Divide and congruence III

Fokkink, Wan; van Glabbeek, Rob; Luttik, Bas

published in
Information and Computation
2019

DOI (link to publisher)
10.1016/j.ic.2019.104435

document version
Publisher's PDF, also known as Version of record

Information and Computation 268 (2019) 104435

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Divide and congruence III: From decomposition of modal formulas to preservation of stability and divergence

Wan Fokkink [a,*], Rob van Glabbeek [b,c,*], Bas Luttik [d,*]

[a] Department of Computer Science, Vrije Universiteit Amsterdam, the Netherlands
[b] Data61, CSIRO, Sydney, Australia
[c] School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
[d] Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, the Netherlands

ARTICLE INFO

Article history:
Received 24 October 2017
Received in revised form 2 April 2019
Accepted 16 July 2019
Available online 24 July 2019

Keywords:
Structural operational semantics
Weak semantics
Modal logic

ABSTRACT

In two earlier papers we derived congruence formats with regard to transition system specifications for weak semantics on the basis of a decomposition method for modal formulas. The idea is that a congruence format for a semantics must ensure that the formulas in the modal characterisation of this semantics are always decomposed into formulas that are again in this modal characterisation. The stability and divergence requirements that are imposed on many of the known weak semantics have so far been outside the realm of this method. Stability refers to the absence of a τ-transition. We show, using the decomposition method, how congruence formats can be relaxed for weak semantics that are stability-respecting. This relaxation for instance brings the priority operator within the range of the stability-respecting branching bisimulation format. Divergence, which refers to the presence of an infinite sequence of τ-transitions, escapes the inductive decomposition method. We circumvent this problem by proving that a congruence format for a stability-respecting weak semantics is also a congruence format for its divergence-preserving counterpart.

1. Introduction

Structural operational semantics [28] provides specification languages with an interpretation in terms of a mathematical notion of behaviour. It generates a labelled transition system, in which states are the closed terms over a (single-sorted, first-order) signature, and transitions between states carry labels. The transitions between states are obtained from a transition system specification (TSS), which consists of a set of proof rules called transition rules. States in labelled transition systems can be identified by a wide range of behavioural equivalences, based on e.g. branching structure or decorated versions of execution sequences. Van Glabbeek [16] classified so-called weak semantics, which take into account the internal action τ. A significant number of the weak semantics based on a bisimulation relation carry a stability or divergence requirement. Stability refers to the absence of a τ-transition and divergence to the presence of an infinite sequence of τ-transitions.

In general a behavioural equivalence induced by a TSS is not guaranteed to be a congruence, i.e. the equivalence class of a term $f(p_1, \ldots, p_n)$ need not be determined by f and the equivalence classes of its arguments $p_1, \ldots, p_n$. Being a congruence is an important property, for instance in order to fit the equivalence into an axiomatic framework. Respecting

* Corresponding authors.
E-mail address: w.j.fokkink@vu.nl (W. Fokkink).

https://doi.org/10.1016/j.ic.2019.104435
0890-5401/

stability or preserving divergence sometimes needs to be imposed in order to obtain a congruence relation, for example in case of the priority operator [1].

Behavioural equivalences can be characterised in terms of the observations that an experimenter could make during a session with a process. Modal logic captures such observations. A modal characterisation of an equivalence on processes consists of a class C of modal formulas such that two processes are equivalent if and only if they satisfy the same formulas in C. For instance, Hennessy-Milner logic [23] constitutes a modal characterisation of (strong) bisimilarity. A cornerstone for the current paper is the work in [6] to decompose formulas from Hennessy-Milner logic with respect to a structural operational semantics in the ntyft format [21] without lookahead. Here the *decomposition* of a modal formula φ w.r.t. a term $t = f(p_1, \ldots, p_n)$ is a selection of n-tuples of modal formulas, one of which needs to be satisfied by the processes p_i in order for t to satisfy φ. Based on this method, congruence formats can be generated for process semantics from their modal characterisation, to ensure that the equivalence is a congruence. Such formats help to avoid repetitive congruence proofs and obtain insight into the congruence property. Key idea is that congruence is ensured if formulas from the modal characterisation of a semantics under consideration are always decomposed into formulas that are again in this modal characterisation. This approach was extended to weak semantics in [15,12]. It crosses the borders between process algebra, structural operational semantics, process semantics, and modal logic.

Here we expand the latter work to weak semantics that respect stability or preserve divergence. We focus on *branching bisimilarity* and *rooted branching bisimilarity* [20] and consider for each a stability-respecting and two divergence-preserving variants. Divergence-preserving branching bisimilarity [20] is the coarsest congruence relation for the parallel composition operator that only equates processes satisfying the same formulas from the well-known temporal logic CTL* minus the next-time operator X [19]. With regard to stability the expansion is relatively straightforward: we extend the modal characterisation of the semantics with one clause to capture that a semantics is stability-respecting, and study the decomposition of this additional clause. Next we show how the congruence formats for branching bisimilarity and rooted branching bisimilarity from [15] can be relaxed, owing to the extended modal characterisation for stability-respecting branching bisimilarity. Notably, the transition rules for the priority operator are within the more relaxed formats.

