \notin var(\rho(y_k))$ for $k \in K_0 \backslash \{k'\}$. By induction, z occurs at most once in $\rho(y_{k'})$, and at a Λ-liquid position. Since r is Λ-ready trace safe, $y_{k'}$ occurs at most once in v, and at a Λ-liquid position. As $var(v) \subseteq \{x_1, \ldots, x_{ar(f)}\} \cup \{y_k \mid k \in K_0\}$, it follows that z occurs at most once in $\rho(v) = u$, and at a Λ-liquid position.

We conclude that z is propagated at most once in $\frac{H}{t \xrightarrow{a} u}$, and at a Λ-liquid position. □

Lemma 3 Let $P = (\Sigma, R)$ be a TSS in decent ntyft format of which the transition rules are Λ-readiness safe. Then any ntytt rule $\frac{H}{t \xrightarrow{a} u}$ irredundantly provable from P is Λ-readiness safe.

Proof: Let the variable z be Λ-floating in $\frac{H}{t \xrightarrow{a} u}$. We need to prove that z is not both propagated and polled in this rule. In case z is the right-hand side of a positive premise, it is not polled, since the rule is decent, by Lemma 1. So assume z occurs exactly once in t, at a Λ-liquid position. We apply structural induction with respect to the irredundant proof π of $\frac{H}{t \xrightarrow{a} u}$ from P.

Induction basis: Suppose π has only one node. Then it must be the case that $H = \{t \xrightarrow{a} u\}$, which implies that u is a variable. There are no polled variables in $\frac{t \xrightarrow{a} u}{t \xrightarrow{a} u}$.

Induction step: Let $r \in R$ be the decent ntyft rule and ρ be the substitution used at the bottom of π, where r is of the form $\frac{\{v_k \xrightarrow{c_k} y_k \mid k \in K\} \cup \{w_\ell \stackrel{d_\ell}{\nrightarrow} \mid \ell \in L\}}{f(x_1, \ldots, x_{ar(f)}) \xrightarrow{a} v}$. Then $f(\rho(x_1), \ldots, \rho(x_{ar(f)})) = t$ and $\rho(v) = u$. Moreover, transition rules $\frac{H_k}{\rho(v_k) \xrightarrow{c_k} \rho(y_k)}$ for $k \in K$ and $\frac{H_\ell}{\rho(w_\ell) \stackrel{d_\ell}{\nrightarrow}}$ for $\ell \in L$ are irredundantly provable from P by means of strict subproofs of π, where $H = \bigcup_{k \in K} H_k \cup \bigcup_{\ell \in L} H_\ell$. As in the proof of Lemma 2 it can be shown that these rules are ntytt, and by Lemma 1 they are decent.

By assumption, z occurs exactly once in $\rho(x_i)$ for some liquid argument i of f, at a Λ-liquid position, and $z \notin var(\rho(x_j))$ for $j \neq i$. Let K_0 denote $\{k \in K \mid y_k \in var(v)\}$. We recall two observations from the proof of Lemma 2.

(A) If the right-hand side y of a positive premise in H occurs in u, then there is a $k \in K_0$ such that the premise is in H_k and $y \in var(\rho(y_k))$.

(B) If $x_i \notin var(v_k)$ for some $k \in K$, then $z \notin var(H_k)$ and $z \notin var(\rho(y_k))$. Likewise, if $x_i \notin var(w_\ell)$ for some $\ell \in L$, then $z \notin var(H_\ell)$.

Suppose z is polled in $\frac{H}{t \xrightarrow{a} u}$, so that we can distinguish the following two cases. We need to prove that z is not propagated in $\frac{H}{t \xrightarrow{a} u}$.

1. Let z be polled in the left-hand side of a premise in $H_{k'}$ for some $k' \in K \backslash K_0$ or in $H_{\ell'}$ for some $\ell' \in L$.

   By (B), $z \in var(H_{k'})$ or $z \in var(H_{\ell'})$ implies $x_i \in var(v_{k'})$ or $x_i \in var(w_{\ell'})$, respectively, so x_i is polled in r. Since r is Λ-readiness safe, and i is a liquid argument of f, it follows that x_i is not propagated in r. Hence, x_i occurs neither in v nor in v_k for $k \in K_0$. By (B), $x_i \notin var(v_k)$ implies $z \notin var(H_k)$ for $k \in K_0$. So, in view of (A), occurrences of z in left-hand sides of positive premises in H are not propagated in $\frac{H}{t \xrightarrow{a} u}$. It remains to prove that $z \notin var(u)$.

   By (B), $x_i \notin var(v_k)$ implies that $z \notin var(\rho(y_k))$ for $k \in K_0$. Moreover, $x_i \notin var(v)$ and z does not occur in $\rho(x_j)$ for $j \neq i$. As $var(v) \subseteq \{x_1, \ldots, x_{ar(f)}\} \cup \{y_k \mid k \in K_0\}$, it follows that z does not occur in $\rho(v) = u$.

2. Let z be polled in the left-hand side of a premise in $H_{k'}$ for some $k' \in K_0$.

   Then z occurs in the left-hand side of a premise in $H_{k'}$ of the form $s \stackrel{c}{\not\longrightarrow}$ or $s \xrightarrow{c} y$ with $y \notin var(u)$. In the latter case, $y_{k'} \in var(v)$ implies that $y \notin var(\rho(y_{k'}))$. So in both cases, z is polled in $\frac{H_{k'}}{\rho(v_{k'}) \xrightarrow{c_{k'}} \rho(y_{k'})}$.

   By (B), $z \in var(H_{k'})$ implies $x_i \in var(v_{k'})$. Since r is Λ-readiness safe, and i is a liquid argument of f, x_i occurs neither in v nor in v_k for $k \in K_0 \backslash \{k'\}$. Moreover, x_i occurs only once in $v_{k'}$, at a Λ-liquid position. Since z occurs exactly once in $\rho(x_i)$, at a Λ-liquid position, and not at all in $\rho(x_j)$ for $j \neq i$, it follows that z occurs exactly once in $\rho(v_{k'})$, at a Λ-liquid position. Since z is polled in $\frac{H_{k'}}{\rho(v_{k'}) \xrightarrow{c_{k'}} \rho(y_{k'})}$, by induction it is not propagated in this rule. So if $(s \xrightarrow{c} y) \in H_{k'}$ and $z \in var(s)$ then y does not occur in $\rho(y_{k'})$, and hence, by (A), not in u. Thus, occurrences of z in left-hand sides of positive premises in $H_{k'}$ are not propagated in $\frac{H}{t \xrightarrow{a} u}$. Furthermore, by (B), $x_i \notin var(v_k)$ implies $x_i \notin var(H_k)$ for $k \in K_0 \backslash \{k'\}$. So, in view of (A), occurrences of z in left-hand sides of positive premises in H are not propagated in $\frac{H}{t \xrightarrow{a} u}$. It remains to prove that $z \notin var(u)$.

   By (B), $x_i \notin var(v_k)$ implies that $z \notin var(\rho(y_k))$ for $k \in K_0 \backslash \{k'\}$. Moreover, since z is not propagated in $\frac{H_{k'}}{\rho(v_{k'}) \xrightarrow{c_{k'}} \rho(y_{k'})}$, $z \notin var(\rho(y_{k'}))$. Finally, $x_i \notin var(v)$ and z does not occur in $\rho(x_j)$ for $j \neq i$. As $var(v) \subseteq \{x_1, \ldots, x_{ar(f)}\} \cup \{y_k \mid k \in K_0\}$, it follows that z does not occur in $\rho(v) = u$.

We conclude that z is not propagated in $\frac{H}{t \xrightarrow{a} u}$. $\square$

Lemma 4 Let $P = (\Sigma, R)$ be a TSS in decent ntyft format of which the transition rules are Λ-failure trace safe. Then any ntytt rule $\frac{H}{t \xrightarrow{a} u}$ irredundantly provable from P is Λ-failure trace safe.

Proof: Let the variable z be Λ-floating in $\frac{H}{t \xrightarrow{a} u}$. We need to prove that z is polled at most once in this rule, at a Λ-liquid position in a positive premise. In case z is the right-hand side of a positive premise, it is not polled, since the rule is decent, by Lemma 1. So assume z occurs exactly once in t, at a Λ-liquid position. We apply structural induction with respect to the irredundant proof π of $\frac{H}{t \xrightarrow{a} u}$ from P.

Induction basis: Suppose π has only one node. Then it must be the case that $H = \{t \xrightarrow{a} u\}$, where u is a variable. There are no polled variables in $\frac{t \xrightarrow{a} u}{t \xrightarrow{a} u}$.

Induction step: Let $r \in R$ be the decent ntyft rule and ρ be the substitution used at the bottom of π, where r is of the form $\frac{\{v_k \xrightarrow{c_k} y_k \mid k \in K\} \cup \{w_\ell \stackrel{d_\ell}{\not\longrightarrow} \mid \ell \in L\}}{f(x_1, \ldots, x_{ar(f)}) \xrightarrow{a} v}$. Then $f(\rho(x_1), \ldots, \rho(x_{ar(f)})) = t$ and $\rho(v) = u$. Moreover, transition rules $\frac{H_k}{\rho(v_k) \xrightarrow{c_k} \rho(y_k)}$ for $k \in K$ and $\frac{H_\ell}{\rho(w_\ell) \stackrel{d_\ell}{\not\longrightarrow}}$ for $\ell \in L$ are irredundantly provable from P by means of strict subproofs of π, where $H = \bigcup_{k \in K} H_k \cup \bigcup_{\ell \in L} H_\ell$. As in the proof of Lemma 2 it can be shown that these rules are ntytt, and by Lemma 1 they are decent.

By assumption, z occurs exactly once in $\rho(x_i)$ for some liquid argument i of f, at a Λ-liquid position, and $z \notin var(\rho(x_j))$ for $j \neq i$. We need to prove that z is polled at most once in $\frac{H}{t \xrightarrow{a} u}$, at a Λ-liquid position in a positive premise. We recall an observation from the proofs of Lemmas 2 and 3.

(B) If $x_i \notin var(v_k)$ for some $k \in K$, then $z \notin var(H_k)$. Likewise, if $x_i \notin var(w_\ell)$ for some $\ell \in L$, then $z \notin var(H_\ell)$.

r is Λ-failure trace safe, and i is a liquid argument of f. So x_i is polled and propagated at most once in r in total, at a Λ-liquid position and not in a negative premise. In particular, $x_i \notin var(w_\ell)$ for $\ell \in L$, so in view of (B), $z \notin var(H_\ell)$ for $\ell \in L$. Suppose $z \in var(H_{k'})$ for some $k' \in K$. By (B), $x_i \in var(v_{k'})$. Then $x_i \notin var(v_k)$ for $k \in K \backslash \{k'\}$, so in view of (B), $z \notin var(H_k)$ for $k \in K \backslash \{k'\}$. Furthermore, x_i occurs only once in $v_{k'}$, at a Λ-liquid position. Since z does not occur in $\rho(x_j)$ for $j \neq i$ and exactly once in $\rho(x_i)$, at a Λ-liquid position, it follows that z occurs exactly once in $\rho(v_{k'})$, at a Λ-liquid position. By induction together with Lemmas 2 and 3, z occurs at most once in $H_{k'}$, at a Λ-liquid position in a positive premise. We conclude that z occurs at most once in H, at a Λ-liquid position in a positive premise. $\square$

Lemma 5 Let $P = (\Sigma, R)$ be a TSS in decent ntyft format of which the transition rules are Λ-failure trace safe. Then any ntytt rule $\frac{H}{t \stackrel{a}{\not\longrightarrow}}$ irredundantly provable from P is Λ-failure trace safe.

Proof: With structural induction w.r.t. the irredundant proof π of $\frac{H}{t \stackrel{a}{\not\longrightarrow}}$ from P we establish:

Let $\frac{H}{t \stackrel{a}{\not\longrightarrow}}$ be an ntytt rule that is irredundantly provable from P. If all occurrences of the variable z in t are at Λ-liquid positions, then z is polled only at Λ-liquid positions and only in negative premises of this rule.

Together with Lemma 1 this immediately yields the desired result.

Induction basis: Suppose π has only one node. Then it must be the case that $H = \{t \stackrel{a}{\not\longrightarrow}\}$. There are no positive premises in $\frac{t \stackrel{a}{\not\longrightarrow}}{t \stackrel{a}{\not\longrightarrow}}$. Moreover, z is polled only at Λ-liquid positions.

Induction step: Let $r \in R$ be the decent ntyft rule and ρ be the substitution used at the bottom of π, where r is of the form $\frac{\{v_k \xrightarrow{c_k} y_k | k\in K\} \cup \{w_\ell \stackrel{d_\ell}{\not\rightarrow} | \ell\in L\}}{f(x_1,\ldots,x_{ar(f)}) \stackrel{a}{\not\rightarrow}}$. Then $f(\rho(x_1),\ldots,\rho(x_{ar(f)})) = t$. Moreover, transition rules $\frac{H_k}{\rho(v_k)\xrightarrow{c_k}\rho(y_k)}$ for $k \in K$ and $\frac{H_\ell}{\rho(w_\ell)\stackrel{d_\ell}{\not\rightarrow}}$ for $\ell \in L$ are irredundantly provable from P by means of strict subproofs of π, where $H = \bigcup_{k\in K} H_k \cup \bigcup_{\ell\in L} H_\ell$. As in the proof of Lemma 2 it can be shown that these rules are ntytt, and by Lemma 1 they are decent.

r is Λ-failure trace safe, so variables x_i for liquid arguments i of f do not occur in the v_k for $k \in K$, and only at Λ-liquid positions in the w_ℓ for $\ell \in L$. By assumption z only occurs in $\rho(x_i)$ for liquid arguments i of f, at Λ-liquid positions, so z does not occur in the $\rho(v_k)$ for $k \in K$, and only at Λ-liquid positions in the $\rho(w_\ell)$ for $\ell \in L$. The decency of $\frac{H_k}{\rho(v_k)\xrightarrow{c_k}\rho(y_k)}$ implies that z does not occur in the left-hand sides of the premises of H_k for $k \in K$. Moreover, by induction z does not occur in the positive premises and only at Λ-liquid positions in the negative premises in H_ℓ for $\ell \in L$. Hence, z is polled only at Λ-liquid positions and only in negative premises in $\frac{H}{t \stackrel{a}{\not\rightarrow}}$. $\square$

6 Reducing well-supported proofs to standard proofs

Theorems 4–7 deal with preorders induced by standard TSSs through the notion of well-founded provability. In this section it is shown that without loss of generality we may use the classical notion of provability (of Definition 10) instead. To this end we show that for any given standard TSS $P = (\Sigma, R)$ in ready simulation format (i.e. in ntyft/ntyxt format without lookahead), there exists a TSS $P^+ = (\Sigma, R^+)$ in decent ntyft format such that $P \vdash_{ws} \alpha \Leftrightarrow P^+ \vdash \alpha$. Moreover, the relevant formats are preserved under the translation of R into R^+. However, in general P^+ will not be a standard TSS. It is for this reason that rules with a negative conclusion have been introduced in Definition 9, and that the precongruence formats were extended to non-standard TSSs in Definition 23.

The conversion from R to R^+ will be performed in three steps. In Section 6.1 we show that for any standard TSS $P = (\Sigma, R)$ in ready simulation format there exists a standard TSS $P' = (\Sigma, R')$ in decent xynft format with the same class of ws-provable literals. Moreover, if P is in ready trace, readiness, resp. failure trace format, then so is P'. In Section 6.2 we show that for standard TSSs in decent xynft format the notion of well-supported provability coincides with a simpler notion of *supported provability*. Finally, in Section 6.3 we show that for any standard TSS $P' = (\Sigma, R')$ in decent xynft format there exists a TSS $P^+ = (\Sigma, R^+)$ in decent ntyft format such that supported provability from P' coincides with classical provability from P^+. Also this translation preserves the ready trace, readiness and failure trace formats. Together, this yields the desired result.

6.1 Reducing ntyft/ntyxt rules without lookahead to decent xynft rules

We show that for every standard TSS P in ready simulation format there exists a standard TSS P' in decent xynft format, such that

(i) $P \vdash_{ws} \alpha \Leftrightarrow P' \vdash_{ws} \alpha$ for any closed literal α,

(ii) if P is in ready trace format, then so is P',

(iii) if P is in readiness format, then so is P',

(iv) and if P is in failure trace format, then so is P'.

The following proposition helps in establishing the first requirement above.

Proposition 1 [21] Let $P = (\Sigma, R)$ and $P' = (\Sigma, R')$ be standard TSSs with the property that $P \vdash \frac{N}{t \xrightarrow{a} t'} \Leftrightarrow P' \vdash \frac{N}{t \xrightarrow{a} t'}$ for any closed transition rule $\frac{N}{t \xrightarrow{a} t'}$ with only negative premises. Then $P \vdash_{ws} \alpha \Leftrightarrow P' \vdash_{ws} \alpha$ for any closed literal α. $\Box$

Proof: By symmetry it suffices to establish "$\Rightarrow$". This goes with structural induction on proofs. Let π be a well-supported proof of α from P. In case α is negative, it must be the case that for every set N of negative closed literals such that $P \vdash \frac{N}{\gamma}$ for γ a closed literal denying α, a strict subproof of π proves a literal δ that denies one in N. Hence for every set N of negative closed literals such that $P' \vdash \frac{N}{\gamma}$ for γ a closed literal denying α, a strict subproof of π proves a literal δ that denies one in N. By induction $P' \vdash_{ws} \delta$ for all those literals δ. It follows that $P' \vdash_{ws} \alpha$.

In case α is positive, just take the bottom portion of π obtained by deleting all nodes above nodes that are labelled with a negative literal. That portion is a proof of a closed transition rule $\frac{N}{\alpha}$ from P in which N is a set of closed negative literals. By assumption, $P' \vdash \frac{N}{\alpha}$. Moreover, $P \vdash_{ws} \gamma$ for all $\gamma \in N$, by strict subproofs of π. Thus by induction $P' \vdash_{ws} \gamma$ for all $\gamma \in N$. By pasting the proofs of γ from P' for $\gamma \in N$ on top of the proof of $\frac{N}{\alpha}$ from P', a well-supported proof of α from P' is obtained. $\Box$

As a first step in the reduction process we show that we can refrain from ntyxt rules.