The divergence preservation property escapes the inductive decomposition method, as it concerns an infinite sequence of τ-transitions. We overcome this problem by presenting a general framework for lifting congruence formats from some weak semantics $\approx$ to a finer weak semantics $\sim$, with the aim to turn congruence formats for stability-respecting weak semantics into congruence formats for their divergence-preserving counterparts. (For the curious reader, it roughly works as follows. Consider a TSS P in a congruence format for $\approx$, and for example a unary function symbol f. A new TSS is created out of P, in which many occurrences of τ in transition rules are suppressed by renaming them into some fresh action name. Furthermore, processes are provided with what we call an oracle transition that reveals some pertinent information on the behaviour of the process, such as whether it can diverge. In this way we ensure that $\approx$ and $\sim$ coincide on the transformed TSS. For each closed term p of P we moreover introduce a constant $\hat{p}$ in the transformed TSS such that $p \sim q$ implies $\hat{p} \sim \hat{q}$, and so $\hat{p} \approx \hat{q}$. We take care that this entire transformation preserves the congruence format for $\approx$. So $\hat{p} \approx \hat{q}$ implies $f(\hat{p}) \approx f(\hat{q})$, and hence, since $\approx$ and $\sim$ coincide on the transformed TSS, $f(\hat{p}) \sim f(\hat{q})$. Finally we cast $f(\hat{p})$ and $f(\hat{q})$ back to processes strongly bisimilar to $f(p)$ and $f(q)$ in the original TSS P, taking care that $\sim$ is preserved. Thus we have gone full circle: $p \sim q$ implies $f(p) \sim f(q)$. This implies that the congruence format for $\approx$ is also a congruence format for $\sim$.) We show four instances where this method can be applied. In two cases $\approx$ is stability-respecting branching bisimilarity and in two cases rooted stability-respecting branching bisimilarity, while the definition of $\sim$ is obtained from $\approx$ by replacing respect for stability by preservation of one of the two aforementioned forms of divergence. Thus it can be concluded that the congruence format for stability-respecting branching bisimilarity is also applicable to divergence-preserving as well as weakly divergence-preserving branching bisimilarity; and likewise for the rooted counterparts of these semantics.

In [2], it was argued that an adaptation of the classical operational semantics of sequential composition in a process theory with the empty process leads to a smoother integration of classical automata theory. A detailed explicit proof is given that rooted divergence-preserving branching bisimilarity is a congruence for the resulting operator, here called sequencing, but also it is observed that this property can be inferred from the congruence format put forward in the current paper.

An extended abstract of the present paper appeared as [13]. This version includes elaborate proofs of the results, provides additional explanations, and discusses sequencing as an extra example application.

2. Preliminaries

This section recalls the basic notions of labelled transition systems and weak semantics and defines stability-respecting and divergence-preserving branching bisimilarity (Sect. 2.1) as well as a modal characterisation of stability-respecting branching bisimilarity (Sect. 2.2). It also presents a brief introduction to structural operational semantics (Sect. 2.3) and recalls some syntactic restrictions on transition rules (Sect. 2.4).

2.1. Stability-respecting and divergence-preserving branching bisimilarity

A *labelled transition system (LTS)* is a triple $(\mathbb{P}, Act, \rightarrow)$, with $\mathbb{P}$ a set of *processes*, Act a set of *actions*, and $\rightarrow \subseteq \mathbb{P} \times Act \times \mathbb{P}$. We normally let $Act = A \cup \{\tau\}$ where τ is an *internal action* and A some set of external or observable actions not containing τ. We write A_τ for $A \cup \{\tau\}$. We use p, q to denote processes, α, β, γ for elements of A_τ, and a, b for elements

of A. We write $p \xrightarrow{\alpha} q$ for $(p,\alpha,q) \in \rightarrow$, $p \xrightarrow{\alpha}$ for $\exists q \in \mathbb{P} : p \xrightarrow{\alpha} q$, and $p \not\xrightarrow{\alpha}$ for $\neg(p \xrightarrow{\alpha})$. Furthermore, $\stackrel{\epsilon}{\Longrightarrow}$ denotes the transitive-reflexive closure of $\xrightarrow{\tau}$. If $p \not\xrightarrow{\tau}$, then we say that p is *stable*.

Definition 1. Let $\mathcal{B} \subseteq \mathbb{P} \times \mathbb{P}$ be a symmetric relation.

- $\mathcal{B}$ is a *branching bisimulation* if $p\,\mathcal{B}\,q$ and $p \xrightarrow{\alpha} p'$ implies that either $\alpha = \tau$ and $p'\,\mathcal{B}\,q$, or $q \stackrel{\epsilon}{\Longrightarrow} q' \xrightarrow{\alpha} q''$ for some q' and q'' with $p\,\mathcal{B}\,q'$ and $p'\,\mathcal{B}\,q''$.
- $\mathcal{B}$ is *stability-respecting* if $p\,\mathcal{B}\,q$ and $p \not\xrightarrow{\tau}$ implies that $q \stackrel{\epsilon}{\Longrightarrow} q' \not\xrightarrow{\tau}$ for some q' with $p\,\mathcal{B}\,q'$.
- $\mathcal{B}$ is *divergence-preserving* if it satisfies the following condition:
  (D) if $p\,\mathcal{B}\,q$ and there is an infinite sequence of processes $(p_k)_{k\in\mathbb{N}}$ such that $p = p_0$, $p_k \xrightarrow{\tau} p_{k+1}$ and $p_k\,\mathcal{B}\,q$ for all $k \in \mathbb{N}$, then there exists an infinite sequence of processes $(q_\ell)_{\ell\in\mathbb{N}}$ such that $q = q_0$, $q_\ell \xrightarrow{\tau} q_{\ell+1}$ for all $\ell \in \mathbb{N}$, and $p_k\,\mathcal{B}\,q_\ell$ for all $k,\ell \in \mathbb{N}$.
  The definition of a *weakly divergence-preserving* relation is obtained by omitting the condition "and $p_k\,\mathcal{B}\,q_\ell$ for all $k,\ell \in \mathbb{N}$". (The condition "and $p_k\,\mathcal{B}\,q$ for all $k \in \mathbb{N}$" is then redundant.)

Processes p,q are *branching bisimilar*, denoted $p \leftrightarroweq_b q$, if there exists a branching bisimulation $\mathcal{B}$ with $p\,\mathcal{B}\,q$. They are *stability-respecting*, *divergence-preserving* or *weakly divergence-preserving* branching bisimilar, denoted by $p \leftrightarroweq_b^s q$, $p \leftrightarroweq_b^\Delta q$ or $p \leftrightarroweq_b^{\Delta^\top} q$, if moreover $\mathcal{B}$ is stability-respecting, divergence-preserving or weakly divergence-preserving, respectively.