Proposition 2 For each standard TSS $P = (\Sigma, R)$ in ready simulation format there exists a standard TSS $P' = (\Sigma, R')$ in ntyft format without lookahead such that the requirements (i)-(iv) above are met.

Proof: Replace each ntyxt rule r in R by a collection of ntyft rules $\{r_f \mid f \in \Sigma\}$, where each r_f is obtained by substituting $f(x_1, \ldots, x_n)$ for the variable x that constitutes the source of r, with $x_1, \ldots, x_n$ variables that do not yet occur in r. Let R' denote the collection of ntyft rules that is thus obtained. Note that if a closed transition rule is provable from a TSS, it has a closed proof. Moreover, each closed proof from P of a closed transition rule is a proof from P' of the same transition rule, and vice versa. Hence, by Proposition 1, $P \vdash_{ws} \alpha \Leftrightarrow P' \vdash_{ws} \alpha$ for any closed literal α. As the rules in R have no lookahead, neither have the rules in R'. In order to check requirements (ii)–(iv) note that the variable x_i is propagated (resp. polled) in r_f exactly when and where x is propagated (resp. polled) in r. Moreover, if x_i is Λ-floating in r_f, an occurrence of x_i in r_f is Λ-liquid iff the corresponding occurrence of x in r is. $\Box$

Next we show that we can restrict attention to *decent* ntyft rules, i.e. we can assume that none of the rules has free variables.

Proposition 3 For each standard TSS $P = (\Sigma, R)$ in ntyft format without lookahead there exists a standard TSS $P' = (\Sigma, R')$ in decent ntyft format such that the requirements (i)-(iv) above are met.

Proof: Replace every rule with free variables by a set of new rules. The new rules are obtained by applying every possible substitution of closed terms for the free variables in the old rule. Now every closed proof from P of a closed transition rule is a proof from P' of the same transition rule, and vice versa. Hence, by Proposition 1, $P \vdash_{ws} \alpha \Leftrightarrow P' \vdash_{ws} \alpha$ for any closed literal α. By construction, the rules in R' have no free variables. As the rules in R have no lookahead, neither have the rules in R'; hence P' is in decent ntyft format. The requirements (ii)–(iv) hold trivially. $\Box$

Finally we have to show that decent ntyft rules can be reduced to decent xynft rules.

Lemma 6 If $A \neq \emptyset$, then the set of all literals over a signature Σ is equally large as the set V of variables.

Proof: Recall from Section 3 that $|\Sigma| \leq |V|$ and $|A| \leq |V|$. Let the *size* of a term be the largest number of nested function symbols in it. In case $\Sigma = \emptyset$ we have $\mathbb{T}(\Sigma) = V$, so $|\mathbb{T}(\Sigma)| = |V|$. Now suppose $\Sigma \neq \emptyset$. With induction to $n \in \mathbb{N}$ it follows that there are $|V|$ terms of size n:

Induction basis: Terms of size 0 are variables. The cardinality of the set of those terms is $|V|$.

Induction step: Suppose the set of all terms of size n has cardinality $|V|$. Then the number of terms of size $n+1$ with leading function symbol f is $|V|^{ar(f)} = |V|$, using that $|V|$ is infinite. Thus the number of terms of size $n+1$ is $|\Sigma| \times |V| = |V|$, using that $|\Sigma| \leq |V|$.

Hence, even if $\Sigma \neq \emptyset$ we have $|\mathbb{T}(\Sigma)| = \Sigma_{i=1}^{\infty}|V| = |V|$. So the set of literals over Σ has cardinality $|V| \times |A| \times |V| + |V| \times |A| = |V|$, using that $0 \neq |A| \leq |V|$. $\square$

Next we need a lemma that is very similar to the forthcoming Proposition 8 in Section 7. There, some intuitive explanation can be found as well.

Lemma 7 Let $P = (\Sigma, R)$ be a standard TSS in decent ntyft format. If $P \vdash \frac{N}{\sigma(t) \xrightarrow{a} t'}$ with N a set of negative literals, then there are a decent xyntt rule $\frac{H}{t \xrightarrow{a} u}$ and a substitution σ' with $P \vdash_{\mathrm{irr}} \frac{H}{t \xrightarrow{a} u}$, $P \vdash \frac{N}{\sigma'(H)}$, $\sigma'(t) = \sigma(t)$ and $\sigma'(u) = t'$.

Proof: First, suppose t is a variable. By default, the decent xyntt rule $\frac{t \xrightarrow{a} y}{t \xrightarrow{a} y}$ is irredundantly provable from P. Let σ' be a substitution with $\sigma'(t) = \sigma(t)$ and $\sigma'(y) = t'$. Clearly, $P \vdash \frac{N}{\sigma'(t \xrightarrow{a} y)}$.

Next, suppose $t = f(t_1, \ldots, t_{ar(f)})$. We apply structural induction to a proof π of the transition rule $\frac{N}{\sigma(t) \xrightarrow{a} t'}$ from P. Let $r \in R$ be the decent ntyft rule and ρ be the substitution used at the bottom of π, where r is of the form $\frac{\{v_k \xrightarrow{c_k} y_k | k \in K\} \cup \{w_\ell \overset{d_\ell}{\not\rightarrow} | \ell \in L\}}{f(x_1, \ldots, x_{ar(f)}) \xrightarrow{a} v}$. Then $\rho(x_i) = \sigma(t_i)$ for $i = 1, \ldots, ar(f)$, $\rho(v) = t'$, $(\rho(w_\ell) \overset{d_\ell}{\not\rightarrow}) \in N$ for $\ell \in L$, and $\frac{N}{\rho(v_k) \xrightarrow{c_k} \rho(y_k)}$ for $k \in K$ are provable from P by means of strict subproofs of π. Since r is decent, $var(v_k)$ for $k \in K$ and $var(w_\ell)$ for $\ell \in L$ are included in $\{x_1, \ldots, x_{ar(f)}\}$. Let ρ_0 be a substitution with $\rho_0(x_i) = t_i$ for $i = 1, \ldots, ar(f)$. As $\rho(x_i) = \sigma(t_i) = \sigma(\rho_0(x_i))$ for $i = 1, \ldots, ar(f)$, we have $\rho(v_k) = \sigma(\rho_0(v_k))$ for $k \in K$ and $\rho(w_\ell) = \sigma(\rho_0(w_\ell))$ for $\ell \in L$. So $\frac{N}{\sigma(\rho_0(v_k)) \xrightarrow{c_k} \rho(y_k)}$ for $k \in K$ are provable from P by means of strict subproofs of π. According to the induction hypothesis, for $k \in K$ there are a decent xyntt rule $\frac{H_k}{\rho_0(v_k) \xrightarrow{c_k} u_k}$ and a substitution σ'_k with $P \vdash_{\mathrm{irr}} \frac{H_k}{\rho_0(v_k) \xrightarrow{c_k} u_k}$, $P \vdash \frac{N}{\sigma'_k(H_k)}$, $\sigma'_k(\rho_0(v_k)) = \sigma(\rho_0(v_k))$ and $\sigma'_k(u_k) = \rho(y_k)$. By Lemma 6, we can choose the sets of variables in the right-hand sides of the positive premises in the H_k (for $k \in K$) pairwise disjoint, and disjoint from $var(t)$. This allows us to define a substitution σ' with:

- $\sigma'(z) = \sigma(z)$ for $z \in var(t)$;
- $\sigma'(z) = \sigma'_k(z)$ for right-hand sides z of positive premises in H_k for $k \in K$.

Let

$$H \quad = \quad \bigcup_{k \in K} H_k \;\cup\; \{\rho_0(w_\ell) \overset{d_\ell}{\not\rightarrow} |\ \ell \in L\} \ .$$

Moreover, let ρ_1 be a substitution with $\rho_1(x_i) = t_i$ for $i = 1, \ldots, ar(f)$ and $\rho_1(y_k) = u_k$ for $k \in K$. We verify that the rule $\frac{H}{t \xrightarrow{a} \rho_1(v)}$ together with the substitution σ' satisfy the desired properties.

As $var(v_k) \subseteq \{x_1, \ldots, x_{ar(f)}\}$, it follows that $var(\rho_0(v_k)) \subseteq var(t)$. Since σ' and σ agree on $var(t)$, $\sigma'(\rho_0(v_k)) = \sigma(\rho_0(v_k)) = \sigma'_k(\rho_0(v_k))$ for $k \in K$. Thus, by the decency of $\frac{H_k}{\rho_0(v_k) \xrightarrow{c_k} u_k}$, σ' and σ'_k agree on all variables occurring in this rule for $k \in K$.

1. ρ_1 and ρ_0 agree on $var(v_k) \subseteq \{x_1, \ldots, x_{ar(f)}\}$, so $\rho_1(v_k) = \rho_0(v_k)$ for $k \in K$. Likewise, $\rho_1(w_\ell) = \rho_0(w_\ell)$ for $\ell \in L$. Since $P \vdash_{\rm irr} r$, we have $P \vdash_{\rm irr} \rho_1(r) = \frac{\{\rho_0(v_k) \xrightarrow{c_k} u_k | k \in K\} \cup \{\rho_0(w_\ell) \stackrel{d_\ell}{\not\longrightarrow} | \ell \in L\}}{t \xrightarrow{a} \rho_1(v)}$. Furthermore, $P \vdash_{\rm irr} \frac{H_k}{\rho_0(v_k) \xrightarrow{c_k} u_k}$ for $k \in K$. As $H = \bigcup_{k \in K} H_k \cup \{\rho_0(w_\ell) \stackrel{d_\ell}{\not\longrightarrow} | \ \ell \in L\}$, it follows that $P \vdash_{\rm irr} \frac{H}{t \xrightarrow{a} \rho_1(v)}$.

2. The right-hand sides of the positive premises in any H_k are distinct variables. By construction, these sets of variables (one for every $k \in K$) are pairwise disjoint, and disjoint from $var(t)$. Hence $\frac{H}{t \xrightarrow{a} \rho_1(v)}$ is an ntytt rule. Since the positive premises in H originate from H_k (for $k \in K$), their left-hand sides are variables. This makes the rule an xyntt rule. The rule is decent by Lemma 1.

3. σ' agrees with σ'_k on variables in H_k for $k \in K$, and with σ on variables in $\rho_0(w_\ell)$ for $\ell \in L$. Since $P \vdash \frac{N}{\sigma'_k(H_k)}$ for $k \in K$, and $(\sigma(\rho_0(w_\ell)) \stackrel{d_\ell}{\not\longrightarrow}) \in N$ for $\ell \in L$ (using that $\sigma(\rho_0(w_\ell)) = \rho(w_\ell)$), we conclude that $P \vdash \frac{N}{\sigma'(H)}$.

4. Since σ' and σ agree on $var(t)$, $\sigma'(t) = \sigma(t)$.

5. $\sigma'(\rho_1(x_i)) = \sigma'(t_i) = \sigma(t_i) = \rho(x_i)$ for $i = 1, \ldots, ar(f)$. Moreover, since σ' and σ'_k agree on $var(u_k)$, $\sigma'(\rho_1(y_k)) = \sigma'(u_k) = \sigma'_k(u_k) = \rho(y_k)$ for $k \in K$. As $var(v) \subseteq \{x_1, \ldots, x_{ar(f)}\} \cup \{y_k \mid k \in K\}$, it follows that $\sigma'(\rho_1(v)) = \rho(v) = t'$. □

Lemma 8 If all the rules in a TSS Q are provable from a TSS P, then all the rules that are provable from Q are also provable from P.

Proof: Straightforward and left to the reader. □

Proposition 4 For each standard TSS $P = (\Sigma, R)$ in decent ntyft format there exists a standard TSS $P' = (\Sigma, R')$ in decent xynft format such that the requirements (i)–(iv) above are met.

Proof: Let R' consist of all decent xynft rules irredundantly provable from P. In order to establish (i) we show that $P' \vdash \frac{N}{t \xrightarrow{a} t'} \Leftrightarrow P \vdash \frac{N}{t \xrightarrow{a} t'}$ for all literals $t \xrightarrow{a} t'$ with t closed and all sets of negative literals N. By Lemma 8 we have $P' \vdash \frac{N}{t \xrightarrow{a} t'} \Rightarrow P \vdash \frac{N}{t \xrightarrow{a} t'}$. For the other direction we use structural induction on t. So assume $P \vdash \frac{N}{f(t_1,\ldots,t_{ar(f)}) \xrightarrow{a} t'}$. By Lemma 7 there are a decent xynft rule $r = \frac{H}{f(x_1,\ldots,x_{ar(f)}) \xrightarrow{a} u}$ and a substitution σ' with $P \vdash_{\rm irr} r$, $P \vdash \frac{N}{\sigma'(\alpha)}$ for $\alpha \in H$, $\sigma'(x_i) = t_i$ for $i = 1, \ldots, ar(f)$ and $\sigma'(u) = t'$. Thus $r \in R'$. For the negative literals α in H the proof of $\frac{N}{\sigma'(\alpha)}$ is trivial. As $\frac{H}{f(x_1,\ldots,x_{ar(f)}) \xrightarrow{a} u}$ is a decent xynft rule, the positive literals $\sigma'(\alpha)$ with $\alpha \in H$ have the form $t_i \xrightarrow{b} v$. Hence by induction $P' \vdash \frac{N}{\sigma'(\alpha)}$ for $\alpha \in H$. Moreover, $\sigma'(r) = \frac{\sigma'(H)}{f(t_1,\ldots,t_{ar(f)}) \xrightarrow{a} t'}$ is a substitution instance of a rule in R', so $P' \vdash \frac{N}{f(t_1,\ldots,t_{ar(f)}) \xrightarrow{a} t'}$. This concludes the proof of requirement (i). Requirements (ii), (iii) and (iv) are immediate corollaries of Lemmas 2, 3 and 4, respectively. $\square$

The results of this subsection are combined as follows:

Corollary 2 Let P be a standard TSS in ready simulation format. Then there exists a standard TSS P' in decent xynft format such that $P' \vdash_{ws} \alpha \Leftrightarrow P \vdash_{ws} \alpha$ for all closed literals α. Moreover if P is in ready trace format (resp. readiness format or failure trace format) then so is P'.

This also implies that $\sqsubseteq_N^{P'}$ equals $\sqsubseteq_N^P$ for each notion of observability N.

6.2 Reducing well-supported proofs to supported proofs

Define the *size* of a node in a well-supported proof (or in any well-founded, upwardly branching tree) to be the supremum of the sizes of the nodes above it, plus one. Write $P \vdash_{ws}^{\kappa} \alpha$ if there is a well-supported proof of the closed literal α from the TSS P of which the root has size no more than the ordinal κ. By straightforward induction, this is equivalent to the following recursive definition.

Definition 24 Let $P = (\Sigma, R)$ be a standard TSS. $P \vdash_{ws}^{\kappa+1} \alpha$ for α a closed literal iff

1. either there is a closed substitution instance $\frac{H}{\alpha}$ of a rule in R with $P \vdash_{ws}^{\kappa} \beta$ for all $\beta \in H$,
2. or α is negative and for every set N of negative closed literals such that $P \vdash \frac{N}{\gamma}$ for γ a closed literal denying α, one has $P \vdash_{ws}^{\kappa} \delta$ for a closed literal δ denying a literal in N.

$P \vdash_{ws}^{0} \alpha$ never holds, and in case κ is a limit ordinal, $P \vdash_{ws}^{\kappa} \alpha$ iff $P \vdash_{ws}^{\lambda} \alpha$ for some $\lambda < \kappa$.

Clearly, $P \vdash_{ws} \alpha$ iff $P \vdash_{ws}^{\kappa} \alpha$ for some ordinal κ. Moreover, if $\lambda < \kappa$ then $P \vdash_{ws}^{\lambda} \alpha \Rightarrow P \vdash_{ws}^{\kappa} \alpha$. Now we introduce the following concept of *supported provability*, that will be shown to coincide with well-supported provability for standard TSSs in decent xynft format.

Definition 25 Let $P = (\Sigma, R)$ be a standard TSS. $P \vdash_{s}^{\kappa+1} \alpha$ for α a closed literal iff

1. either there is a closed substitution instance $\frac{H}{\alpha}$ of a rule in R with $P \vdash_{s}^{\kappa} \beta$ for all $\beta \in H$,
2. or α is negative and for every closed substitution instance $\frac{H}{\gamma}$ of a rule in R with γ denying α, one has $P \vdash_{s}^{\kappa} \delta$ for a closed literal δ denying a literal in H.

$P \vdash_{s}^{0} \alpha$ never holds, and in case κ is a limit ordinal, $P \vdash_{s}^{\kappa} \alpha$ iff $P \vdash_{s}^{\lambda} \alpha$ for some $\lambda < \kappa$.
α is said to be *s-provable* from P, notation $P \vdash_{s} \alpha$, iff $P \vdash_{s}^{\kappa} \alpha$ for some ordinal κ.

Due to the absence in this paper of literals of the form $t \stackrel{a}{\not\to} t'$, the notion of supported provability defined above is in general less powerful than the notion of supported provability from [21] (cf. Counterexample R in that paper). However, on TSSs in decent xynft format both notions coincide, as will follow from Proposition 5.

Lemma 9 Let $\frac{H}{t \xrightarrow{a} t'}$ be a closed substitution instance of an ntytt rule r without lookahead, and let H' be a set of closed literals that differs from H only in the right-hand sides of its positive members. Then there exists a closed term t'' such that also $\frac{H'}{t \xrightarrow{a} t''}$ is a substitution instance of r.

Proof: Straightforward. □

Lemma 10 Let $P = (\Sigma, R)$ be a standard TSS in ntytt format without lookahead and α be a negative literal. Let κ be an ordinal. Then the following are equivalent.

(i) For every set N of negative closed literals such that $P \vdash \frac{N}{\gamma}$ for γ a closed literal denying α, one has $P \vdash^{\kappa}_{ws} \delta$ for a closed literal δ denying a literal in N (i.e. case 2 of Definition 24).