We have $\leftrightarroweq_b \supset \leftrightarroweq_b^s \supset \leftrightarroweq_b^{\Delta^\top} \supset \leftrightarroweq_b^\Delta$.

The relations $\leftrightarroweq_b$, $\leftrightarroweq_b^s$, $\leftrightarroweq_b^\Delta$ and $\leftrightarroweq_b^{\Delta^\top}$ are equivalences [3,16,18]. However, they are not *congruences* with respect to most process algebras from the literature, meaning that the equivalence class of a process $f(p_1,\ldots,p_n)$, with f an n-ary function symbol, is not always determined by the equivalence classes of its arguments, i.e. the processes $p_1,\ldots,p_n$. Therefore an additional rootedness condition is imposed.

Definition 2. *Rooted* branching bisimilarity, $\leftrightarroweq_{rb}$, is the largest symmetric relation on $\mathbb{P}$ such that $p \leftrightarroweq_{rb} q$ and $p \xrightarrow{\alpha} p'$ implies that $q \xrightarrow{\alpha} q'$ for some q' with $p' \leftrightarroweq_b q'$. Likewise, *rooted* stability-respecting, divergence-preserving or weakly divergence-preserving branching bisimilarity, denoted by $p \leftrightarroweq_{rb}^s q$, $p \leftrightarroweq_{rb}^\Delta q$ or $p \leftrightarroweq_{rb}^{\Delta^\top} q$, is the largest symmetric relation $\mathcal{R}$ on $\mathbb{P}$ such that $p\,\mathcal{R}\,q$ and $p \xrightarrow{\alpha} p'$ implies that $q \xrightarrow{\alpha} q'$ for some q' with $p' \leftrightarroweq_b^s q'$, $p' \leftrightarroweq_b^\Delta q'$ or $p' \leftrightarroweq_b^{\Delta^\top} q'$, respectively.

Our main aim is to develop congruence formats for stability-respecting and divergence-preserving semantics. These congruence formats will impose syntactic restrictions on the transition rules that are used to generate the underlying LTS. The congruence formats will be determined using the characterising modal logics for the semantics.

We will sometimes refer to strong bisimulation semantics, which ignores the special status of the τ.

Definition 3. A symmetric relation $\mathcal{B} \subseteq \mathbb{P} \times \mathbb{P}$ is a *strong bisimulation* if $p\,\mathcal{B}\,q$ and $p \xrightarrow{a} p'$ implies that $q \xrightarrow{a} q'$ for some q' with $p'\,\mathcal{B}\,q'$. Processes p,q are *strongly bisimilar*, denoted $p \leftrightarroweq q$, if there exists a strong bisimulation $\mathcal{B}$ with $p\,\mathcal{B}\,q$.

The various notions of bisimilarity defined above are examples of so-called behavioural equivalences. For a general formulation of the results in Sect. 5, it is convenient to formally define a notion of behavioural equivalence that includes at least the examples above. Note that a common feature of their definitions is that they associate with every LTS $\mathrm{G} = (\mathbb{P}_\mathrm{G}, Act_\mathrm{G}, \rightarrow_\mathrm{G})$ a binary relation $\sim_\mathrm{G}$. (For instance, in the case of strong bisimilarity, the relation $\sim_\mathrm{G}$ associated with G is defined as the binary relation $\leftrightarroweq \subseteq \mathbb{P}_\mathrm{G} \times \mathbb{P}_G$ such that $p \leftrightarroweq q$ if (and only if) there exists a strong bisimulation $\mathcal{B} \subseteq \mathbb{P}_\mathrm{G} \times \mathbb{P}_G$ such that $p\,\mathcal{B}\,q$.) One may, thus, think of a behavioural equivalence as a family of binary relations indexed by LTSs. It turns out that we need to impose just one extra condition on such families to arrive at a suitable formalisation of the notion of behavioural equivalence. The condition states that the relation associated with the disjoint union of two LTSs restricted to one of the components coincides with the relation associated with that component.

Definition 4. Two LTSs $\mathrm{G} = (\mathbb{P}_\mathrm{G}, Act_\mathrm{G}, \rightarrow_\mathrm{G})$ and $\mathrm{H} = (\mathbb{P}_\mathrm{H}, Act_\mathrm{H}, \rightarrow_\mathrm{H})$ are called *disjoint* if $\mathbb{P}_\mathrm{G} \cap \mathbb{P}_\mathrm{H} = \emptyset$. In that case $\mathrm{G} \uplus \mathrm{H}$ denotes their union $(\mathbb{P}_\mathrm{G} \cup \mathbb{P}_\mathrm{H}, Act_\mathrm{G} \cup Act_\mathrm{H}, \rightarrow_\mathrm{G} \cup \rightarrow_\mathrm{H})$.

A *behavioural equivalence* $\sim$ on LTSs is a family of equivalence relations $\sim_\mathrm{G}$, one for every LTS G, such that for each pair of disjoint LTSs $\mathrm{G} = (\mathbb{P}_\mathrm{G}, Act_\mathrm{G}, \rightarrow_\mathrm{G})$ and H we have $g \sim_\mathrm{G} g' \Leftrightarrow g \sim_{\mathrm{G}\uplus\mathrm{H}} g'$ for any $g, g' \in \mathbb{P}_\mathrm{G}$.

The notions of bisimilarity defined above clearly qualify as behavioural equivalences. Given two behavioural equivalences $\sim$ and $\approx$, we write $\sim \subseteq \approx$ iff $\sim_\mathrm{G} \subseteq \approx_\mathrm{G}$ for each LTS G.

2.2. Modal characterisation

Modal logic formulas express properties on the behaviour of processes in an LTS. Following [16], we extend Hennessy-Milner logic [23] with the modal connectives $\langle\epsilon\rangle\varphi$ and $\langle\hat{\tau}\rangle\varphi$, expressing that a process can perform zero or more, respectively zero or one, τ-transitions to a process where φ holds.

Definition 5. The class $\mathbb{O}$ of *modal formulas* is defined as follows, where I ranges over all index sets and α over A_τ:

$$\mathbb{O} \qquad \varphi ::= \bigwedge_{i\in I}\varphi_i \mid \neg\varphi \mid \langle\alpha\rangle\varphi \mid \langle\epsilon\rangle\varphi \mid \langle\hat{\tau}\rangle\varphi$$

We use abbreviations $\top$ for the empty conjunction, $\varphi_1\wedge\varphi_2$ for $\bigwedge_{i\in\{1,2\}}\varphi_i$, $\varphi\langle\alpha\rangle\varphi'$ for $\varphi\wedge\langle\alpha\rangle\varphi'$, and $\varphi\langle\hat{\tau}\rangle\varphi'$ for $\varphi\wedge\langle\hat{\tau}\rangle\varphi'$.

$p\models\varphi$ denotes that process p satisfies formula φ. The first two operators represent the standard Boolean operators conjunction and negation. We define that (1) $p\models\langle\alpha\rangle\varphi$ if $p\xrightarrow{\alpha}p'$ for some p' with $p'\models\varphi$, (2) $p\models\langle\epsilon\rangle\varphi$ if $p\stackrel{\epsilon}{\Longrightarrow}p'$ for some p' with $p'\models\varphi$, and (3) $p\models\langle\hat{\tau}\rangle\varphi$ if $p\models\varphi$ or $p\models\langle\tau\rangle\varphi$.

For each $L\subseteq\mathbb{O}$, we write $p\sim_L q$ if p and q satisfy the same formulas in L. We say that L is a *modal characterisation* of some behavioural equivalence $\sim$ if $\sim_L$ coincides with $\sim$. We write $\varphi\equiv\varphi'$ if $p\models\varphi\Leftrightarrow p\models\varphi'$ for all processes p. The class $L^{\equiv}$ denotes the closure of $L\subseteq\mathbb{O}$ under $\equiv$. Trivially, $p\sim_L q\Leftrightarrow p\sim_{L^{\equiv}}q$.

Definition 6. [16] The subclasses $\mathbb{O}_b$ and $\mathbb{O}_{rb}$ of $\mathbb{O}$ are defined as follows, where a ranges over A and α over A_τ:

$$\mathbb{O}_b \qquad \varphi ::= \bigwedge_{i\in I}\varphi_i \mid \neg\varphi \mid \langle\epsilon\rangle(\varphi\langle\hat{\tau}\rangle\varphi) \mid \langle\epsilon\rangle(\varphi\langle a\rangle\varphi)$$

$$\mathbb{O}_{rb} \qquad \overline{\varphi} ::= \bigwedge_{i\in I}\overline{\varphi}_i \mid \neg\overline{\varphi} \mid \langle\alpha\rangle\varphi \mid \varphi \qquad (\varphi\in\mathbb{O}_b)$$

$\mathbb{O}_b$ and $\mathbb{O}_{rb}$ are modal characterisations of $\leftrightarrows_b$ and $\leftrightarrows_{rb}$, respectively (see [15]).

The idea behind stability is: (I) if $p\leftrightarrows_b^s q$ and $p\stackrel{\epsilon}{\Longrightarrow}p'\stackrel{\tau}{\not\longrightarrow}$ with $p\leftrightarrows_b^s p'$, then $q\stackrel{\epsilon}{\Longrightarrow}q'\stackrel{\tau}{\not\longrightarrow}$ with $q\leftrightarrows_b^s q'$. In the definition of $\leftrightarrows_b^s$ this was formulated more weakly: (II) if $p\leftrightarrows_b^s q$ and $p\stackrel{\tau}{\not\longrightarrow}$, then $q\stackrel{\epsilon}{\Longrightarrow}q'\stackrel{\tau}{\not\longrightarrow}$ with $q\leftrightarrows_b^s q'$. To argue that formulations (I) and (II) are equivalent, suppose $p\leftrightarrows_b^s q$ and $p\stackrel{\epsilon}{\Longrightarrow}p'\stackrel{\tau}{\not\longrightarrow}$ with $p\leftrightarrows_b^s p'$. Clearly $q\stackrel{\epsilon}{\Longrightarrow}q''$ with $p'\leftrightarrows_b^s q''$. Now the weaker property (II) yields $q''\stackrel{\epsilon}{\Longrightarrow}q'\stackrel{\tau}{\not\longrightarrow}$ with $q''\leftrightarrows_b^s q'$. So $q\stackrel{\epsilon}{\Longrightarrow}q'\stackrel{\tau}{\not\longrightarrow}$ with $q\leftrightarrows_b^s q'$. That formulations (I) and (II) coincide is important to grasp the following modal characterisation of stability-respecting branching bisimilarity, because the additional clause at the end of the definition of $\mathbb{O}_b^s$ is based on formulation (I).

Definition 7. [16] The subclasses $\mathbb{O}_b^s$ and $\mathbb{O}_{rb}^s$ of $\mathbb{O}$ are defined as follows:

$$\mathbb{O}_b^s \quad \varphi ::= \bigwedge_{i\in I}\varphi_i \mid \neg\varphi \mid \langle\epsilon\rangle(\varphi\langle\hat{\tau}\rangle\varphi) \mid \langle\epsilon\rangle(\varphi\langle a\rangle\varphi) \mid \langle\epsilon\rangle(\neg\langle\tau\rangle\top\wedge\overline{\varphi}) \quad (\overline{\varphi}\in\mathbb{O}_{rb}^s)$$

$$\mathbb{O}_{rb}^s \quad \overline{\varphi} ::= \bigwedge_{i\in I}\overline{\varphi}_i \mid \neg\overline{\varphi} \mid \langle\alpha\rangle\varphi \mid \varphi \quad (\varphi\in\mathbb{O}_b^s)$$

The additional clause $\langle\epsilon\rangle(\neg\langle\tau\rangle\top\wedge\overline{\varphi})$ in the definition of $\mathbb{O}_b^s$ expresses stability. The first part $\langle\epsilon\rangle(\neg\langle\tau\rangle\top\ldots)$ captures $p\stackrel{\epsilon}{\Longrightarrow}p'\stackrel{\tau}{\not\longrightarrow}$, while the second part $\ldots\wedge\overline{\varphi}$ captures the stability-respecting branching bisimulation class of p'. Note that since p' is stable and $\leftrightarrows_b^s$ and $\leftrightarrows_{rb}^s$ coincide on stable processes, we can take the second part from $\mathbb{O}_{rb}^s$. The proof of the following theorem is presented in the appendix.