(ii) For every closed substitution instance $\frac{H}{\gamma}$ of a rule in R with γ denying α, H contains either a positive literal β denying a closed literal δ with $P \vdash^{\kappa+1}_{ws} \delta$, or a negative literal β denying a closed literal δ with $P \vdash^{\kappa}_{ws} \delta$.

Proof: Suppose (ii) holds. Let N be a set of negative closed literals such that $P \vdash \frac{N}{\gamma}$ for γ a closed literal denying α. Let $\frac{H}{\gamma}$ be the closed substitution instance of the rule in R used at the bottom of the proof of $\frac{N}{\gamma}$. Then H must contain either a positive literal β denying a closed literal δ with $P \vdash^{\kappa+1}_{ws} \delta$, or a negative literal β denying a closed literal δ with $P \vdash^{\kappa}_{ws} \delta$. In the latter case, using that P is standard, it must be that $\beta \in N$ and we are done. In the former case, we have $P \vdash \frac{N}{\beta}$, so by Definition 24 (taking δ and β for α and γ, respectively), one has $P \vdash^{\kappa}_{ws} \delta'$ for a closed literal δ' denying a literal in N, which had to be proved.

Suppose (ii) does not hold. Let $\frac{H}{\gamma}$ be a closed substitution instance of a rule in R with γ denying α, such that H does not contain the specified literal β, i.e. $P \nvdash^{\kappa+1}_{ws} u \stackrel{c}{\not\longrightarrow}$ for every positive premise $u \stackrel{c}{\longrightarrow} u'$ in H, and $P \nvdash^{\kappa}_{ws} u \stackrel{c}{\longrightarrow} u'$ for every negative premise $u \stackrel{c}{\not\longrightarrow}$ in H and every closed term u'. It suffices to find a set of negative closed literals N with $P \vdash \frac{N}{\gamma'}$ for γ' a closed literal denying α, such that $P \nvdash^{\kappa}_{ws} \delta$ for all closed literals δ denying a literal in N.

By Definition 24, for every positive premise $\beta = u \stackrel{c}{\longrightarrow} u'$ in H there must be a set N_β of negative closed literals with $P \vdash \frac{N_\beta}{u \stackrel{c}{\longrightarrow} u''}$ for u'' a closed term, such that $P \nvdash^{\kappa}_{ws} \delta$ for all closed literals δ denying a literal in N_β. Let H^+ be the set of positive and H^- the set of negative literals in H, and let $N = \bigcup_{\beta \in H^+} N_\beta \cup H^-$. Then $P \nvdash^{\kappa}_{ws} \delta$ for all closed literals δ denying a literal in N. By Lemma 9 there exists a substitution instance $\frac{H'}{\gamma'}$ of a rule in P, where H' is obtained from H by replacing the right-hand sides u' of its positive members by u'', and γ' denies α. Hence, $P \vdash \frac{N}{\gamma'}$.
□

Proposition 5 Let $P = (\Sigma, R)$ be a standard TSS in decent xynft format. Then

$$P \vdash_s \alpha \Leftrightarrow P \vdash_{ws} \alpha.$$

Proof: For "⇒" we prove $P \vdash^{\kappa}_{s} \alpha \Rightarrow P \vdash^{\kappa}_{ws} \alpha$ with induction to κ. The case that $\kappa = 0$ or κ is a limit ordinal is trivial. So suppose $P \vdash^{\kappa+1}_{s} \alpha$, and consider the two cases provided by Definition 25.

CASE 1: Let $\frac{H}{\alpha}$ be a closed substitution instance of a rule in R with $P \vdash^{\kappa}_{s} \beta$ for all $\beta \in H$. Then, by induction, $P \vdash^{\kappa}_{ws} \beta$ for all $\beta \in H$, and hence $P \vdash^{\kappa+1}_{ws} \alpha$.

CASE 2: Suppose case 2 of Definition 25 applies, i.e. for every closed substitution instance $\frac{H}{\gamma}$ of a rule in R with γ denying α, H contains a literal β denying a closed literal δ with $P \vdash^{\kappa}_{s} \delta$. As the induction hypothesis yields $P \vdash^{\kappa}_{s} \delta \Rightarrow P \vdash^{\kappa}_{ws} \delta$, and by definition $P \vdash^{\kappa}_{ws} \delta \Rightarrow P \vdash^{\kappa+1}_{ws} \delta$, case (ii) of Lemma 10 applies, and hence also case (i). It follows that $P \vdash^{\kappa+1}_{ws} \alpha$.

For "⇐" we prove $P \vdash^{\kappa}_{ws} \alpha \Rightarrow P \vdash_{s} \alpha$ with induction to κ, and a nested induction to the *size* of the left-hand side of α, i.e. to the largest number of nested function symbols in it. Again, the case that $\kappa = 0$ or κ is a limit ordinal is trivial. So suppose $P \vdash^{\kappa+1}_{ws} \alpha$, and consider the two cases provided by Definition 24.

CASE 1: Let $\frac{H}{\alpha}$ be a closed substitution instance of a rule in R with $P \vdash_{ws}^{\kappa} \beta$ for all $\beta \in H$. Then, by induction, $P \vdash_s \beta$ for all $\beta \in H$, and hence $P \vdash_s \alpha$.

CASE 2: Suppose case 2 of Definition 24 applies. Let $\frac{H}{\gamma}$ be a closed substitution instance of a rule r in R with γ denying α. By Lemma 10, H must contain either a positive literal β denying a literal δ with $P \vdash_{ws}^{\kappa+1} \delta$, or a negative literal β denying a literal δ with $P \vdash_{ws}^{\kappa} \delta$. In the latter case, the induction hypothesis gives $P \vdash_s \delta$. In the former case, using that r is a decent xynft rule, the left-hand side of δ (and β) is smaller than the left-hand side of α (and γ), so the nested induction hypothesis allows to conclude that $P \vdash_s \delta$. It follows that $P \vdash_s \alpha$. □

The following counterexample shows that the restriction to decent xynft format is essential here.

Example 1 Let $A = \{a\}$, let Σ consist of the constant c, and let R be the decent ntyxt rule

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

No closed rule of the form $\frac{N}{c \xrightarrow{a} u}$, where N contains only closed negative literals, is provable from $P = (\Sigma, R)$. Hence, $P \vdash_{ws} c \stackrel{a}{\not\longrightarrow}$. However, since $\frac{c \xrightarrow{a} c}{c \xrightarrow{a} c}$ is a closed substitution instance of the rule in R, we have $P \not\vdash_s c \stackrel{a}{\not\longrightarrow}$.

This shows that Proposition 5 does not extend to rules in which the source is a variable, and explains why we needed Proposition 2. The example also applies when we change R into

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

This shows that Proposition 5 does not extend to rules with free variables, and explains why we needed Proposition 3. When we change R into

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

the example shows that Proposition 5 does not extend to ntyft rules, and explains why we needed Proposition 4. In order to see that Proposition 5 does not extend to rules with lookahead, let Σ consist of the constant c and the unary function symbol f, and let R be

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

This time we obtain $P \vdash_{ws} f(c) \stackrel{a}{\not\longrightarrow}$, but $P \not\vdash_s f(c) \stackrel{a}{\not\longrightarrow}$.

6.3 Reducing supported proofs to standard proofs

We proceed to show that for any given standard TSS $P = (\Sigma, R)$ in decent ntyft format there exists a TSS $P^+ = (\Sigma, R^+)$ in decent ntyft format such that $P \vdash_s \alpha \Leftrightarrow P^+ \vdash \alpha$. Again, the translation of R into R^+ preserves the ready trace, readiness and failure trace formats. The construction below uses the absence of lookahead in an essential way.

Definition 26 Let $P = (\Sigma, R)$ be a TSS, $t \in \mathbb{T}(\Sigma)$ and $a \in A$.

- $cl(R)$ denotes the collection of closed substitution instances of rules in R.
- $R \restriction (t \xrightarrow{a})$ denotes the set of rules in R with conclusion $t \xrightarrow{a} u$ for some $u \in \mathbb{T}(\Sigma)$.
- $pick(R, t \stackrel{a}{\not\longrightarrow})$ denotes the collection of transition rules $\frac{H}{t \stackrel{a}{\not\longrightarrow}}$ in which H is obtained by taking one premise from every rule in $R \restriction (t \xrightarrow{a})$.
- $deny(R)$ denotes the collection of rules obtained from R by changing in each rule every positive premise $u \xrightarrow{c} u'$ into $u \stackrel{c}{\not\longrightarrow}$, and every negative premise $u \stackrel{c}{\not\longrightarrow}$ into $u \xrightarrow{c} y$, where the y are all different variables, not occurring elsewhere in the rule. (That there are that many different variables follows from Lemma 6; if needed the variables in the rule that do not occur in its source may be renamed.)
- $prove(R, t \stackrel{a}{\not\longrightarrow})$ denotes $deny(pick(R, t \stackrel{a}{\not\longrightarrow}))$.
- Let $\overline{R}$ be the union of $cl(R)$ and all collections $cl(prove(cl(R), \alpha))$ for negative closed literals α.

Note that $cl(R) \restriction (t \xrightarrow{a})$ with t closed is the set of closed substitution instances $\frac{H}{\gamma}$ of rules in R with γ denying $t \stackrel{a}{\not\longrightarrow}$. Thus $cl(prove(cl(R), \alpha))$, for α a negative literal, contains only closed rules $\frac{K}{\alpha}$ such that for every closed substitution instance $\frac{H}{\gamma}$ of a rule in R with γ denying α, a literal in K denies a literal in H. Moreover, for every closed rule $\frac{K}{\alpha}$ of the latter kind, there is a $K' \subseteq K$ such that $\frac{K'}{\alpha} \in cl(prove(cl(R), \alpha))$. From this we obtain:

Lemma 11 Let $P = (\Sigma, R)$ and $P' = (\Sigma, R')$ be TSSs with P standard and $cl(R') = \overline{R}$. Then $P' \vdash \alpha \Leftrightarrow P \vdash_s \alpha$. □

Definition 27 A TSS $P = (\Sigma, R)$ is in *uniform ntyft format* if it is in ntyft format and for every function symbol $f \in \Sigma$ there is a specific sequence of all different variables $\vec{x}_f = (x_1, \ldots, x_{ar(f)})$, such that all sources of rules in R have the form $f(\vec{x}_f)$ for some $f \in \Sigma$.

Let P be in uniform ntyft format. Then R^+ denotes the union of R and all collections $prove(R, f(\vec{x}_f) \stackrel{a}{\not\longrightarrow})$ for $f \in \Sigma$ and $a \in A$.

Example 2 Let $A = \{a, b\}$ and let Σ consist of the constant c and the binary function symbol f. Let R be:

$$c \xrightarrow{a} c \qquad \frac{x_1 \xrightarrow{a} y}{f(x_1, x_2) \xrightarrow{b} y} \qquad \frac{x_2 \xrightarrow{a} y \quad x_1 \stackrel{b}{\not\longrightarrow}}{f(x_1, x_2) \xrightarrow{b} y}$$

Then R^+ is R together with:

$$c \stackrel{b}{\not\longrightarrow} \qquad f(x_1, x_2) \stackrel{a}{\not\longrightarrow} \qquad \frac{x_1 \stackrel{a}{\not\longrightarrow} \quad x_2 \stackrel{a}{\not\longrightarrow}}{f(x_1, x_2) \stackrel{b}{\not\longrightarrow}} \qquad \frac{x_1 \stackrel{a}{\not\longrightarrow} \quad x_1 \xrightarrow{b} z}{f(x_1, x_2) \stackrel{b}{\not\longrightarrow}}$$

Definition 28 Write $r \equiv r'$ for transition rules r and r' if they differ only through a bijective renaming of their variables (*α-conversion*). Write $R \equiv R'$ if for each $r \in R$ there is an $r' \in R'$ with $r \equiv r'$ and vice versa.

Write $r \approx r'$ for transition rules r and r' if they differ only in their targets and in the right-hand sides of their premises. Write $R \approx R'$ if for each $r \in R$ there is an $r' \in R'$ with $r \approx r'$ and vice versa.

Note that for any TSS $P=(\Sigma,R)$ in decent ntyft format there is a TSS $P'=(\Sigma,R')$ in decent uniform ntyft format with $R\equiv R'$. Moreover, $P\vdash_s\beta \Leftrightarrow P'\vdash_s\beta$ for any literal β; also P' is in ready trace/readiness/failure trace format iff P is. Finally note that if $R\approx R'$ and all rules in R and R' have negative conclusions then $deny(R)\equiv deny(R')$.

Lemma 12 Let $P=(\Sigma,R)$ be a TSS in decent uniform ntyft format, and let σ be a closed substitution. Then $\sigma(R\restriction(f(\vec{x}_f)\xrightarrow{a}))\approx cl(R)\restriction(\sigma(f(\vec{x}_f))\xrightarrow{a})$.

Proof: "$\subseteq$": If rule $r\in R$ has conclusion $f(\vec{x}_f)\xrightarrow{a}u$ then $\sigma(r)\in\sigma(R)\subseteq cl(R)$ has conclusion $\sigma(f(\vec{x}_f))\xrightarrow{a}\sigma(u)$.

"$\supseteq$": Let $r\in cl(R)\restriction(\sigma(f(\vec{x}_f))\xrightarrow{a})$. Then $r=\rho(r')$ for $r'\in R$ and ρ a closed substitution, and the conclusion of r is $\sigma(f(\vec{x}_f))\xrightarrow{a}u$ for some term u. As P is in uniform ntyft format this implies that the conclusion of r' is $f(\vec{x}_f)\xrightarrow{a}u'$ for some term u', and $\rho(x_i)=\sigma(x_i)$ for $i=1,\ldots,ar(f)$. As r is decent, the left-hand sides of its premises contain no other variables than $x_1,\ldots,x_{ar(f)}$. Therefore $r=\rho(r')\approx\sigma(r')\in\sigma(R\restriction(f(\vec{x}_f)\xrightarrow{a}))$. □

Proposition 6 Let $P=(\Sigma,R)$ be a standard TSS in decent uniform ntyft format. Then also $P^+=(\Sigma,R^+)$ is in decent uniform ntyft format, and $P^+\vdash\alpha\Leftrightarrow P\vdash_s\alpha$.

Proof: The first statement is immediate from the construction. For the second, using Lemma 11, it suffices to show that $cl(R^+)\equiv\overline{R}$. First of all, for any $f\in\Sigma$, $a\in A$ and closed substitution σ,

$\sigma(pick(R,f(\vec{x}_f)\stackrel{a}{\nrightarrow}))=$

$\{\frac{\sigma(H)}{\sigma(f(\vec{x}_f))\stackrel{a}{\nrightarrow}}\mid H$ is obtained by taking one premise from each rule in $R\restriction f(\vec{x}_f)\xrightarrow{a}\}=$

$\{\frac{K}{\sigma(f(\vec{x}_f))\stackrel{a}{\nrightarrow}}\mid K$ is obtained by taking one premise from each rule in $\sigma(R\restriction f(\vec{x}_f)\xrightarrow{a})\}\stackrel{\text{Lemma 12}}{\approx}$

$\{\frac{K}{\sigma(f(\vec{x}_f))\stackrel{a}{\nrightarrow}}\mid K$ is obtained by taking one premise from each rule in $cl(R)\restriction\sigma(f(\vec{x}_f))\xrightarrow{a}\}=$

$pick(cl(R),\sigma(f(\vec{x}_f)\stackrel{a}{\nrightarrow}))$.

Therefore, $cl(R^+)=cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}cl(prove(R,f(\vec{x}_f)\stackrel{a}{\nrightarrow}))=$
$cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}cl(deny(pick(R,f(\vec{x}_f)\stackrel{a}{\nrightarrow})))=$
$cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}cl(deny(cl(pick(R,f(\vec{x}_f)\stackrel{a}{\nrightarrow}))))=$
$cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}cl(deny(\bigcup_{\sigma:V\to T(\Sigma)}\sigma(pick(R,f(\vec{x}_f)\stackrel{a}{\nrightarrow}))))\equiv$
$cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}cl(deny(\bigcup_{\sigma:V\to T(\Sigma)}pick(cl(R),\sigma(f(\vec{x}_f)\stackrel{a}{\nrightarrow}))))=$
$cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}cl(deny(\bigcup_{t_1,\ldots,t_{ar(f)}\in T(\Sigma)}pick(cl(R),f(t_1,\ldots,t_{ar(f)})\stackrel{a}{\nrightarrow})))=$
$cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}\bigcup_{t_1,\ldots,t_{ar(f)}\in T(\Sigma)}cl(deny(pick(cl(R),f(t_1,\ldots,t_{ar(f)})\stackrel{a}{\nrightarrow})))=$
$cl(R)\cup\bigcup_{a\in A}\bigcup_{f\in\Sigma}\bigcup_{t_1,\ldots,t_{ar(f)}\in T(\Sigma)}cl(prove(cl(R),f(t_1,\ldots,t_{ar(f)})\stackrel{a}{\nrightarrow}))=$
$cl(R)\cup\bigcup_{\alpha\text{ a closed negative literal}}cl(prove(cl(R),\alpha))=\overline{R}$. □

The following counterexample shows that absence of free variables is essential here.

Example 3 Let $A=\{a\}$, let Σ consist of the constants a, c and ϵ, and let R be the following collection of rules:

$$a\xrightarrow{a}\epsilon \qquad \frac{x\xrightarrow{a}y}{c\xrightarrow{a}y}$$

Note that $P = (\Sigma, R)$ is a standard TSS in uniform ntyft format. Clearly $P \vdash c \xrightarrow{a} \epsilon$ and thus certainly $P \vdash_s c \xrightarrow{a} \epsilon$ and thus $P \not\vdash_s c \stackrel{a}{\not\longrightarrow}$. Nevertheless, R^+ would be R together with the rules $\epsilon \stackrel{a}{\not\longrightarrow}$ and $\dfrac{x \stackrel{a}{\not\longrightarrow}}{c \stackrel{a}{\not\longrightarrow}}$, so we would have $(\Sigma, R^+) \vdash c \stackrel{a}{\not\longrightarrow}$.

The following counterexample shows that absence of lookahead is essential too.