Theorem 8. *$p\leftrightarrows_b^s q\Leftrightarrow p\sim_{\mathbb{O}_b^s}q$ and $p\leftrightarrows_{rb}^s q\Leftrightarrow p\sim_{\mathbb{O}_{rb}^s}q$, for all $p,q\in\mathbb{P}$.*

2.3. Structural operational semantics

A *signature* is a set Σ of function symbols f with arity $ar(f)$. A function symbol of arity 0 is called a *constant*. Let V be an infinite set of variables, with typical elements x,y,z; we assume $|\Sigma|,|A|\leq|V|$. A syntactic object is *closed* if it does not contain any variables. The sets $\mathbb{T}(\Sigma)$ and $\mathrm{T}(\Sigma)$ of terms over Σ and V and closed terms over Σ, respectively, are defined as usual; t,u,v,w denote terms, p,q denote closed terms, and $var(t)$ is the set of variables that occur in term t. A *substitution* σ is a partial function from V to $\mathbb{T}(\Sigma)$. The result $\sigma(t)$ of applying a substitution σ to a term $t\in\mathbb{T}(\Sigma)$ is the term obtained by replacing in t all occurrences of variables x in the domain of σ by $\sigma(x)$. A *closed substitution* is a substitution that is defined on all variables in V and maps every variable to a closed term.

Structural operational semantics [28] generates an LTS in which the processes are the closed terms. The labelled transitions between processes are obtained from a transition system specification, which consists of a set of proof rules called transition rules.

Definition 9. A (*positive* or *negative*) *literal* is an expression $t \xrightarrow{\alpha} u$ or $t \xrightarrow{\alpha} \not\rightarrow$. A *(transition) rule* is of the form $\frac{H}{\lambda}$ with H a set of literals called the *premises*, and λ a literal called the *conclusion*; the terms at the left- and right-hand side of λ are called the *source* and *target* of the rule, respectively. A rule $\frac{\emptyset}{\lambda}$ is also written λ. A rule is *standard* if it has a positive conclusion. A *transition system specification (TSS)*, written (Σ, Act, R), consists of a signature Σ, a set of actions Act, and a collection R of transition rules over Σ. A TSS is *standard* if all its rules are.

A TSS is meant to specify an LTS in which the transitions are the closed positive literals that can be proved using the rules of the TSS. It is straightforward to associate an appropriate notion of provability in the special case of standard TSSs with only positive premises. In the general case, in which the rules of the TSS may have negative premises, consistency is a concern. Literals $t \xrightarrow{\alpha} u$ and $t \xrightarrow{\alpha} \not\rightarrow$ are said to *deny* each other; a notion of provability associated with TSSs is *consistent* if it is not possible to prove two literals that deny each other. To arrive at a consistent notion of provability in the general case, we proceed in two steps: first we define the notion of *irredundant proof*, which on a standard TSS does not allow the derivation of negative literals at all, and then arrive at a notion of *well-supported proof* that allows the derivation of negative literals whose denials are manifestly underivable by irredundant proofs. In [17] it was shown that the notion of well-supported provability is consistent.

Definition 10. [6] Let $P = (\Sigma, Act, R)$ be a TSS. An *irredundant proof* from P of a rule $\frac{H}{\lambda}$ is a well-founded tree with the nodes labelled by literals and some of the leaves marked "hypothesis". The root of this tree has label λ and H is the set of labels of the hypotheses. Moreover, the tree must satisfy the property that if μ is the label of a node that is not a hypothesis and K is the set of labels of the children of this node, then $\frac{K}{\mu}$ is a substitution instance of a rule in R. If there exists such a tree for the rule $\frac{H}{\lambda}$, then we say that it is irredundantly provable from P (notation: $P \vdash_{irr} \frac{H}{\lambda}$).

We note that if a leaf in a proof from P is not marked as hypothesis, then it is a substitution instance of a rule without premises in R.

Definition 11. [17] Let $P = (\Sigma, Act, R)$ be a standard TSS. A *well-supported proof* from P of a closed literal λ is a well-founded tree with the nodes labelled by closed literals, such that the root is labelled by λ, and if μ is the label of a node and K is the set of labels of the children of this node, then:

1. either μ is positive and $\frac{K}{\mu}$ is a closed substitution instance of a rule in R;
2. or μ is negative and for each set N of closed negative literals with $\frac{N}{\nu}$ irredundantly provable from P and ν a closed positive literal denying μ, a literal in K denies one in N.

$P \vdash_{ws} \lambda$ denotes that a well-supported proof from P of λ exists. A standard TSS P is *complete* if for each p and α, either $P \vdash_{ws} p \xrightarrow{\alpha} \not\rightarrow$ or there exists a closed term q such that $P \vdash_{ws} p \xrightarrow{\alpha} q$.

If $P = (\Sigma, Act, R)$ is a complete TSS, then the LTS *associated with* P is the LTS $(\mathrm{T}(\Sigma), Act, \rightarrow)$ with $\rightarrow = \{(p, \alpha, q) \mid P \vdash_{ws} p \xrightarrow{\alpha} q\}$. We do not associate an LTS with an incomplete TSS.

2.4. Congruence formats

Let $P = (\Sigma, Act, R)$ be a transition system specification, and let $\sim_P$ be an equivalence relation defined on the set of closed terms $\mathrm{T}(\Sigma)$. Then $\sim_P$ is a *congruence* for P if, for each $f \in \Sigma$, we have that $p_i \sim_P q_i$ implies $f(p_1, \ldots, p_{ar(f)}) \sim_P f(q_1, \ldots, q_{ar(f)})$. Note that this is the case if for each open term $t \in \mathbb{T}(\Sigma)$ and each pair of closed substitutions $\rho, \rho' : V \rightarrow \mathrm{T}(\Sigma)$ we have $(\forall x \in var(t).\ \rho(x) \sim_P \rho'(x)) \Rightarrow \rho(t) \sim_P \rho'(t)$.

Recall that we have associated with every complete TSS an LTS of which the states are the closed terms of the TSS, and that a behavioural equivalence $\sim$ associates with every such transition system an equivalence on its set of states. Thus, $\sim$ associates with every TSS P an equivalence $\sim_P$ on its set of closed terms. By a *congruence format* for a behavioural equivalence $\sim$ we mean a class of TSSs such that for every TSS P in the class the equivalence $\sim_P$ is a congruence. Usually, a congruence format is defined by means of a list of syntactic restrictions on the rules of TSSs. We proceed to recap some terminology for syntactic restrictions on rules [6,21,22].

Definition 12. An *ntytt rule* is a 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, an *ntyft rule* if its source contains exactly one function symbol and no multiple occurrences of variables, and an *nxytt rule* if the left-hand sides of its premises are variables.

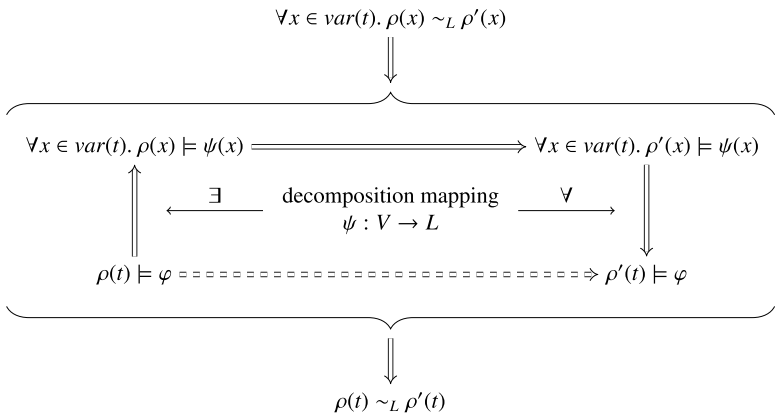

Fig. 1. Modal decomposition.

A well-known congruence format for strong bisimulation semantics is the class of TSSs that consists of ntyft and ntyxt rules only [21,22]. Congruence formats for other semantics are generally obtained by imposing additional restrictions on this ntyft/ntyxt format.

Definition 13. A variable in a rule is *free* if it occurs neither in the source nor in right-hand sides of premises. A rule has *lookahead* if some variable occurs in the right-hand side of a premise and in the left-hand side of a premise. A rule is *decent* if it has no lookahead and does not contain free variables.

Each combination of syntactic restrictions on 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.

Definition 14. A TSS is in *ready simulation format* if it consists of ntyft and ntyxt rules that have no lookahead.

In congruence formats for weak semantics, lookahead of two consecutive actions from A must be forbidden. To see this, consider the extension of CCS [25] with a unary operator f defined by the rule

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

Then $ab\mathbf{0} \leftrightarrows_{rb} a\tau b\mathbf{0}$, whereas $f(ab\mathbf{0}) \not\leftrightarrows_{rb} f(a\tau b\mathbf{0})$. Therefore congruence formats for weak semantics are generally obtained by imposing additional restrictions on the ready simulation format.

3. Modal decomposition

Our goal in this paper is to present and discuss congruence formats for (rooted) stability-respecting and divergence-preserving branching bisimilarity. To derive these congruence formats and prove their correctness, we shall use the modal decomposition approach first proposed in [6], which conveniently employs a modal characterisation of the behavioural equivalence and a modal decomposition result. In this section, we shall explain the main ideas underlying the approach.

Let $\sim$ be the behavioural equivalence under consideration. To prove that some class of TSSs is a congruence format for $\sim$, we should establish that $\sim_P$ is a congruence for every TSS P in the class under consideration. That is, we should establish that for every $t \in \mathbb{T}(\Sigma)$ and for every pair of closed substitutions $\rho, \rho' : V \to \mathrm{T}(\Sigma)$ such that $\rho(x) \sim_P \rho'(x)$ for all $x \in var(t)$ we have that $\rho(t) \sim_P \rho'(t)$. Now, if L is a modal characterisation of $\sim$, then our proof obligation can be reformulated as: for every $t \in \mathbb{T}(\Sigma)$ and for every pair of closed substitutions $\rho, \rho' : V \to \mathrm{T}(\Sigma)$, if, for all $x \in var(t)$, $\rho(x)$ and $\rho'(x)$ satisfy the same formulas in L, then $\rho(t)$ and $\rho'(t)$ satisfy the same formulas in L. Clearly, by symmetry and using standard logical reasoning, it suffices to assume that $\rho(x)$ and $\rho'(x)$ satisfy the same formulas in L for all $x \in var(t)$, consider an arbitrary formula $\varphi \in L$ such that $\rho(t) \models \varphi$, and prove that $\rho'(t) \models \varphi$.