Example 4 Let $A = \{a\}$, let Σ consist of the constants a and ϵ and the unary function symbol f, and let R be the following collection of rules:

$$a \xrightarrow{a} \epsilon \qquad \frac{x \xrightarrow{a} y \quad y \stackrel{a}{\not\longrightarrow}}{f(x) \xrightarrow{a} y}$$

Again $P = (\Sigma, R)$ is a standard TSS in uniform ntyft format. As there are no rules with conclusion $\epsilon \xrightarrow{a} t$ for some term t, one obtains $P \vdash_s \epsilon \stackrel{a}{\not\longrightarrow}$, and hence $P \vdash_s f(a) \xrightarrow{a} \epsilon$. Nevertheless, R^+ would be R together with the rules $\epsilon \stackrel{a}{\not\longrightarrow}$, $\dfrac{x \stackrel{a}{\not\longrightarrow}}{f(x) \stackrel{a}{\not\longrightarrow}}$ and $\dfrac{y \xrightarrow{a} z}{f(x) \stackrel{a}{\not\longrightarrow}}$, so we would have $(\Sigma, R^+) \vdash f(c) \stackrel{a}{\not\longrightarrow}$.

It remains to show the relevant formats are preserved under the translation of R into R^+. Recall that these formats were extended to non-standard TSSs in Definition 23.

Proposition 7 Let $P = (\Sigma, R)$ be a TSS in decent uniform ntyft format. If P is in ready trace format (resp. readiness format or failure trace format), then so is $P^+ = (\Sigma, R^+)$.

Proof: By Definition 23 the statements for the ready trace and readiness formats are trivial.

Let P be in failure trace format, let $f \in \Sigma$ and $a \in A$, and let z be Λ-floating in a rule r in $\mathit{prove}(R, f(\vec{x}_f) \stackrel{a}{\not\longrightarrow})$. In case z is the right-hand side of a positive premise, it is not polled in r. So assume z occurs in the source of r. As in the rules of $R \upharpoonright (f(\vec{x}_f) \xrightarrow{a})$ the variable z is also Λ-floating, it is polled only in Λ-liquid positions in those rules, and only in positive premises. The same holds for the rules in $\mathit{pick}(R, f(\vec{x}_f) \stackrel{a}{\not\longrightarrow})$. Hence in the rules of $\mathit{deny}(\mathit{pick}(R, f(\vec{x}_f) \stackrel{a}{\not\longrightarrow}))$ it is polled only in Λ-liquid positions in negative premises. $\Box$

The results of this section can now be combined as follows, extending the definition of R^+ to TSSs (Σ, R) in ready simulation format.

Definition 29 Let $P = (\Sigma, R)$ be a TSS in ready simulation format. Then $P^+ = (\Sigma, R^+)$ is defined as the TSS obtained by subsequently

- convert P to ntyft format without lookahead following the construction in the proof of Prop. 2,
- convert the result to decent ntyft format following the construction in the proof of Proposition 3,
- convert the result to decent xynft format following the construction in the proof of Proposition 4,
- convert the result to decent uniform xynft format by bijectively renaming the variables in each of its rules,
- and applying the $^+$ construction of Definition 27.

Note that R^+ is fully determined by R up to α-conversion.

Corollary 3 Let $P = (\Sigma, R)$ be a standard TSS in ready simulation format. Then $P^+ = (\Sigma, R^+)$ is in decent ntyft format, $P^+ \vdash \alpha \Leftrightarrow P \vdash_{ws} \alpha$ for all closed literals α, and if P is in ready trace format (resp. readiness format or failure trace format) then so is P^+.

7 Reducing decent ntyft rules to decent nxytt rules

The following proposition says, for P a TSS in decent ntyft format, that for any function symbol f there are a number of decent nxyft rules with source $f(x_1,\ldots,x_{ar(f)})$ irredundantly provable from P that are "just right" for f. Here "just right" means that for any (closed) literal $f(t_1,\ldots,t_{ar(f)}) \overset{a}{\nrightarrow}$ or $f(t_1,\ldots,t_{ar(f)}) \xrightarrow{a} t'$ that is provable from P there is a (closed) proof using one of those rules as the last step. Moreover, the same holds not only for function symbols f, but for arbitrary open terms. If for an open term t with $var(t)=\{x_1,\ldots,x_n\}$ we would introduce an n-ary function symbol f_t such that $f_t(x_1,\ldots,x_n)$ is just a shorthand for t, then P also irredundantly proves a set of decent nxyft rules that are just right for f_t.

Proposition 8 and its proof below can be read in four ways: pertaining to negative or positive literals (with the material pertaining only to positive literals enclosed in square brackets), and with and without the adjective "closed".

Proposition 8 Let $P=(\Sigma,R)$ be a TSS in decent ntyft format. If $P\vdash\sigma(t)\overset{a}{\nrightarrow}$ [resp. $P\vdash\sigma(t)\xrightarrow{a}t'$] for t a term, [t' a (closed) term] and σ a (closed) substitution, then there are a decent nxytt rule $\frac{H}{t\overset{a}{\nrightarrow}}$ [resp. $\frac{H}{t\xrightarrow{a}u}$] and a (closed) substitution σ' with $P\vdash_{\rm irr}\frac{H}{t\overset{a}{\nrightarrow}}$ [resp. $P\vdash_{\rm irr}\frac{H}{t\xrightarrow{a}u}$], $P\vdash\sigma'(H)$, $\sigma'(t)=\sigma(t)$ [and $\sigma'(u)=t'$].

Proof: First, suppose t is a variable. By default, the decent nxytt rule $\frac{t\overset{a}{\nrightarrow}}{t\overset{a}{\nrightarrow}}$ [resp. $\frac{t\xrightarrow{a}y}{t\xrightarrow{a}y}$] is irredundantly provable from P. Let σ' be a (closed) substitution with $\sigma'(t)=\sigma(t)$ [and $\sigma'(y)=t'$]. Clearly, $P\vdash\sigma'(t\overset{a}{\nrightarrow})$ [resp. $P\vdash\sigma'(t\xrightarrow{a}y)$].

Next, suppose $t=f(t_1,\ldots,t_{ar(f)})$. We apply structural induction to a (closed) proof π of the literal $\sigma(t)\overset{a}{\nrightarrow}$ [resp. $\sigma(t)\xrightarrow{a}t'$] from P. Let $r\in R$ be the decent ntyft rule and ρ be the (closed) substitution used at the bottom of π, where r is of the form $\frac{\{v_k\xrightarrow{c_k}y_k\mid k\in K\}\cup\{w_\ell\overset{d_\ell}{\nrightarrow}\mid \ell\in L\}}{f(x_1,\ldots,x_{ar(f)})\overset{a}{\nrightarrow}}$ [resp. $\frac{\{v_k\xrightarrow{c_k}y_k\mid k\in K\}\cup\{w_\ell\overset{d_\ell}{\nrightarrow}\mid \ell\in L\}}{f(x_1,\ldots,x_{ar(f)})\xrightarrow{a}v}$]. Then $\rho(x_i)=\sigma(t_i)$ for $i=1,\ldots,ar(f)$, [$\rho(v)=t'$] and $\rho(v_k)\xrightarrow{c_k}\rho(y_k)$ for $k\in K$ and $\rho(w_\ell)\overset{d_\ell}{\nrightarrow}$ for $\ell\in L$ are provable from P by means of strict subproofs of π. Since r is decent, $var(v_k)$ for $k\in K$ and $var(w_\ell)$ for $\ell\in L$ are included in $\{x_1,\ldots,x_{ar(f)}\}$. Let ρ_0 be a substitution with $\rho_0(x_i)=t_i$ for $i=1,\ldots,ar(f)$. As $\rho(x_i)=\sigma(t_i)=\sigma(\rho_0(x_i))$ for $i=1,\ldots,ar(f)$, we have $\rho(v_k)=\sigma(\rho_0(v_k))$ for $k\in K$ and $\rho(w_\ell)=\sigma(\rho_0(w_\ell))$ for $\ell\in L$. So $\sigma(\rho_0(v_k))\xrightarrow{c_k}\rho(y_k)$ for $k\in K$ and $\sigma(\rho_0(w_\ell))\overset{d_\ell}{\nrightarrow}$ for $\ell\in L$ are provable from P by means of strict subproofs of π. According to the induction hypothesis, for $k\in K$ there are a decent nxytt rule $\frac{H_k}{\rho_0(v_k)\xrightarrow{c_k}u_k}$ and a (closed) substitution σ'_k with $P\vdash_{\rm irr}\frac{H_k}{\rho_0(v_k)\xrightarrow{c_k}u_k}$, $P\vdash\sigma'_k(H_k)$, $\sigma'_k(\rho_0(v_k))=\sigma(\rho_0(v_k))$ and $\sigma'_k(u_k)=\rho(y_k)$. Likewise, for $\ell\in L$ there are a decent nxytt rule $\frac{H_\ell}{\rho_0(w_\ell)\overset{d_\ell}{\nrightarrow}}$ and a (closed) substitution σ'_ℓ with $P\vdash_{\rm irr}\frac{H_\ell}{\rho_0(w_\ell)\overset{d_\ell}{\nrightarrow}}$, $P\vdash\sigma'_\ell(H_\ell)$ and $\sigma'_\ell(\rho_0(w_\ell))=\sigma(\rho_0(w_\ell))$. By Lemma 6, we can choose the sets of variables in the right-hand sides of the positive premises in the H_k (for $k\in K$) and H_ℓ (for $\ell\in L$) pairwise disjoint, and disjoint from $var(t)$. This allows us to define a (closed) substitution σ' with:

- $\sigma'(z)=\sigma(z)$ for $z\in var(t)$;
- $\sigma'(z)=\sigma'_k(z)$ for right-hand sides z of positive premises in H_k for $k\in K$;
- $\sigma'(z)=\sigma'_\ell(z)$ for right-hand sides z of positive premises in H_ℓ for $\ell\in L$.

Let

$$H \;=\; \bigcup_{k\in K} H_k \;\cup\; \bigcup_{\ell\in L} H_\ell\ .$$

Moreover, let ρ_1 be a substitution with $\rho_1(x_i)=t_i$ for $i=1,\ldots,ar(f)$ and $\rho_1(y_k)=u_k$ for $k\in K$. We verify that the rule $\frac{H}{t\xrightarrow{a}\!\!\!\!\!/\ \ }$ [resp. $\frac{H}{t\xrightarrow{a}\rho_1(v)}$] together with the substitution σ' satisfy the desired properties.

As $var(v_k)\subseteq\{x_1,\ldots,x_{ar(f)}\}$, it follows that $var(\rho_0(v_k))\subseteq var(t)$. Since σ' and σ agree on $var(t)$, $\sigma'(\rho_0(v_k))=\sigma(\rho_0(v_k))=\sigma'_k(\rho_0(v_k))$ for $k\in K$. Thus, by the decency of $\frac{H_k}{\rho_0(v_k)\xrightarrow{c_k}u_k}$, σ' and σ'_k agree on all variables occurring in this rule for $k\in K$. Likewise, σ' and σ'_ℓ agree on all variables occurring in $\frac{H_\ell}{\rho_0(w_\ell)\xrightarrow{d_\ell}\!\!\!\!\!/\ \ }$ for $\ell\in L$.

1. ρ_1 and ρ_0 agree on $var(v_k)\subseteq\{x_1,\ldots,x_{ar(f)}\}$, so $\rho_1(v_k)=\rho_0(v_k)$ for $k\in K$. Likewise, $\rho_1(w_\ell)$ $=\rho_0(w_\ell)$ for $\ell\in L$. Since $P\vdash_{\rm irr} r$, we have $P\vdash_{\rm irr}\rho_1(r)=\frac{\{\rho_0(v_k)\xrightarrow{c_k}u_k\mid k\in K\}\cup\{\rho_0(w_\ell)\xrightarrow{d_\ell}\!\!\!\!\!/\ \ \mid \ell\in L\}}{t\xrightarrow{a}\!\!\!\!\!/\ \ }$ [resp. $\frac{\{\rho_0(v_k)\xrightarrow{c_k}u_k\mid k\in K\}\cup\{\rho_0(w_\ell)\xrightarrow{d_\ell}\!\!\!\!\!/\ \ \mid \ell\in L\}}{t\xrightarrow{a}\rho_1(v)}$]. Furthermore, $P\vdash_{\rm irr}\frac{H_k}{\rho_0(v_k)\xrightarrow{c_k}u_k}$ for $k\in K$ and $P\vdash_{\rm irr}$ $\frac{H_\ell}{\rho_0(w_\ell)\xrightarrow{d_\ell}\!\!\!\!\!/\ \ }$ for $\ell\in L$. As $H=\bigcup_{k\in K}H_k\cup\bigcup_{\ell\in L}H_\ell$, it follows that $P\vdash_{\rm irr}\frac{H}{t\xrightarrow{a}\!\!\!\!\!/\ \ }$ [resp. $P\vdash_{\rm irr}\frac{H}{t\xrightarrow{a}\rho_1(v)}$].

2. The right-hand sides of the positive premises in any H_k or H_ℓ are distinct variables. By construction, these sets of variables (one for every $k\in K$ and $\ell\in L$) are pairwise disjoint, and disjoint from $var(t)$. Hence $\frac{H}{t\xrightarrow{a}\!\!\!\!\!/\ \ }$ [resp. $\frac{H}{t\xrightarrow{a}\rho_1(v)}$] is an ntytt rule. Since the premises in H originate from H_k (for $k\in K$) and H_ℓ (for $\ell\in L$), their left-hand sides are variables. This makes the rule an nxytt rule. The rule is decent by Lemma 1.

3. σ' agrees with σ'_k on variables in H_k for $k\in K$, and σ' agrees with σ'_ℓ on variables in H_ℓ for $\ell\in L$. Since $P\vdash\sigma'_k(H_k)$ for $k\in K$ and $P\vdash\sigma'_\ell(H_\ell)$ for $\ell\in L$, we conclude that $P\vdash\sigma'(H)$.

4. Since σ' and σ agree on $var(t)$, $\sigma'(t)=\sigma(t)$.

5. [$\sigma'(\rho_1(x_i))=\sigma'(t_i)=\sigma(t_i)=\rho(x_i)$ for $i=1,\ldots,ar(f)$. Moreover, since σ' and σ'_k agree on $var(u_k)$, $\sigma'(\rho_1(y_k))=\sigma'(u_k)=\sigma'_k(u_k)=\rho(y_k)$ for $k\in K$. As $var(v)\subseteq\{x_1,\ldots,x_{ar(f)}\}\cup\{y_k\mid k\in K\}$, it follows that $\sigma'(\rho_1(v))=\rho(v)=t'$.] □

Note the similarity between Lemma 7 and Proposition 8 and their proofs. The essential difference is that Lemma 7 employs a definition of "just right" that deals with transition rules with negative premises instead of mere literals. The price to be paid for that is that the constructed rules may contain arbitrary negative premises, instead of premises of the form $x\xrightarrow{a}\!\!\!\!\!/\ \ $. Furthermore Lemma 7 deals only with standard TSSs (and does not mention the closed case).

The following corollary will not be needed in the remainder of this paper; however it may be interesting in its own right.

Corollary 4 Let $P=(\Sigma,R)$ be a TSS in decent ntyft format. Then there exists a TSS $P'=(\Sigma,R')$ in decent nxyft format such that $P'\vdash\alpha\Leftrightarrow P\vdash\alpha$ for all closed literals α. Moreover if P is in ready trace format (resp. readiness format or failure trace format) then so is P'.

Proof: Let R' consist of all decent nxyft rules irredundantly provable from P. That $P' \vdash \alpha \Rightarrow P \vdash \alpha$ follows immediately from Lemma 8. For the other direction we use structural induction on the source of α. So assume $P \vdash \alpha$ with α being $f(t_1,\ldots,t_{ar(f)}) \stackrel{a}{\not\longrightarrow}$ or $f(t_1,\ldots,t_{ar(f)}) \stackrel{a}{\longrightarrow} t'$. By Proposition 8 there are a decent nxyft rule $r = \frac{H}{f(x_1,\ldots,x_{ar(f)}) \stackrel{a}{\not\longrightarrow}}$ [resp. $\frac{H}{f(x_1,\ldots,x_{ar(f)}) \stackrel{a}{\longrightarrow} u}$] and a substitution σ' with $P \vdash_{\rm irr} r$, $P \vdash \sigma'(H)$, $\sigma'(x_i) = t_i$ for $i = 1,\ldots,ar(f)$ and $\sigma'(u) = t'$. Thus $r \in R'$. As r is a decent nxyft rule, the literals $\sigma'(\beta)$ with $\beta \in H$ have the form $t_i \stackrel{b}{\longrightarrow} v$ or $t_i \stackrel{b}{\not\longrightarrow}$. Hence by induction $P' \vdash \sigma'(H)$. Moreover, $\sigma'(r) = \frac{\sigma'(H)}{\alpha}$ is a substitution instance of a rule in R', so $P' \vdash \alpha$. This concludes the proof of the first statement. The second is an immediate corollary of Lemmas 2, 3, 4 and 5. $\square$

8 Crux

This section witnesses the proofs of Theorems 4-7; see Corollary 5. The bulk of this section consists of lemmas building up to this corollary.

Definition 30 Let $P = (\Sigma, R)$ be a standard TSS in ready simulation format. A *P-ruloid* is a decent nxytt rule, irredundantly provable from $P^+ = (\Sigma, R^+)$.

Lemma 13 Let P be a standard TSS in ready simulation format. Then $P \vdash_{ws} \sigma(t) \stackrel{a}{\not\longrightarrow}$ [resp. $P \vdash_{ws} \sigma(t) \stackrel{a}{\longrightarrow} t'$] for t a term, [t' a closed term] and σ a closed substitution, iff there are a P-ruloid $\frac{H}{t \stackrel{a}{\not\longrightarrow}}$ [resp. $\frac{H}{t \stackrel{a}{\longrightarrow} u}$] and a closed substitution σ' with $P \vdash_{ws} \sigma'(H)$, $\sigma'(t) = \sigma(t)$ [and $\sigma'(u) = t'$].

Proof: By Corollary 3, P^+ is in decent ntyft format and $P^+ \vdash \alpha \Leftrightarrow P \vdash_{ws} \alpha$ for all closed literals α. Using this, "if" follows immediately from Lemma 8, and "only if" from Proposition 8. $\square$

The following definition assigns to each term and each observation in $\mathbb{O}_{RS}$ a collection of mappings from variables to $\mathbb{O}_{RS}$. This construct plays a crucial rôle in the technical developments to follow. Intuitively, $t_P^{-1}(\varphi)$ consists of observational reformulations of the P-ruloids with source t, that validate the observation φ for the term $\sigma(t)$, for any closed substitution σ, in terms of the observations that can be made for the terms $\sigma(x)$ for $x \in var(t)$.

Definition 31 Let $P = (\Sigma, R)$ be a standard TSS in ready simulation format. Then $\cdot_P^{-1} : \mathbb{T}(\Sigma) \rightarrow (\mathbb{O}_{RS} \rightarrow \mathcal{P}(V \rightarrow \mathbb{O}_{RS}))$ is defined by:

- $t_P^{-1}(\top) = \{\psi\}$ with $\psi(x) = \top$ for $x \in V$.
- $\psi \in t_P^{-1}(\widetilde{a})$ iff there is a P-ruloid $\frac{H}{t \stackrel{a}{\not\longrightarrow}}$ and $\psi : V \rightarrow \mathbb{O}$ is given by

$$\begin{array}{ll} \psi(x) = \displaystyle\bigwedge_{(x \stackrel{b}{\not\longrightarrow}) \in H} \widetilde{b} \;\wedge \bigwedge_{(x \stackrel{c}{\longrightarrow} y) \in H} c\top & \text{for } x \in var(t) \\ \psi(x) = \top & \text{for } x \notin var(t). \end{array}$$

- $\psi \in t_P^{-1}(a\varphi)$ iff there are a P-ruloid $\frac{H}{t \stackrel{a}{\longrightarrow} u}$ and a $\chi \in u_P^{-1}(\varphi)$ and $\psi : V \rightarrow \mathbb{O}$ is given by

$$\begin{array}{ll} \psi(x) = \chi(x) \wedge \displaystyle\bigwedge_{(x \stackrel{b}{\not\longrightarrow}) \in H} \widetilde{b} \;\wedge \bigwedge_{(x \stackrel{c}{\longrightarrow} y) \in H} c\chi(y) & \text{for } x \in var(t) \\ \psi(x) = \top & \text{for } x \notin var(t). \end{array}$$

- $t_P^{-1}(\bigwedge_{i\in I}\varphi_i) = \{\bigwedge_{i\in I}\psi_i \mid \psi_i \in t_P^{-1}(\varphi_i) \text{ for } i \in I\}$.

When clear from the context, the subscript P will be omitted.

It is not hard to see that if $\psi \in t^{-1}(\varphi)$ then $\psi(x) \cong \top$ for all $x \notin \mathit{var}(t)$.

Lemma 14 Let $P = (\Sigma, R)$ be a standard TSS in ready simulation format. Let $\varphi \in \mathbb{O}$. For any term $t \in \mathbb{T}(\Sigma)$ and closed substitution $\sigma : V \to T(\Sigma)$ one has

$$\sigma(t) \models_P \varphi \quad\Leftrightarrow\quad \exists\psi \in t_P^{-1}(\varphi)\ \forall x \in \mathit{var}(t) : \sigma(x) \models_P \psi(x).$$

Proof: With induction on the structure of φ.

- $\varphi = \top$. In this case $\sigma(t) \models \top$ and for the only parameterized observation $\psi \in t^{-1}(\top)$ one has $\sigma(x) \models \psi(x) = \top$ for all $x \in \mathit{var}(t)$.

- Suppose $\sigma(t) \models \widetilde{a}$. Then $P \vdash_{ws} \sigma(t) \stackrel{a}{\not\longrightarrow}$, by Definition 14. Thus, by Lemma 13 there must be a P-ruloid $\frac{H}{t \stackrel{a}{\not\longrightarrow}}$ and a closed substitution σ' with $P \vdash_{ws} \sigma'(H)$ and $\sigma'(t) = \sigma(t)$, i.e. $\sigma'(x) = \sigma(x)$ for $x \in \mathit{var}(t)$. Define ψ as indicated in Definition 31. By definition, $\psi \in t^{-1}(\widetilde{a})$. Let $x \in \mathit{var}(t)$. For $(x \xrightarrow{c} y) \in H$ one has $P \vdash_{ws} \sigma'(x) \xrightarrow{c} \sigma'(y)$, so $\sigma'(x) \models c\top$. Moreover, for $(x \stackrel{b}{\not\longrightarrow}) \in H$ one has $P \vdash_{ws} \sigma'(x) \stackrel{b}{\not\longrightarrow}$, so $\sigma'(x) \models \widetilde{b}$. Hence $\sigma(x) = \sigma'(x) \models \psi(x)$.

  Now suppose that there is a $\psi \in t^{-1}(\widetilde{a})$ such that $\sigma(x) \models \psi(x)$ for all $x \in \mathit{var}(t)$. This means that there is a P-ruloid $\frac{\{x \xrightarrow{a_i} y_i \mid i \in I_x,\ x \in \mathit{var}(t)\} \cup \{x \stackrel{b_j}{\not\longrightarrow} \mid j \in J_x,\ x \in \mathit{var}(t)\}}{t \stackrel{a}{\not\longrightarrow}}$ such that $\sigma(x) \models \bigwedge_{j\in J_x} \widetilde{b_j} \wedge \bigwedge_{i \in I_x} a_i\top$ for all $x \in \mathit{var}(t)$. Thus, for $x \in \mathit{var}(t)$ and $i \in I_x$, $P \vdash_{ws} \sigma(x) \xrightarrow{a_i} t_i$ for some $t_i \in T(\Sigma)$, and for $x \in \mathit{var}(t)$ and $j \in J_x$, $P \vdash_{ws} \sigma(x) \stackrel{b_j}{\not\longrightarrow}$. Let σ' be a closed substitution with $\begin{array}{ll}\sigma'(x) = \sigma(x) & \text{for } x \in \mathit{var}(t)\\ \sigma'(y_i) = t_i & \text{for } i \in I_x \text{ and } x \in \mathit{var}(t).\end{array}$ Here we use that the variables x and y_i are all different. Now $P \vdash_{ws} \sigma'(x) \xrightarrow{a_i} \sigma'(y_i)$ for $x \in \mathit{var}(t)$ and $i \in I_x$, and $P \vdash_{ws} \sigma'(x) \stackrel{b_j}{\not\longrightarrow}$ for $x \in \mathit{var}(t)$ and $j \in J_x$. So by Lemma 13 $P \vdash_{ws} \sigma'(t) \stackrel{a}{\not\longrightarrow}$, which implies $\sigma(t) = \sigma'(t) \models \widetilde{a}$.

- Suppose $\sigma(t) \models a\varphi$. Then by Definition 14 there is a $t' \in T(\Sigma)$ with $P \vdash_{ws} \sigma(t) \xrightarrow{a} t'$ and $t' \models \varphi$. Thus, by Lemma 13 there must be a P-ruloid $\frac{H}{t \xrightarrow{a} u}$ and a closed substitution σ' with $P \vdash_{ws} \sigma'(H)$, $\sigma'(t) = \sigma(t)$, i.e. $\sigma'(x) = \sigma(x)$ for $x \in \mathit{var}(t)$, and $\sigma'(u) = t'$. Since $\sigma'(u) \models \varphi$, the induction hypothesis can be applied, and there must be a $\chi \in u^{-1}(\varphi)$ such that $\sigma'(z) \models \chi(z)$ for all $z \in \mathit{var}(u)$. Furthermore $\sigma'(z) \models \chi(z) \cong \top$ for all $z \notin \mathit{var}(u)$. Now define ψ as indicated in Definition 31. By definition, $\psi \in t^{-1}(a\varphi)$. Let $x \in \mathit{var}(t)$. For $(x \xrightarrow{c} y) \in H$ one has $P \vdash_{ws} \sigma'(x) \xrightarrow{c} \sigma'(y) \models \chi(y)$, so $\sigma'(x) \models c\chi(y)$. Moreover, for $(x \stackrel{b}{\not\longrightarrow}) \in H$ one has $P \vdash_{ws} \sigma'(x) \stackrel{b}{\not\longrightarrow}$, so $\sigma'(x) \models \widetilde{b}$. It follows that $\sigma(x) = \sigma'(x) \models \psi(x)$.

  Now suppose that there is a $\psi \in t^{-1}(a\varphi)$ such that $\sigma(x) \models \psi(x)$ for all $x \in \mathit{var}(t)$. This means that there is a P-ruloid $\frac{\{x \xrightarrow{a_i} y_i \mid i \in I_x,\ x \in \mathit{var}(t)\} \cup \{x \stackrel{b_j}{\not\longrightarrow} \mid j \in J_x,\ x \in \mathit{var}(t)\}}{t \xrightarrow{a} u}$ and a parameterized observation $\chi \in u^{-1}(\varphi)$ such that $\sigma(x) \models \chi(x) \wedge \bigwedge_{j\in J_x} \widetilde{b_j} \wedge \bigwedge_{i\in I_x} a_i\chi(y_i)$ for all $x \in var(t)$. It follows that, for $x \in var(t)$ and $i \in I_x$, $P \vdash_{ws} \sigma(x) \xrightarrow{a_i} t_i$ for some $t_i \in T(\Sigma)$ with $t_i \models \chi(y_i)$. Let σ' be a closed substitution with $\begin{array}{ll}\sigma'(x) = \sigma(x) & \text{for } x \in var(t) \\ \sigma'(y_i) = t_i & \text{for } i \in I_x \text{ and } x \in var(t).\end{array}$ Here we use that the variables x and y_i are all different. Now $\sigma'(z) \models \chi(z)$ for $z \in var(u)$, using that u contains only variables that occur in the premises of the ruloid. Thus the induction hypothesis can be applied, and $\sigma'(u) \models \varphi$. Moreover, $P \vdash_{ws} \sigma'(x) \xrightarrow{a_i} \sigma'(y_i)$ for $x \in var(t)$ and $i \in I_x$, and $P \vdash_{ws} \sigma'(x) \stackrel{b_j}{\not\longrightarrow}$ for $x \in var(t)$ and $j \in J_x$. So by Lemma 13 $P \vdash_{ws} \sigma'(t) \xrightarrow{a} \sigma'(u)$, which implies $\sigma(t) = \sigma'(t) \models a\varphi$.

- $\sigma(t) \models \bigwedge_{i\in I} \varphi_i \ \Leftrightarrow\ \forall i \in I : \sigma(t) \models \varphi_i \ \Leftrightarrow\ \forall i \in I\ \exists \psi_i \in t^{-1}(\varphi_i)\ \forall x \in var(t) : \sigma(x) \models \psi_i(x) \ \Leftrightarrow$ $\Leftrightarrow\ \exists \psi \in t^{-1}(\bigwedge_{i\in I} \varphi_i)\ \forall x \in var(t) : \sigma(x) \models \psi(x)$. □

Example 5 Let $A = \{a, b\}$ and let Σ consist of the constant c and the unary function symbol f. Let R be:

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

Suppose $\psi \in f(f(x))^{-1}(ba\top)$. The only P-ruloid with a conclusion of the form $f(f(x)) \xrightarrow{b} \_$ is $\dfrac{x \xrightarrow{b} y}{f(f(x)) \xrightarrow{b} f(y)}$. So $\psi(x) = \chi(x) \wedge b\chi(y)$ with $\chi \in f(y)^{-1}(a\top)$. The only P-ruloid with a conclusion $f(y) \xrightarrow{a} \_$ is $\dfrac{y \xrightarrow{b} z}{f(y) \xrightarrow{a} f(z)}$. So $\chi(y) = \chi'(y) \wedge b\chi'(z)$ with $\chi' \in f(z)^{-1}(\top)$. Since $\chi'(y) = \chi'(z) = \top$ we have $\chi(y) \cong b\top$. Moreover $x \notin var(f(y))$ implies $\chi(x) \cong \top$. Hence $\psi(x) \cong bb\top$.

By Lemma 14 a closed term $f(f(u))$ can execute a b followed by an a iff the closed term u can execute two consecutive b's.

In order to arrive at the desired precongruence results we need to know that if a standard TSS is in the desired format N, and φ is a potential N-observation of a closed term $\sigma(t)$, then the observations of $\sigma(x)$ for $x \in var(t)$ that determine whether or not φ is an N-observation of $\sigma(t)$ are also N-observations. This is established in the following two lemmas. In fact, our formats have been found by investigating what was needed to make these lemmas hold.

The work is divided over two lemmas. Lemma 15 deals with those variables of t that are floating in the P-ruloids with source t. As the proof inductively refers to terms that can be thought of as successors of t after performing a number of actions, and as these terms may contain variables y representing successors of the arguments of t after performing several actions, the observations employed should be the ones from Definition 6. Lemma 16 extends this to arbitrary variables. As non-floating variables represent processes in their initial state only, that lemma may use the richer language of observations employed in Definition 7, which is much less cumbersome. This enables the absence of restrictions on non-floating variables in the definitions of the formats.

Lemma 15 Let $P = (\Sigma, R)$ be a standard TSS in ready simulation format, and let Λ be an unary predicate on arguments of function symbols in Σ. Let $t \in \mathbb{T}(\Sigma)$, $\varphi \in \mathbb{O}_{RS}$, $\psi \in t_P^{-1}(\varphi)$ and $x \in var(t)$, such that x occurs only once in t, and at a Λ-liquid position.

- If the rules in R^+ are Λ-ready trace safe and $\varphi \in \mathbb{O}_{RT}$ then $\psi(x) \in \mathbb{O}_{RT}$.

- If the rules in R^+ are Λ-readiness safe and $\varphi \in \mathbb{O}_R$ then $\psi(x) \in \mathbb{O}_R$.
- If the rules in R^+ are Λ-failure trace safe and $\varphi \in \mathbb{O}_{FT}$ then $\psi(x) \in \mathbb{O}_{FT}$.
- If the rules in R^+ are Λ-failure trace safe and $\varphi \in \mathbb{O}_F$ then $\psi(x) \in \mathbb{O}_F$.

Proof: Let $P = (\Sigma, R)$ be a standard TSS in ready simulation format, $t \in \mathbb{T}(\Sigma)$ and $x \in var(t)$. Note that if $\psi \in t^{-1}(\widetilde{a})$ then $\psi(x)$ has the form $\bigwedge_{j\in J} \widetilde{b_j} \wedge \bigwedge_{k\in K} c_k\top$. Thus if $\psi \in t^{-1}(\bigwedge_{i\in I} \widetilde{a_i})$ then $\psi(x)$ has the form $\bigwedge_{i\in I}(\bigwedge_{j\in J_i} \widetilde{b_{ij}} \wedge \bigwedge_{k\in K_i} c_{ik}\top)$. The latter formula is equivalent to one of the form $\bigwedge_{j\in J'} \widetilde{b'_j} \wedge \bigwedge_{k\in K'} c'_k\top$.

Furthermore note that if $\psi \in t^{-1}(a\top)$ then $\psi(x)$ has the form $\top \wedge \bigwedge_{j\in J} \widetilde{b_j} \wedge \bigwedge_{k\in K} c_j\top \cong \bigwedge_{j\in J} \widetilde{b_j} \wedge \bigwedge_{k\in K} c_k\top$. Here it is used that if $\chi \in u^{-1}(\top)$ for some $u \in \mathbb{T}(\Sigma)$ then $\chi(x) = \top$. Thus if $\psi \in t^{-1}(\bigwedge_{i\in I} a_i\top)$ then $\psi(x)$ is again equivalent to a formula of the form $\bigwedge_{j\in J'} \widetilde{b'_j} \wedge \bigwedge_{k\in K'} c'_k\top$.

Combining these observations it follows that if $\psi \in t^{-1}(\bigwedge_{i\in I} \widetilde{a_i} \wedge \bigwedge_{j\in J} b_j\top)$ then also $\psi(x)$ is equivalent to a formula of the form above.