Fig. 1 schematically illustrates how a modal decomposition result can support the correctness argument. To effectively use the assumption that $\rho(x)$ and $\rho'(x)$ satisfy the same formulas in L, it is convenient to express a general correspondence between the satisfaction of formulas in L by $\rho(t)$ and the satisfaction of formulas in L by $\rho(x)$ for all $x \in var(t)$. With every $t \in \mathbb{T}(\Sigma)$ and every formula $\varphi \in L$ we wish to associate an assignment ψ of formulas to the variables occurring in t such that, for all closed substitutions ρ, we have that

$$\rho(t) \models \varphi \ \Leftrightarrow\ \forall x \in var(t).\ \rho(x) \models \psi(x) \ . \tag{1}$$

There does not, in general, exist an assignment ψ that satisfies (1) for all substitutions ρ. We shall see, however, that we can associate with every $t \in \mathbb{T}(\Sigma)$ and $\varphi \in L$ a set of assignments ψ satisfying the implication from right to left for all substitutions ρ; we call such assignments *decomposition mappings* for t and φ. Moreover, we shall see that for every substitution ρ there exists a decomposition mapping that also satisfies the implication from left to right.

The approach described above is applied in [15] to derive congruence formats for (rooted) branching bisimilarity and (rooted) η-bisimilarity, and establish their correctness. Here we do the same for (rooted) *stability-respecting* branching bisimilarity. To this end, we have to show that if the TSS is in the congruence format for (rooted) stability-respecting branching bisimilarity, to be presented in Sect. 4, then the decomposition mappings associated with formulas in the modal characterisation for (rooted) stability-respecting branching bisimilarity yield again formulas in the modal characterisation for (rooted) stability-respecting branching bisimilarity. Since the modal characterisation of (rooted) stability-respecting branching bisimilarity is a small extension of the modal characterisation of (rooted) branching bisimilarity, we can build upon the preservation result for the latter semantics.

In the remainder of this section, we shall further explain and provide some intuitions for the technical ingredients of the modal decomposition approach, referring to the result from [15]. First, we shall introduce in Sect. 3.1 the auxiliary notion of *ruloid* associated with a TSS, which allows us, for every term t and every closed substitution ρ, to characterise the derivability of a transition from $\rho(t)$ in terms of the (non-)derivability of transitions from $\rho(x)$ (with x ranging over $var(t)$). Then we associate, in Sect. 3.2, with every term a set of decomposition mappings. In Sect. 3.3 we present and briefly discuss the congruence format for (rooted) branching bisimilarity from [15].

3.1. Ruloids

Our modal decomposition result should establish a correspondence between the satisfaction of formulas in L by $\rho(t)$ and the satisfaction of formulas in $\mathbb{O}$ by $\rho(x)$ for all $x \in var(t)$. Since the satisfaction of formulas of the shape $\langle\alpha\rangle\varphi$ refers to the transition relation, it is convenient to characterise the provability in a presupposed TSS P of a transition from $\rho(t)$ in terms of the provability or refutability in P of transitions from the $\rho(x)$. The characterisation is provided by the notion of P-ruloid. A P-ruloid is a decent nxytt rule $\frac{H}{\lambda}$ that is irredundantly provable in a TSS that is the result of a transformation of P. In [6], the transformation is presented at length and proved correct. Here we shall explain it superficially, providing just enough detail to be able to convincingly argue later that our congruence formats are preserved under his transformation.

First P is converted to a standard TSS in decent ntyft format. In this conversion from [22], free variables in a rule are replaced by arbitrary closed terms, and if the source is of the form x, then this variable is replaced by a term $f(x_1, \ldots, x_{ar(f)})$ for each n-ary function symbol f in the signature of P, where the variables $x_1, \ldots, x_{ar(f)}$ are fresh. Next, using a construction from [10], left-hand sides of positive premises in rules of P are reduced to variables. In the final transformation step, non-standard rules with a negative conclusion $t \not\xrightarrow{\alpha}$ are introduced. The motivation is that instead of the notion of well-founded provability of Definition 11, we want a more constructive notion like Definition 10, by making it possible that a negative premise is matched with a negative conclusion. A non-standard rule $\frac{H}{f(x_1,\ldots,x_{ar(f)}) \not\xrightarrow{\alpha}}$ is obtained by picking one premise from each standard rule with a conclusion of the form $f(x_1, \ldots, x_{ar(f)}) \xrightarrow{\alpha} t$, and including the denial of each of the selected premises as a premise in H.

The resulting TSS, which is in decent ntyft format, is denoted by P^+. In [6] it was established, for all closed literals μ, that $P \vdash_{ws} \mu$ if and only if μ is irredundantly provable from P^+. The P-ruloids are those decent nxytt rules that are irredundantly provable from P^+. In [6, Lem. 13—aliased 8.2] it was established that $P \vdash_{ws} \rho(t) \xrightarrow{a} q$, with ρ a closed substitution, if and only if there is a ruloid $\frac{H}{t \xrightarrow{a} t'}$ and a closed substitution ρ' that agrees with ρ on $var(t)$ such that $\rho(t') = q$ and $P \vdash_{ws} \rho'(H)$.

3.2. Decomposition of modal formulas

The decomposition method proposed in [15] gives a special treatment to arguments of function symbols that are deemed *patient*; a predicate marks the arguments that get this special treatment.

Definition 15. [5,9] Let Γ be a predicate on arguments of function symbols. A standard ntyft rule is a Γ-*patience rule* if it is of the form

$$\frac{x_i \xrightarrow{\tau} y}{f(x_1, \ldots, x_{ar(f)}) \xrightarrow{\tau} f(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_{ar(f)})}$$

with $\Gamma(f, i)$. A TSS is Γ-*patient* if it contains all Γ-patience rules. A standard ntytt rule is Γ-*patient* if it is irredundantly provable from the Γ-patience rules; else it is called Γ-*impatient*.

A patience rule for an argument i of a function symbol f expresses that terms of the form $f(p_1, \ldots, p_{ar(f)})$ can mimic the τ-transitions of argument p_i (cf. [5,9]). Typically, in process algebra, there are patience rules for the arguments of

parallel composition and for the first argument of sequential composition, but not for the arguments of the alternative composition $+$ or for the second argument of sequential composition.