- Let the rules in R^+ be Λ-ready trace safe and $\varphi \in \mathbb{O}_{RT}$. We apply structural induction on φ. Take $t \in \mathbb{T}(t)$, $\psi \in t^{-1}(\varphi)$ and $x \in var(t)$, such that x occurs only once in t, and at a Λ-liquid position.
  - In case $\varphi = \top$ we have $\psi(x) = \top \in \mathbb{O}_{RT}$.
  - Let $\varphi = \bigwedge_{i\in I} \widetilde{a_i} \wedge \bigwedge_{j\in J} b_j\top \wedge \varphi'$ with $\varphi' \in \mathbb{O}_{RT}$. Then $\psi(x) \cong \bigwedge_{k\in K} \widetilde{c_k} \wedge \bigwedge_{\ell\in L} d_\ell\top \wedge \psi'(x)$, where $\psi' \in t^{-1}(\varphi')$. By induction $\psi'(x) \in \mathbb{O}_{RT}$, and hence also $\psi(x) \in \mathbb{O}_{RT}$.
  - Let $\varphi = a\varphi'$ with $\varphi' \in \mathbb{O}_{RT}$. Then there are a P-ruloid $\frac{H}{t \xrightarrow{a} u}$ and $\chi \in u^{-1}(\varphi')$ such that
  $$\psi(x) = \chi(x) \wedge \bigwedge_{(x \stackrel{b}{\nrightarrow}) \in H} \widetilde{b} \wedge \bigwedge_{(x \xrightarrow{c} y) \in H} c\chi(y).$$
  By Lemma 2, x is propagated at most once in $\frac{H}{t \xrightarrow{a} u}$, and only at a Λ-liquid position. This implies that of the set of variables $W = \{x\} \cup \{y \mid \exists c : (x \xrightarrow{c} y) \in H\}$ at most one member, say z, occurs in $var(u)$. Since z is Λ-floating in $\frac{H}{t \xrightarrow{a} u}$ and occurs at most once in t, Lemma 2 moreover guarantees that z occurs only once in u, and at a Λ-liquid position. By induction we obtain $\chi(z) \in \mathbb{O}_{RT}$. For all other variables $w \in W$ we have $w \notin var(u)$ and so $\chi(w) \cong \top$. Thus $\psi(x)$ is of the form $\bigwedge_{j\in J} \widetilde{b_j} \wedge \bigwedge_{k\in K} c_k\top$ or $\chi(z) \wedge \bigwedge_{j\in J} \widetilde{b_j} \wedge \bigwedge_{k\in K} c_k\top$ or $\bigwedge_{j\in J} \widetilde{b_j} \wedge \bigwedge_{k\in K} c_k\top \wedge c\chi(z)$ with $\chi(z) \in \mathbb{O}_{RT}$. In all three cases $\psi(x) \in \mathbb{O}_{RT}$.
- Let the rules in R^+ be Λ-readiness safe and $\varphi \in \mathbb{O}_R$. We apply structural induction on φ. Take $t \in \mathbb{T}(t)$, $\psi \in t^{-1}(\varphi)$ and $x \in var(t)$, such that x occurs only once in t, and at a Λ-liquid position.
  - In case $\varphi = \top$ we have $\psi(x) = \top \in \mathbb{O}_R$.
  - Let $\varphi = \bigwedge_{i\in I} \widetilde{a_i} \wedge \bigwedge_{j\in J} b_j\top$. Then $\psi(x) \cong \bigwedge_{k\in K} \widetilde{c_k} \wedge \bigwedge_{\ell\in L} d_\ell\top \in \mathbb{O}_R$.
  - Let $\varphi = a\varphi'$ with $\varphi' \in \mathbb{O}_R$. Then there are a P-ruloid $\frac{H}{t \xrightarrow{a} u}$ and $\chi \in u^{-1}(\varphi')$ such that
  $$\psi(x) = \chi(x) \wedge \bigwedge_{(x \stackrel{b}{\nrightarrow}) \in H} \widetilde{b} \wedge \bigwedge_{(x \xrightarrow{c} y) \in H} c\chi(y).$$
  By Lemma 2, x is propagated at most once in $\frac{H}{t \xrightarrow{a} u}$, and only at a Λ-liquid position. Moreover, by Lemma 3, x is not both propagated and polled in $\frac{H}{t \xrightarrow{a} u}$. We consider three cases.

  * Suppose $x \in var(u)$. Then x is propagated, so it occurs only once in u, and at a Λ-liquid position. By induction $\chi(x) \in \mathbb{O}_R$. Furthermore, H has no premises of the form $x \stackrel{b}{\nrightarrow}$ or $x \xrightarrow{c} y$. Hence $\psi(x) \cong \chi(x) \in \mathbb{O}_R$.
  * Suppose x is propagated, but does not occur in u. Then $\chi(x) \cong \top$. Furthermore, H contains no premises of the form $x \stackrel{b}{\nrightarrow}$ and exactly one of the form $x \xrightarrow{c} y$, where y occurs in u. So $\psi(x) \cong c\chi(y)$. As y is Λ-floating in $\frac{H}{t \xrightarrow{a} u}$ and does not occur in t, Lemma 2 guarantees that y occurs only once in u, and at a Λ-liquid position. By induction $\chi(y) \in \mathbb{O}_R$, so $\psi(x) \in \mathbb{O}_R$.
  * Suppose x is not propagated. Then none of the variables y with $(x \xrightarrow{c} y) \in H$ for some $c \in A$ occurs in u. Hence for all those variables we have $\chi(y) \cong \top$. Moreover $x \notin var(u)$ so $\chi(x) \cong \top$. Thus $\psi(x)$ is of the form $\bigwedge_{j \in J} \widetilde{b_j} \wedge \bigwedge_{k \in K} c_k \top \in \mathbb{O}_R$.

- Let the rules in R^+ be Λ-failure trace safe and $\varphi \in \mathbb{O}_{FT}$. We apply structural induction on φ. Take $t \in \mathbb{T}(t)$, $\psi \in t^{-1}(\varphi)$ and $x \in var(t)$, such that x occurs only once in t, and at a Λ-liquid position.
  - In case $\varphi = \top$ we have $\psi(x) = \top \in \mathbb{O}_{FT}$.
  - Let $\varphi = \bigwedge_{i \in I} \widetilde{a_i} \wedge \varphi'$ with $\varphi' \in \mathbb{O}_{FT}$. Then $\psi(x) = \bigwedge_{i \in I} \psi_i(x) \wedge \psi'(x)$ where $\psi_i(x) \in t^{-1}(\widetilde{a_i})$ for $i \in I$ and $\psi'(x) \in t^{-1}(\varphi')$. For $i \in I$ there is a P-ruloid $\frac{H}{t \stackrel{a_i}{\nrightarrow}}$ such that

$$\psi_i(x) = \bigwedge_{(x \stackrel{b}{\nrightarrow}) \in H} \widetilde{b} \;\wedge \bigwedge_{(x \xrightarrow{c} y) \in H} c\top .$$

    By Lemma 5, H has no premises of the form $x \xrightarrow{c} y$. Therefore $\psi_i(x) \cong \bigwedge_{j \in J} \widetilde{b_j}$. Furthermore, by induction $\psi'(x) \in \mathbb{O}_{FT}$, so $\psi(x) = \bigwedge_{i \in I} \psi_i(x) \wedge \psi'(x) \in \mathbb{O}_{FT}$.
  - Let $\varphi = a\varphi'$ with $\varphi' \in \mathbb{O}_{FT}$. Then there are a P-ruloid $\frac{H}{t \xrightarrow{a} u}$ and $\chi \in u^{-1}(\varphi')$ such that

$$\psi(x) = \chi(x) \wedge \bigwedge_{(x \stackrel{b}{\nrightarrow}) \in H} \widetilde{b} \;\wedge \bigwedge_{(x \xrightarrow{c} y) \in H} c\chi(y).$$

    Using Lemmas 2 and 3, the same case distinction as in the readiness case applies, and in the first two cases we find $\psi(x) \in \mathbb{O}_{FT}$. Consider the third case, in which x is not propagated. Then $x \notin var(u)$ so $\chi(x) \cong \top$. By Lemma 4, x is polled at most once in $\frac{H}{t \xrightarrow{a} u}$, and in a positive premise. Hence H contains no literals of the form $x \stackrel{b}{\nrightarrow}$, and no more than one literal $x \xrightarrow{c} y$. If there is such a literal, y does not occur in u, so $\chi(y) \cong \top$. Hence $\psi(x) \cong \top$ or $\psi(x) \cong c\top$. In both cases $\psi(x) \in \mathbb{O}_{FT}$.
- The proof of the last statement (with $\varphi \in \mathbb{O}_F$) is a trivial simplification of the previous one. This proof is left to the reader. □

Lemma 16 Let $P = (\Sigma, R)$ be a standard TSS in ready simulation format, $t \in \mathbb{T}(\Sigma)$, $\varphi \in \mathbb{O}_{RS}$, $\psi \in t_P^{-1}(\varphi)$ and $x \in var(t)$. Then $\psi(x) \in \mathbb{O}^{\wedge}_{RS}$ and the following hold:

- If P is in ready trace format and $\varphi \in \mathbb{O}^{\wedge}_{RT}$ then $\psi(x) \in \mathbb{O}^{\wedge}_{RT}$.
- If P is in readiness format and $\varphi \in \mathbb{O}^{\wedge}_{R}$ then $\psi(x) \in \mathbb{O}^{\wedge}_{R}$.
- If P is in failure trace format and $\varphi \in \mathbb{O}^{\wedge}_{FT}$ then $\psi(x) \in \mathbb{O}^{\wedge}_{FT}$.
- If P is in failure trace format and $\varphi \in \mathbb{O}^{\wedge}_{F}$ then $\psi(x) \in \mathbb{O}^{\wedge}_{F}$.

Proof: That $\psi(x) \in \mathbb{O}_{RS} = \mathbb{O}^{\wedge}_{RS}$ is immediate from the definitions.

- Let P be in ready trace format and $\varphi \in \mathbb{O}_{RT}$. We apply structural induction on φ. Take $t \in \mathbb{T}(t)$, $\psi \in t^{-1}(\varphi)$ and $x \in var(t)$.
  - In case $\varphi = \top$ we have $\psi(x) = \top \in \mathbb{O}_{RT} \subseteq \mathbb{O}^{\wedge}_{RT}$.
  - Let $\varphi = \bigwedge_{i\in I} \widetilde{a_i} \wedge \bigwedge_{j\in J} b_j\top \wedge \varphi'$ with $\varphi' \in \mathbb{O}_{RT}$. Then $\psi(x) \cong \bigwedge_{k\in K} \widetilde{c_k} \wedge \bigwedge_{\ell\in L} d_\ell\top \wedge \psi'(x)$, where $\psi' \in t^{-1}(\varphi')$. Clearly $\widetilde{c_k} \in \mathbb{O}_{RT}$ for $k \in K$ and $d_\ell\top \in \mathbb{O}_{RT}$ for $\ell \in L$. By induction $\psi'(x) \in \mathbb{O}^{\wedge}_{RT}$, and hence also $\psi(x) \in \mathbb{O}^{\wedge}_{RT}$.
  - Let $\varphi = a\varphi'$ with $\varphi' \in \mathbb{O}_{RT}$. Then there are a P-ruloid $\frac{H}{t \xrightarrow{a} u}$ and $\chi \in u^{-1}(\varphi')$ such that
  $$\psi(x) = \chi(x) \wedge \bigwedge_{(x \xrightarrow{b}\!\!\!\!/\,)\in H} \widetilde{b} \;\wedge \bigwedge_{(x \xrightarrow{c} y)\in H} c\chi(y).$$
  As P is in ready trace format, by Corollary 3 so is P^+, so the rules in R^+ are Λ-ready trace safe for some unary predicate on arguments of function symbols Λ. The variables y with $(x \xrightarrow{a} y) \in H$ are Λ-floating in $\frac{H}{t \xrightarrow{a} u}$ and do not occur in t. Thus, by Lemma 2, each of them occurs at most once in u, and at a Λ-liquid position. In case y does not occur in u we have $\chi(y) \cong \top \in \mathbb{O}_{RT}$; otherwise Lemma 15 yields $\chi(y) \in \mathbb{O}_{RT}$. Hence $c\chi(y) \in \mathbb{O}_{RT}$ for each $(x \xrightarrow{c} y) \in H$. Clearly $\bigwedge_{(x \xrightarrow{b}\!\!\!\!/\,)\in H} \widetilde{b} \in \mathbb{O}_{RT}$, and by induction $\chi(x) \in \mathbb{O}^{\wedge}_{RT}$. Therefore $\psi(x) \in \mathbb{O}^{\wedge}_{RT}$.
  - Let $\varphi = \bigwedge_{i\in I} \varphi_i$ with $\varphi_i \in \mathbb{O}_{RT}$ for $i \in I$. Then $\psi(x) = \bigwedge_{i\in I} \psi_i(x)$ with $\psi_i(x) \in t^{-1}(\varphi_i)$ for $i \in I$. By induction (using the three cases treated above) $\psi_i(x) \in \mathbb{O}^{\wedge}_{RT}$ for $i \in I$. Therefore $\psi(x) \in \mathbb{O}^{\wedge}_{RT}$.
- The proofs of the remaining three statements proceed in exactly the same way. □

Now we are ready to prove Theorems 4–7. In the light of Definition 15 these theorems can be reformulated as in the following corollary, where N ranges over {ready simulation, ready trace, readiness, failure trace, failure} and *failure format* means failure trace format.

Corollary 5 (Precongruence) Let $P = (\Sigma, R)$ be a standard TSS in N format, $t \in \mathbb{T}(\Sigma)$ and σ, σ' closed substitutions. If $\sigma(x) \sqsubseteq^P_N \sigma'(x)$ for $x \in var(t)$, then $\sigma(t) \sqsubseteq^P_N \sigma'(t)$.

Proof: By Corollary 1 we have to show that $\mathcal{O}^{\wedge}_N(\sigma(x)) \subseteq \mathcal{O}^{\wedge}_N(\sigma'(x))$ for $x \in var(t)$ implies $\mathcal{O}^{\wedge}_N(\sigma(t)) \subseteq \mathcal{O}^{\wedge}_N(\sigma'(t))$. Suppose $\mathcal{O}^{\wedge}_N(\sigma(x)) \subseteq \mathcal{O}^{\wedge}_N(\sigma'(x))$ for $x \in var(t)$. Let $\varphi \in \mathcal{O}^{\wedge}_N(\sigma(t))$, i.e. $\varphi \in \mathbb{O}^{\wedge}_N$ and $\sigma(t) \models \varphi$. By Lemma 14 $\exists \psi \in t^{-1}(\varphi)\ \forall x \in var(t) : \sigma(x) \models \psi(x)$. By Lemma 16 $\forall x \in var(t) : \psi(x) \in \mathbb{O}^{\wedge}_N$. Thus $\forall x \in var(t) : \psi(x) \in \mathcal{O}^{\wedge}_N(\sigma(x)) \subseteq \mathcal{O}^{\wedge}_N(\sigma'(x))$. Hence $\forall x \in var(t) : \sigma'(x) \models \psi(x)$. So by Lemma 14 $\sigma'(t) \models \varphi$. It follows that $\varphi \in \mathcal{O}^{\wedge}_N(\sigma'(t))$, which had to be proved. □

9 Counterexamples

This section presents a string of counterexamples of complete standard TSSs in ntyft/ntyxt format, to show that the syntactic restrictions of our precongruence formats are essential. In [24] a series of counterexamples can be found showing that the syntactic restrictions of the ntyft/ntyxt format are essential as well.

9.1 Basic process algebra

The examples in this section assume basic process algebra [4]. We assume a collection *Act* of constants, called *atomic actions*, representing indivisible behaviour, and two special constants: the *deadlock* δ does not display any behaviour, while the *empty process* ε [40] terminates successfully. Basic process algebra moreover includes function symbols $\_+\_$ and $\_\cdot\_$ of arity two, called *alternative composition* and *sequential composition*, respectively. Intuitively, $t_1 + t_2$ executes either t_1 or t_2, while $t_1 \cdot t_2$ first executes t_1 and upon successful termination executes t_2. These intuitions are made precise by means of the standard transition rules for BPA$_{\delta\varepsilon}$ presented in Table 1, where the label v ranges over the set *Act* of atomic actions together with a special label $\surd$, representing successful termination. Note that this TSS is positive and in ready simulation format. It is not hard to check that the TSS is in failure trace format (and so by default in ready trace and readiness format), if we take at least the first argument of sequential composition to be liquid. In particular, in the first rule for sequential composition the floating variable y occurs in a liquid argument of the target, and in the second rule for sequential composition the floating variable x_1 is polled only once and not propagated.

$$v \xrightarrow{v} \varepsilon \quad (v \neq \surd) \qquad\qquad \varepsilon \xrightarrow{\surd} \delta$$

$$\frac{x_1 \xrightarrow{v} y}{x_1 + x_2 \xrightarrow{v} y} \qquad\qquad \frac{x_2 \xrightarrow{v} y}{x_1 + x_2 \xrightarrow{v} y}$$

$$\frac{x_1 \xrightarrow{v} y}{x_1 \cdot x_2 \xrightarrow{v} y \cdot x_2} \quad (v \neq \surd) \qquad \frac{x_1 \xrightarrow{\surd} y_1 \quad x_2 \xrightarrow{v} y_2}{x_1 \cdot x_2 \xrightarrow{v} y_2}$$

Table 1: Transition rules for BPA$_{\delta\varepsilon}$

Terms $t_1 \cdot t_2$ are abbreviated to $t_1 t_2$. Brackets are used for disambiguation only, assuming associativity of $+$ and $\cdot$, and letting $\cdot$ bind stronger than $+$. In the remainder of this section we assume that $Act = \{a, b, c, d\}$. Moreover, we assume unary function symbols f and h and a binary function symbol g.

9.2 Lookahead

The following counterexample shows that the ready simulation format (and its more restrictive analogues) cannot allow lookahead.

Example 6 We extend BPA$_{\delta\varepsilon}$ with the following rule, containing lookahead:

$$\frac{x \xrightarrow{b} y_1 \quad y_1 \xrightarrow{c} y_2}{f(x) \xrightarrow{a} \delta}$$

It is easy to see that $bd \sqsubseteq_{RS} bc + bd$ (so *a fortiori* $bd \sqsubseteq_N bc + bd$ for $N \in \{RT, R, FT, F\}$). The empty trace is a completed trace of $f(bd)$ but not of $f(bc + bd)$ (as $f(bc + bd) \xrightarrow{a} \delta$). Hence, $f(bd) \not\sqsubseteq_{CT} f(bc + bd)$ (and *a fortiori* $f(bd) \not\sqsubseteq_N f(bc + bd)$ for $N \in \{RS, RT, R, FT, F\}$).

9.3 Multiple propagations

The following counterexample shows that the ready trace format (and its more restrictive analogues) cannot allow a liquid argument of the source to be propagated more than once in the left-hand sides of the positive premises.

Example 7 Let the arguments of f and g be liquid. We extend $\mathrm{BPA}_{\delta\varepsilon}$ with the rules:

$$\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{a} f(y)} \qquad \frac{x \xrightarrow{b} y_1 \quad x \xrightarrow{b} y_2}{f(x) \xrightarrow{b} g(y_1, y_2)} \qquad \frac{x_1 \xrightarrow{c} y_1 \quad x_2 \xrightarrow{d} y_2}{g(x_1, x_2) \xrightarrow{d} \delta}$$

In the second rule, the liquid argument x of the source is propagated in the left-hand sides of both premises.

It is easy to see that $a(bc + bd) \sqsubseteq_{RT} abc + abd$ (so *a fortiori* $a(bc + bd) \sqsubseteq_N abc + abd$ for $N \in \{R, FT, F\}$). Note that abd is a trace of $f(a(bc + bd))$ (as $f(a(bc + bd)) \xrightarrow{a} f(bc + bd) \xrightarrow{b} g(c, d) \xrightarrow{d} \delta$), but not of $f(abc + abd)$. Hence, $f(a(bc + bd)) \not\sqsubseteq_T f(abc + abd)$ (and *a fortiori* $f(a(bc+bd)) \not\sqsubseteq_N f(abc+abd)$ for $N \in \{RT, R, FT, F\}$).

A similar example can be given to show that the ready trace format cannot allow a liquid argument of the source or a right-hand side of a positive premise to occur more than once in the target. Likewise, an example can be given to show that the ready trace format cannot allow a liquid argument of the source to be propagated in the left-hand side of a positive premise and at the same time to occur in the target.

9.4 Propagation at a non-liquid position

If in the example above the argument of f were defined to be frozen, then in the first rule the right-hand side y of the premise would occur in a non-liquid position in the target. This shows that the ready trace format cannot allow right-hand sides of positive premises to occur at non-liquid positions in the target. Variants of Example 7 show that the ready trace format cannot allow liquid arguments of the source to be propagated at non-liquid positions either.

Example 8 Replace the second rule in Example 7 by the two rules

$$\frac{h(x) \xrightarrow{b} y}{f(x) \xrightarrow{b} y} \qquad \frac{x \xrightarrow{b} y_1 \quad x \xrightarrow{b} y_2}{h(x) \xrightarrow{b} g(y_1, y_2)}$$

Taking the arguments of f and g liquid, but that of h frozen, the resulting TSS sins against the ready trace format only in that in the first rule above the floating variable x is propagated at a non-liquid position, namely as argument of h. Clearly the same mishap as in Example 7 ensues. The same argument applies when replacing the second rule in Example 7 by the two rules

$$f(x) \xrightarrow{a} h(x) \qquad \frac{x \xrightarrow{b} y_1 \quad x \xrightarrow{b} y_2}{h(x) \xrightarrow{b} g(y_1, y_2)}$$

The examples above show that if a floating variable x is propagated in a term $f(x)$, then the argument of f should be classified as liquid. Furthermore, if x is propagated in a term $f(h(x))$, then *both* the argument of f and that of h should be classified as liquid. Namely, if only the argument of f would be liquid, a rule with conclusion $h(x) \xrightarrow{b} g(x,x)$ could have fatal consequences, and if only the argument of h would be liquid, a rule with conclusion $f(x) \xrightarrow{b} g(x,x)$ could be fatal. (It is left to the reader to fill in the details.) This justifies the definition of Λ-liquid in Section 4.

9.5 Propagation in combination with polling

The following counterexample shows that the readiness format cannot allow that a liquid argument of the source is both propagated and polled. (The TSS in this example is in a flawed congruence format for failure equivalence from [20].)

Example 9 Let the arguments of f and h be liquid. We extend $\mathrm{BPA}_{\delta\varepsilon}$ with the rules:

$$\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{a} f(y)} \qquad \frac{x \xrightarrow{b} y}{f(x) \xrightarrow{b} h(x)} \qquad \frac{x \xrightarrow{c} y}{h(x) \xrightarrow{c} h(y)} \qquad \frac{x \xrightarrow{d} y}{h(x) \xrightarrow{d} \delta}$$

In the second rule, the liquid argument x of the source is both propagated and polled.

It is easy to see that $a(b+cd)+ac$ and $a(b+c)+acd$ are readiness and failure equivalent (but not ready trace or failure trace equivalent). Note that $abcd$ is a trace of $f(a(b+cd)+ac)$ (as $f(a(b+cd)+ac) \xrightarrow{a} f(b+cd) \xrightarrow{b} h(b+cd) \xrightarrow{c} h(d) \xrightarrow{d} \delta$), but not of $f(a(b+c)+acd)$. Hence, $f(a(b+cd)+ac)$ and $f(a(b+c)+acd)$ are not even trace equivalent.

It is easy to see that $a(b+c)+ab+ac$ and $ab+ac$ are failure trace and failure equivalent (but not ready trace or readiness equivalent). Note that abc is a trace of $f(a(b+c)+ab+ac)$ (as $f(a(b+c)+ab+ac) \xrightarrow{a} f(b+c) \xrightarrow{b} h(b+c) \xrightarrow{c} h(\varepsilon)$), but not of $f(ab+ac)$. Hence, $f(a(b+c)+ab+ac)$ and $f(ab+ac)$ are not even trace equivalent.

9.6 Multiple pollings

The following counterexample shows that the failure trace format cannot allow that a liquid argument of the source is polled more than once.

Example 10 Let the argument of f be liquid. We extend $\mathrm{BPA}_{\delta\varepsilon}$ with the rules:

$$\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{a} f(y)} \qquad \frac{x \xrightarrow{b} y_1 \quad x \xrightarrow{c} y_2}{f(x) \xrightarrow{d} \delta}$$

In the second rule, the liquid argument x of the source is polled in the two positive premises.

We recall that $a(b+c)+ab+ac$ and $ab+ac$ are failure trace and failure equivalent. Note that ad is a trace of $f(a(b+c)+ab+ac)$ (as $f(a(b+c)+ab+ac) \xrightarrow{a} f(b+c) \xrightarrow{d} \delta$), but not of $f(ab+ac)$. Hence, $f(a(b+c)+ab+ac)$ and $f(ab+ac)$ are not even trace equivalent.

9.7 Polling at a non-liquid position

The following variant of Example 10 shows that the failure trace format cannot allow that a liquid argument of the source is polled at non-liquid positions.

Example 11 Let the argument of f be liquid and the argument of h be frozen. We extend $\mathrm{BPA}_{\delta\varepsilon}$ with the rules:

$$\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{a} f(y)} \qquad \frac{h(x) \xrightarrow{b} y}{f(x) \xrightarrow{b} \delta} \qquad \frac{x \xrightarrow{b} y_1 \quad x \xrightarrow{c} y_2}{h(x) \xrightarrow{d} \delta}$$

In the second rule, the liquid argument x of the source is polled at a non-liquid position. Clearly the same mishap as in Example 10 ensues.

9.8 Polling in a negative premise

The following counterexample shows that the failure trace format cannot allow that a liquid argument of the source is polled in a negative premise.

Example 12 Let the argument of f be liquid. We extend $\mathrm{BPA}_{\delta\varepsilon}$ with the rules:

$$\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{a} f(y)} \qquad \frac{x \stackrel{b}{\not\longrightarrow}}{f(x) \xrightarrow{d} \delta} \qquad \frac{x \stackrel{c}{\not\longrightarrow}}{f(x) \xrightarrow{d} \delta}$$

In the second and third rule, the liquid argument x of the source is polled in the negative premise.

We recall that $a(b+c)+ab+ac$ and $ab+ac$ are failure trace and failure equivalent. Note that a is a completed trace of $f(a(b+c)+ab+ac)$ (as $f(a(b+c)+ab+ac) \xrightarrow{a} f(b+c) \stackrel{d}{\not\longrightarrow}$), but not of $f(ab+ac)$. Hence, $f(a(b+c)+ab+ac)$ and $f(ab+ac)$ are not even completed trace equivalent.

10 Applications

This section contains some applications of our precongruence formats to TSSs from the literature.

10.1 Priority

Priority [2] is a unary function symbol that assumes an ordering on transition labels. The term $\Theta(t)$ executes the transitions of t, with the restriction that a transition $t' \xrightarrow{a} t_1$ only gives rise to a transition $\Theta(t') \xrightarrow{a} \Theta(t_1)$ if there does not exist a transition $t' \xrightarrow{b} t_2$ with $a < b$. This intuition is captured by the rule for the priority operator in Table 2, which is added to the rules for $\mathrm{BPA}_{\delta\varepsilon}$ in Table 1. The resulting standard TSS is in ready simulation format.

$$\boxed{\frac{x \xrightarrow{v} y \quad x \stackrel{w}{\not\longrightarrow} \text{ for } v < w}{\Theta(x) \xrightarrow{v} \Theta(y)}}$$

Table 2: Transition rule for priority

As the floating variable y in the rule for priority is propagated in $\Theta(y)$, the argument of Θ has to be liquid. The floating variable x is propagated only once (and trivially at a liquid position). Hence the TSS for $\mathrm{BPA}_{\delta\varepsilon}$ with priority is in ready trace format.

Corollary 6 The ready simulation and ready trace preorders are precongruences with respect to $\mathrm{BPA}_{\delta\varepsilon}$ with priority.

The TSS for $\mathrm{BPA}_{\delta\varepsilon}$ with priority is not in readiness format. Namely, in the case of a non-trivial ordering on transition labels, the floating variable x in the rule for priority is both propagated and polled. We show that the readiness, failure trace and failure preorders are not precongruences with respect to $\mathrm{BPA}_{\delta\varepsilon}$ with priority. Recall that $a(b+c)+acd$ and $a(b+cd)+ac$ are readiness equivalent. If the ordering consists of $c<b$, then acd is a trace of $\Theta(a(b+c)+acd)$ (as it can execute $\xrightarrow{a}\Theta(cd)\xrightarrow{c}\Theta(d)\xrightarrow{d}\Theta(\varepsilon)$), but not of $\Theta(a(b+cd)+ac)$. Hence, $\Theta(a(b+c)+acd)$ and $\Theta(a(b+cd)+ac)$ are not even trace equivalent.

Clearly $ab+a(c+d)+a(b+c+d)$ and $ab+a(c+d)$ are failure trace equivalent. If the ordering consists of $c<b$, then $\emptyset\, a\,\{c\}\, d\,\emptyset$ is a failure trace of $\Theta(ab+a(c+d)+a(b+c+d))$ (as it can execute $\xrightarrow{a}\Theta(b+c+d)\xrightarrow{d}\Theta(\varepsilon)$), but not of $\Theta(ab+a(c+d))$. Hence, $\Theta(ab+a(c+d)+a(b+c+d))$ and $\Theta(ab+a(c+d))$ are not failure trace equivalent.

10.2 Initial priority

Initial priority is a unary function symbol that assumes an ordering on transition labels. The term $\theta(t)$ executes the transitions of t, with the restriction that an initial transition $t\xrightarrow{a}t_1$ only gives rise to an initial transition $\theta(t)\xrightarrow{a}t_1$ if there does not exist an initial transition $t\xrightarrow{b}t_2$ with $a<b$. This intuition is captured by the rule for the initial priority operator in Table 3, which is added to the rules for $\mathrm{BPA}_{\delta\varepsilon}$ in Table 1. The resulting standard TSS is in ready simulation format.

$$\frac{x\xrightarrow{v}y \qquad x\overset{w}{\not\longrightarrow} \quad \text{for } v<w}{\theta(x)\xrightarrow{v}y}$$

Table 3: Transition rule for initial priority

We take the argument of initial priority to be frozen. Now the TSS for $\mathrm{BPA}_{\delta\varepsilon}$ with initial priority is in failure trace format. Note that in the rule for initial priority the variable x, which is polled and propagated, is non-floating.

Corollary 7 The ready simulation, ready trace, readiness, failure trace and failure preorders are precongruences with respect to $\mathrm{BPA}_{\delta\varepsilon}$ with initial priority.

In the case of a non-trivial ordering on transition labels, the rule for initial priority is outside de Simone's format, due to the presence of negative premises.

10.3 Binary Kleene star

The *binary Kleene star* $t_1{}^*t_2$ [26] repeatedly executes t_1 until it executes t_2. This operational behaviour is captured by the rules in Table 4, which are added to the rules for $\mathrm{BPA}_{\delta\varepsilon}$ in Table 1. The resulting positive TSS is in failure trace format if we take the first argument of sequential composition to be liquid and the first argument of the binary Kleene star to be frozen. Note that in the first rule for the binary Kleene star the floating variable y is propagated only once, at a liquid position, and not polled. Moreover in this rule the variable x_1, which is propagated twice, is non-floating.

$$\frac{x_1 \xrightarrow{v} y}{x_1{}^*x_2 \xrightarrow{v} y \cdot (x_1{}^*x_2)} \; (v \neq \surd) \qquad \frac{x_2 \xrightarrow{v} y}{x_1{}^*x_2 \xrightarrow{v} y}$$

Table 4: Transition rules for the binary Kleene star

Corollary 8 The n-nested simulation (for $n \geq 1$), ready simulation, ready trace, readiness, failure trace and failure preorders are precongruences w.r.t. $\mathrm{BPA}_{\delta\varepsilon}$ with the binary Kleene star.

It was noted in [1] that the second rule for the binary Kleene star does not fit the congruence format for ready trace equivalence from [20]. Namely, in this rule the variables x_1 and y in the target are connected by the premise. Neither does this rule fit de Simone's format, due to the fact that x_1 occurs in the left-hand side of the premise and in the target.

10.4 Sequencing

Sequencing $t_1;t_2$ executes t_1 until it can do no further transitions, after which it starts executing t_2. Basic process algebra with sequencing contains the atomic actions in Act, alternative composition and sequencing, while the deadlock and the empty process are fused to a single constant $\mathbf{0}$. The transition rules for $\mathrm{BPA}^{;}_{\mathbf{0}}$ are presented in Table 5, where v and w range over Act. This standard TSS is in ready simulation format. The rules for sequencing were taken from [7].

$$v \xrightarrow{v} \mathbf{0}$$

$$\frac{x_1 \xrightarrow{v} y}{x_1 + x_2 \xrightarrow{v} y} \qquad \frac{x_2 \xrightarrow{v} y}{x_1 + x_2 \xrightarrow{v} y}$$

$$\frac{x_1 \xrightarrow{v} y}{x_1;x_2 \xrightarrow{v} y;x_2} \qquad \frac{x_1 \stackrel{v}{\not\longrightarrow} \text{ for } v \in Act \quad x_2 \xrightarrow{w} y}{x_1;x_2 \xrightarrow{w} y}$$

Table 5: Transitions rules for $\mathrm{BPA}^{;}_{\mathbf{0}}$

As the floating variable y in the first rule for sequencing is propagated in $y;x_2$, the first argument of sequencing has to be liquid. In the first rule for sequencing the floating variable x_1 is propagated only once (and trivially at a liquid position), and not polled. In the second rule for sequencing the floating variable x_1 is not propagated, while the floating variable y is propagated only once (and trivially at a liquid position), and not polled. Hence the TSS for $\mathrm{BPA}^{;}_{\mathbf{0}}$ is in readiness format.

Corollary 9 The ready simulation, ready trace and readiness preorders are precongruences with respect to $\mathrm{BPA}^{;}_{\mathbf{0}}$.

The TSS for $\mathrm{BPA}^{;}_{\mathbf{0}}$ is not in failure trace format. Namely, in the second rule for sequencing the floating variable x_1 is polled in negative premises. We show that the failure preorder is not a precongruence with respect to $\mathrm{BPA}_{\mathbf{0}}^{;}$. Let sequencing bind stronger than alternative composition. Clearly $a;(b+c)+a;c+a$ and $a;(b+c)+a$ are failure equivalent. Note that $(a,\{b\})$ is a failure pair of $(a;(b+c)+a;c+a);b$ (as it can execute $\xrightarrow{a} \mathbf{0};c;b \stackrel{b}{\not\longrightarrow}$), but not of $(a;(b+c)+a);b$. Hence, $(a;(b+c)+a;c+a);b$ and $(a;(b+c)+a);b$ are not failure equivalent.

The failure trace preorder is a precongruence with respect to $\mathrm{BPA}_{\mathbf{0}}^{;}$. Namely $X_0a_1X_1\cdots a_nX_n$ is a failure trace of $p;q$ iff either $X_0a_1X_1\cdots a_{m-1}X_{m-1}a_m Act$ is a failure trace of p for some $m \leq n$ and $X_m a_{m+1}X_{m+1}\cdots a_nX_n$ is a failure trace of q, or $X_0a_1X_1\cdots a_nX_nb\emptyset$ is a failure trace of p for some $b \in Act$. Thus sequencing is an example of an operator from the literature that preserves failure traces but that lies outside the scope of the failure trace format.

10.5 Action refinement

The binary *action refinement* operator $t_1[a \to t_2]$ for atomic actions a, based on a similar operator in [22], replaces each a transition in t_1 by t_2. Its transition rules are presented in Table 6, which are added to the rules for $\mathrm{BPA}_{\mathbf{0}}^{;}$ in Table 5. If we take the first argument of sequencing and of action refinement to be liquid and the second argument of action refinement to be frozen, then this standard TSS is in readiness format. Note that in the second rule for action refinement the floating variable y_1 is propagated at a liquid position, while the variable x_2, which is propagated twice, is non-floating.

$$\frac{x_1 \xrightarrow{w} y}{x_1[v \to x_2] \xrightarrow{w} y[v \to x_2]} \quad (v \neq w) \qquad\qquad \frac{x_1 \xrightarrow{v} y_1 \quad x_2 \xrightarrow{w} y_2}{x_1[v \to x_2] \xrightarrow{w} y_2;(y_1[v \to x_2])}$$

Table 6: Transition rules for action refinement

Corollary 10 The ready simulation, ready trace and readiness preorders are precongruences with respect to $\mathrm{BPA}_{\mathbf{0}}^{;}$ with action refinement.

11 Partial traces

The proof technique developed in this paper can be generally applied to generate a precongruence format for a preorder from the observational definition of this preorder. As an example we sketch how this technique yields a precongruence format for the partial trace preorder. The details are left to the reader. We say that a TSS is in *partial trace format* if it is positive and in failure trace format.

Theorem 8 If a TSS is in partial trace format, then the partial trace preorder that it induces is a precongruence.

The outline of the proof of Theorem 8 is as follows. Assume a predicate Λ on arguments of function symbols over a signature Σ. We say that an ntytt rule is Λ-*partial trace safe* if it is Λ-failure trace safe and either its conclusion and premises are all positive, or they are all negative. In line with Lemmas 2-5 we obtain: if P is a TSS in decent ntyft format of which the transition rules are Λ-partial trace safe, then any ntytt rule irredundantly provable from P is Λ-partial trace safe. As in

Corollary 3 we obtain: if $P = (\Sigma, R)$ is a standard TSS in partial trace format, then $P^+ = (\Sigma, R^+)$ is in partial trace format and $P^+ \vdash \alpha \Leftrightarrow P \vdash_{ws} \alpha$. As in Lemma 15 we obtain: if the rules in R^+ are Λ-partial trace safe and $\varphi \in \mathbb{O}_T$ then $\psi(x) \in \mathbb{O}_T$. (Note that, as observations $\widetilde{a}$ do not occur in $\mathbb{O}_T$, the case $\psi \in t_P^{-1}(\widetilde{a})$ need not be considered.) As in Lemma 16 we obtain: if P is in partial trace format and $\varphi \in \mathbb{O}_T^\wedge$ then $\psi(x) \in \mathbb{O}_T^\wedge$. Finally, as in Corollary 5 we obtain: if P is a standard TSS in partial trace format and $\sigma(x) \sqsubseteq_T^P \sigma'(x)$ for $x \in var(t)$ then $\sigma(t) \sqsubseteq_T^P \sigma'(t)$. This immediately implies Theorem 8.

In the precongruence format for the partial trace preorder one could in principle allow lookahead. We leave it as an open question whether the partial trace format extended with lookahead is indeed a precongruence format for the partial trace preorder.

The following counterexample shows that in the case of the partial trace *preorder*, the restriction to positive TSSs in the partial trace format is essential.

Example 13 Let the argument of f be frozen. We extend $\mathrm{BPA}_{\delta\varepsilon}$ with the rule:

$$\frac{x \stackrel{b}{\not\longrightarrow}}{f(x) \stackrel{c}{\longrightarrow} \delta}$$

Clearly $a \sqsubseteq_T a + b$. Note that c is a trace of $f(a)$ (as $f(a) \stackrel{c}{\longrightarrow} \delta$), but not of $f(a+b)$. Hence, $f(a) \not\sqsubseteq_T f(a+b)$.

According to the following theorem, in the case of partial trace *equivalence* one can allow TSSs with negative premises.

Theorem 9 If a standard TSS is in failure trace format, then the partial trace equivalence that it induces is a congruence.

The outline of the proof of Theorem 9 is as follows. Let $\mathbb{O}_T^{\wedge\sim}$ consist of all formulas

$$\bigwedge_{i\in I} \varphi_i \wedge \bigwedge_{j\in J} \widetilde{a_j}$$

with $\varphi_i \in \mathbb{O}_T$ and $a_j \in A$. Let $O_T^{\wedge\sim}(p) := \{\varphi \in \mathbb{O}_T^{\wedge\sim} \mid p \models \varphi\}$. If $p \sqsubseteq_T q$ and $q \models \bigwedge_{j\in J} \widetilde{a_j}$ then $p \models \bigwedge_{j\in J} \widetilde{a_j}$. Thus $p =_T q$ iff $O_T^{\wedge\sim}(p) = O_T^{\wedge\sim}(q)$. As in Lemma 15 we obtain: if the rules in R^+ are Λ-failure trace safe and $\varphi \in \mathbb{O}_T$ then $\psi(x) \in \mathbb{O}_T$. As in Lemma 16 we obtain: if P is in failure trace format and $\varphi \in \mathbb{O}_T^{\wedge\sim}$ then $\psi(x) \in \mathbb{O}_T^{\wedge\sim}$. Finally, as in Corollary 5 we obtain: if P is a standard TSS in failure trace format and $\sigma(x) =_T^P \sigma'(x)$ for $x \in var(t)$ then $\sigma(t) =_T^P \sigma'(t)$. This immediately implies Theorem 9.

12 Conservative extension

Traditionally, papers on congruence formats also introduce syntactic restrictions on TSSs to ensure that one TSS is a *conservative extension* of another [10, 23, 24]. Here, we extend results on conservative extension from those three papers to incomplete TSSs, building on propositions obtained in earlier sections. However, since these propositions were proved for incomplete TSSs in ready simulation format only, our conservative extension result is restricted to that format.

Definition 32 Let $P_1 = (\Sigma_1, R_1)$ and $P_2 = (\Sigma_2, R_2)$ be standard TSSs with $\Sigma_1 \subseteq \Sigma_2$. P_2 is said to be a *conservative extension* of P_1 if $P_2 \vdash_{ws} \alpha \Leftrightarrow P_1 \vdash_{ws} \alpha$ for any closed Σ_2-literal α whose left-hand side is a term over Σ_1.

This definition is equivalent to the ones in [10, 23, 24], except that there only the case is considered where $R_1 \subseteq R_2$.

Lemma 17 Let $P_1 = (\Sigma_1, R_1)$ and $P_2 = (\Sigma_2, R_2)$ be TSSs with $\Sigma_1 \subseteq \Sigma_2$ and $R_1 \subseteq R_2$, such that each rule in R_1 is decent and each rule in $R_2 - R_1$ has a function symbol $f \in \Sigma_2 - \Sigma_1$ in its source. Then $P_2 \vdash_{\rm irr} \frac{H}{\alpha} \Leftrightarrow P_1 \vdash_{\rm irr} \frac{H}{\alpha}$ for any transition rule $\frac{H}{\alpha}$ over Σ_2 whose source is a term over Σ_1. Thus in particular $P_2 \vdash \alpha \Leftrightarrow P_1 \vdash \alpha$ for any Σ_2-literal α whose left-hand side is a term over Σ_1.

Proof: "$\Leftarrow$": Trivially, any proof of $\frac{H}{\alpha}$ from P_1 is also a proof of $\frac{H}{\alpha}$ from P_2.

"$\Rightarrow$": with induction on the structure of irredundant proofs. Let π be an irredundant proof of $\frac{H}{\alpha}$ from P_2. We show that π is also an irredundant proof of $\frac{H}{\alpha}$ from P_1. Let K be the set of labels of the nodes directly above the root node q of π (which is labelled with α). Then $\frac{K}{\alpha}$ is a substitution instance of a transition rule ρ in R_2. As the left-hand side of α is a term over Σ_1, the rule ρ must be in R_1. Thus ρ is decent, which implies that all variables occurring in the left-hand sides of its premises also occur in its source. Therefore, any function symbol occurring in the left-hand side of a literal in K also occurs in the left-hand side of α and hence belongs to Σ_1. For all $\beta \in K$ there is an irredundant proof π_β, which is a proper part of π, of a rule $\frac{H_\beta}{\beta}$ from P_2. We have $\bigcup_{\beta \in K} H_\beta = H$. Moreover, the left-hand side of β is a term over Σ_1. So, by induction, π_β is an irredundant proof of β from P_1 for all $\beta \in K$. Thus β is a Σ_1-literal. As ρ is decent, all variables occurring in its target also occur in its source or in one of its premises. Therefore, any function symbol occurring in the right-hand side of α also occurs in its left-hand side or in K, and hence belongs to Σ_1. Hence π is an irredundant proof of α from P_1. $\square$

Theorem 10 Let $P_1 = (\Sigma_1, R_1)$ and $P_2 = (\Sigma_2, R_2)$ be standard TSSs in decent ntyft format with $\Sigma_1 \subseteq \Sigma_2$ and $R_1 \subseteq R_2$, such that each rule in $R_2 - R$ has a function symbol $f \in \Sigma_2 - \Sigma_1$ in its source. Then P_2 is a conservative extension of P_1.

Proof: Let $P_1 = (\Sigma_1, R_1)$ and $P_2 = (\Sigma_2, R_2)$ be as stipulated. Let $P_1' = (\Sigma_1, R_1')$ and $P_2' = (\Sigma, R_2')$ be the TSSs in decent xynft format obtained from P_1 and P_2 as indicated in the proof of Proposition 4: R_1' (resp. R_2') consists of all decent xynft rules irredundantly provable from P_1 (resp. P_2). Now clearly $R_1' \subseteq R_2'$, and by Lemma 17 each rule in $R_2' - R_1'$ has a function symbol $f \in \Sigma_2 - \Sigma_1$ in its source. By means of a coordinated bijective renaming of the variables in R_1' and R_2' we obtain TSSs in decent uniform xynft format P_1'' and P_2'' that still satisfy the stipulations of Theorem 10. On these the construction of Definition 27 may be applied to obtain $P_1^+ = (\Sigma, R_1^+)$ and $P_2^+ = (\Sigma_2, R_2^+)$. By Definitions 26 and 27 we have $R_1^+ \subseteq R_2^+$ and each rule in $R_2^+ - R_1^+$ has a function symbol $f \in \Sigma_2 - \Sigma_1$ in its source. Thus, by Propositions 4, 5 and 6 and Lemma 17 we have $P_2 \vdash_{ws} \alpha \Leftrightarrow P_2' \vdash_{ws} \alpha \Leftrightarrow P_2' \vdash_s \alpha \Leftrightarrow P_2'' \vdash_s \alpha \Leftrightarrow P_2^+ \vdash \alpha \Leftrightarrow P_1^+ \vdash \alpha \Leftrightarrow \cdots \Leftrightarrow P_1 \vdash_{ws} \alpha$ for any closed Σ_2-literal α whose left-hand side is a term over Σ_1. $\square$

13 Possible extensions

In this paper we have presented precongruence formats for a range of preorders, which are *a fortiori* also congruence formats for the induced equivalences. We are not aware of any further plausible extensions of the precongruence formats (apart from conceptual extensions such as higher-orderness and syntactic sugar such as predicates and terms as transition labels).

It may be possible to formulate more liberal congruence formats for ready simulation equivalence and the decorated trace equivalences. Namely, one could in principle allow lookahead for frozen arguments of the source. To be more precise, one could (re)define that an occurrence of a variable in an ntytt rule is *propagated* if the occurrence is either in the target or in the left-hand side of a positive premise of which the right-hand side is polled or propagated. Furthermore, one could define that a positive premise in an ntytt rule has *lookahead* if its right-hand side occurs in the left-hand side of some premise in this rule. Assume a predicate Λ on arguments of function symbols. Then a standard ntytt rule is Λ-*ready simulation equivalence safe* if each Λ-floating variable is only propagated at Λ-liquid positions, and not in positive premises with lookahead. A standard ntytt rule is Λ-*ready trace equivalence safe* if moreover each Λ-floating variable is propagated at most once, at a Λ-liquid position. A standard ntytt rule is Λ-*readiness equivalence safe* if moreover each Λ-floating variable is not both propagated and polled. A standard ntytt rule is Λ-*failure trace equivalence safe* if moreover each Λ-floating variable is polled at most once, at a Λ-liquid position in a positive premise. A TSS is in N equivalence format (for $N \in \{\text{ready simulation}, \text{ready trace}, \text{readiness}, \text{failure trace}\}$) if it is in ntyft/ntyxt format and its rules are Λ-N equivalence safe with respect to some Λ.

We leave it as an open question whether these four formats are indeed congruence formats for ready simulation equivalence, ready trace equivalence, readiness equivalence, failure trace equivalence and failure equivalence. The proof technique employed in this paper uses the absence of lookahead in an essential way. In particular, as illustrated by Example 4, the construction of R^+ in Section 6 only works in the absence of lookahead. So the open question cannot be answered by a simple adaptation of the precongruence proofs in this paper. Example 6 shows that the *pre*congruence formats cannot allow lookahead.

References

[1] L. Aceto, W.J. Fokkink & A. Ingólfsdóttir (1998): *A menagerie of non-finitely based process semantics over BPA*: from ready simulation to completed traces.* Mathematical Structures in Computer Science 8(3), pp. 193–230.

[2] J.C.M. Baeten, J.A. Bergstra & J.W. Klop (1986): *Syntax and defining equations for an interrupt mechanism in process algebra.* Fundamenta Informaticae IX(2), pp. 127–168.

[3] J.C.M. Baeten, J.A. Bergstra & J.W. Klop (1987): *Ready-trace semantics for concrete process algebra with the priority operator.* The Computer Journal 30(6), pp. 498–506.

[4] J.A. Bergstra & J.W. Klop (1984): *Process algebra for synchronous communication.* Information and Control 60(1/3), pp. 109–137.

[5] J.A. Bergstra, J.W. Klop & E.-R. Olderog (1988): *Readies and failures in the algebra of communicating processes.* SIAM Journal on Computing 17(6), pp. 1134–1177.

[6] B. Bloom (1993): *Ready, set, go: structural operational semantics for linear-time process algebras.* Report TR 93-1372, Cornell University.

[7] B. Bloom (1994): *When is partial trace equivalence adequate?* Formal Aspects of Computing 6(3), pp. 317–338.

[8] B. Bloom, W.J. Fokkink & R.J. van Glabbeek (2000): *Precongruence formats for decorated trace preorders.* In *Proceedings 15th IEEE Symposium on Logic in Computer Science (LICS'2000)*, Santa Barbara, IEEE Computer Society Press, pp. 107–118.

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

[10] R.N. Bol & J.F. Groote (1996): *The meaning of negative premises in transition system specifications.* *Journal of the ACM* 43(5), pp. 863–914.

[11] S.D. Brookes, C.A.R. Hoare & A.W. Roscoe (1984): *A theory of communicating sequential processes.* *Journal of the ACM* 31(3), pp. 560–599.

[12] K.L. Clark (1978): *Negation as failure.* In H. Gallaire and J. Minker, eds., *Logic and Databases.* Plenum Press, New York.

[13] R. De Nicola & M.C.B. Hennessy (1984): *Testing equivalences for processes.* *Theoretical Computer Science* 34(1/2), pp. 83–133.

[14] F. Fages (1991): *A new fixpoint semantics for general logic programs compared with the well-founded and the stable model semantics.* *New Generation Computing* 9(3/4), pp. 425–443.

[15] W.J. Fokkink (2000): *Language preorder as a precongruence.* *Theoretical Computer Science,* 243(1-2), pp. 391–408.

[16] W.J. Fokkink (2000): *Rooted branching bisimulation as a congruence.* *Journal of Computer and System Sciences* 60(1), pp. 13–37.

[17] W.J. Fokkink & R.J. van Glabbeek (1996): *Ntyft/ntyxt rules reduce to ntree rules.* *Information and Computation* 126(1), pp. 1–10.

[18] A. van Gelder, K. Ross & J.S. Schlipf (1991): *The well-founded semantics for general logic programs.* *Journal of the ACM* 38(3), pp. 620–650.

[19] R.J. van Glabbeek (2001): *The linear time – branching time spectrum I; the semantics of concrete, sequential processes.* In J.A. Bergstra, A. Ponse & S.A. Smolka, editors: *Handbook of Process Algebra*, chapter 1, Elsevier, pp. 3–99. Available at `http://boole.stanford.edu/-pub/spectrum1.ps.gz`. Extended abstract in J.C.M. Baeten and J.W. Klop, eds., *Proceedings 1st Conference on Concurrency Theory (CONCUR'90)*, Amsterdam, LNCS 458, Springer, 1990, pp. 278–297.

[20] R.J. van Glabbeek (1993): *Full abstraction in structural operational semantics (extended abstract).* In M. Nivat, C. Rattray, T. Rus, and G. Scollo, eds., *Proceedings 3rd Conference on Algebraic Methodology and Software Technology (AMAST'93)*, Enschede, Workshops in Computing, Springer, pp. 77–84.

[21] R.J. van Glabbeek (1995): *The meaning of negative premises in transition system specifications II.* Report STAN-CS-TN-95-16, Department of Computer Science, Stanford University. Extended abstract in F. Meyer auf der Heide and B. Monien, eds., *Proceedings 23rd Colloquium on Automata, Languages and Programming (ICALP'96)*, Paderborn, LNCS 1099, Springer, 1996, pp. 502–513.

[22] R.J. van Glabbeek & W.P. Weijland (1989): *Refinement in branching time semantics.* In *Proceedings 1st Conference on Algebraic Methodology and Software Technology (AMAST'89)*, Iowa City, pp. 197-201.

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

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

[25] M. Hennessy & R. Milner (1985): *Algebraic laws for nondeterminism and concurrency.* *Journal of the ACM* 32(1), pp. 137–161.

[26] S.C. Kleene (1956): *Representation of events in nerve nets and finite automata.* In C. Shannon and J. McCarthy, eds., *Automata Studies*, Princeton University Press, pp. 3–41.

[27] D. Kozen (1982): *Results on the propositional μ-calculus.* In M. Nielsen and E. Meineche Schmidt, eds., *Proceedings 9th Colloquium on Automata, Languages and Programming (ICALP'82)*, Aarhus, LNCS 140, Springer, pp. 348–359.

[28] K.G. Larsen (1986): *Context-Dependent Bisimulation between Processes.* PhD Thesis, University of Edinburgh.

[29] K.G. Larsen & A. Skou (1991): *Bisimulation through probabilistic testing.* *Information and Computation* 94(1), pp. 1–28.

[30] K.G. Larsen & L. Xinxin (1991): *Compositionality through an operational semantics of contexts.* *Journal of Logic and Computation*, 1(6), pp. 761–795.

[31] R. Milner (1989): *Communication and Concurrency.* Prentice Hall, Englewood Cliffs.

[32] E.-R. Olderog & C.A.R. Hoare (1986): *Specification-oriented semantics for communicating processes.* *Acta Informatica* 23(1), pp. 9–66.

[33] D.M.R. Park (1981): *Concurrency and automata on infinite sequences.* In P. Deussen, ed., *5th GI Conference*, Karlsruhe, LNCS 104, Springer, pp. 167–183.

[34] I.C.C. Phillips (1987): *Refusal testing.* *Theoretical Computer Science* 50(3), pp. 241–284.

[35] G.D. Plotkin (1981): *A structural approach to operational semantics.* Report DAIMI FN-19, Computer Science Department, Aarhus University.

[36] A. Pnueli (1985): *Linear and branching structures in the semantics and logics of reactive systems.* In W. Brauer, ed., *Proceedings 12th Colloquium on Automata, Languages and Programming (ICALP'85)*, Nafplion, LNCS 194, Springer, pp. 15–32.

[37] W.C. Rounds & S.D. Brookes (1981): *Possible futures, acceptances, refusals, and communicating processes.* In *Proceedings 22nd IEEE Symposium on Foundations of Computer Science (FOCS'81)*, Nashville, IEEE Computer Society Press, pp. 140–149.

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

[39] F.W. VAANDRAGER (1991): *On the relationship between process algebra and input/output automata (extended abstract).* In *Proceedings 6th Symposium on Logic in Computer Science, (LICS'91)*, Amsterdam, IEEE Computer Society Press, pp. 387–398.

[40] J.L.M. VRANCKEN (1997): *The algebra of communicating processes with empty process. Theoretical Computer Science* 177(2), pp. 287–328.