Definition 16. [6] Let Γ be a predicate on $\{(f,i) \mid 1 \leq i \leq ar(f),\ \ f \in \Sigma\}$. If $\Gamma(f,i)$, then argument i of f is Γ-*liquid*; otherwise it is Γ-*frozen*. An occurrence of x in t is Γ-*liquid* if either $t = x$, or $t = f(t_1, \ldots, t_{ar(f)})$ and the occurrence is Γ-liquid in t_i for a liquid argument i of f; otherwise the occurrence is Γ-*frozen*.

We now show how to decompose formulas from $\mathbb{O}$. To each term t and formula φ we assign a set $t^{-1}(\varphi)$ of decomposition mappings $\psi : V \to \mathbb{O}$. Each of these mappings $\psi \in t^{-1}(\varphi)$ has the property that, for all closed substitutions ρ, $\rho(t) \models \varphi$ if $\rho(x) \models \psi(x)$ for all $x \in var(t)$. Vice versa, whenever $\rho(t) \models \varphi$, there is a decomposition mapping $\psi \in t^{-1}(\varphi)$ with $\rho(x) \models \psi(x)$ for all $x \in var(t)$.

Definition 17. [15] Let $P = (\Sigma, A_\tau, R)$ be a Γ-patient standard TSS in ready simulation format. We define $\cdot^{-1} : \mathbb{T}(\Sigma) \times \mathbb{O} \to \mathcal{P}(V \to \mathbb{O})$ as the function that for each $t \in \mathbb{T}(\Sigma)$ and $\varphi \in \mathbb{O}$ returns the smallest set $t^{-1}(\varphi) \in \mathcal{P}(V \to \mathbb{O})$ of decomposition mappings $\psi : V \to \mathbb{O}$ satisfying the following six conditions. Let t denote a univariate term, i.e. without multiple occurrences of the same variable. (Cases 1–5 associate with every univariate term t a set $t^{-1}(\varphi)$. Then, in Case 6, the definition is generalised to terms that are not univariate, using that every term can be obtained by applying a non-injective substitution to univariate term.)

1. $\psi \in t^{-1}(\bigwedge_{i\in I} \varphi_i)$ iff there are $\psi_i \in t^{-1}(\varphi_i)$ for each $i \in I$ such that
$$\psi(x) = \bigwedge_{i\in I} \psi_i(x) \qquad \text{for all } x \in V$$
2. $\psi \in t^{-1}(\neg\varphi)$ iff there is a function $h : t^{-1}(\varphi) \to var(t)$ such that
$$\psi(x) = \begin{cases} \bigwedge\limits_{\chi \in h^{-1}(x)} \neg\chi(x) & \text{if } x \in var(t) \\ \top & \text{if } x \notin var(t) \end{cases}$$
3. $\psi \in t^{-1}(\langle\alpha\rangle\varphi)$ iff there is a P-ruloid $\frac{H}{t \xrightarrow{\alpha} u}$ and a $\chi \in u^{-1}(\varphi)$ such that
$$\psi(x) = \begin{cases} \chi(x) \ \wedge \bigwedge\limits_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\chi(y) \ \wedge \bigwedge\limits_{x \xrightarrow{\gamma} \!\!\!\!\!/ \ \in H} \neg\langle\gamma\rangle\top & \text{if } x \in var(t) \\ \top & \text{if } x \notin var(t) \end{cases}$$
4. $\psi \in t^{-1}(\langle\epsilon\rangle\varphi)$ iff one of the following holds:
   (a) either there is a $\chi \in t^{-1}(\varphi)$ such that
   $$\psi(x) = \begin{cases} \langle\epsilon\rangle\chi(x) & \text{if } x \text{ occurs } \Gamma\text{-liquid in } t \\ \chi(x) & \text{otherwise} \end{cases}$$
   (b) or there is a Γ-impatient P-ruloid $\frac{H}{t \xrightarrow{\tau} u}$ and a $\chi \in u^{-1}(\langle\epsilon\rangle\varphi)$ such that
   $$\psi(x) = \begin{cases} \top & \text{if } x \notin var(t) \\ \langle\epsilon\rangle\Big(\chi(x) \ \wedge \bigwedge\limits_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\chi(y) \ \wedge \bigwedge\limits_{x \xrightarrow{\gamma} \!\!\!\!\!/ \ \in H} \neg\langle\gamma\rangle\top\Big) & \text{if } x \text{ occurs } \Gamma\text{-liquid in } t \\ \chi(x) \wedge \bigwedge\limits_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\chi(y) \ \wedge \bigwedge\limits_{x \xrightarrow{\gamma} \!\!\!\!\!/ \ \in H} \neg\langle\gamma\rangle\top & \text{otherwise} \end{cases}$$
5. $\psi \in t^{-1}(\langle\hat{\tau}\rangle\varphi)$ iff one of the following holds:
   (a) either $\psi \in t^{-1}(\varphi)$;
   (b) or there is an x_0 that occurs Γ-liquid in t, and a $\chi \in t^{-1}(\varphi)$ such that
   $$\psi(x) = \begin{cases} \langle\hat{\tau}\rangle\chi(x) & \text{if } x = x_0 \\ \chi(x) & \text{otherwise} \end{cases}$$

(c) or there is a Γ-impatient P-ruloid $\frac{H}{t \xrightarrow{\tau} u}$ and a $\chi \in u^{-1}(\varphi)$ such that

$$\psi(x) = \begin{cases} \chi(x) \wedge \bigwedge_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle \chi(y) \ \wedge \bigwedge_{x \xrightarrow{\gamma} \not\in H} \neg\langle\gamma\rangle\top & \text{if } x \in var(t) \\ \top & \text{otherwise} \end{cases}$$

6. $\psi \in \sigma(t)^{-1}(\varphi)$ for a non-injective substitution $\sigma : var(t) \to V$ iff there is a $\chi \in t^{-1}(\varphi)$ such that

$$\psi(x) = \bigwedge_{z \in \sigma^{-1}(x)} \chi(z) \qquad \text{for all } x \in V$$

The following theorem will be the key to the forthcoming congruence results.

Theorem 18. *[15] Let $P = (\Sigma, A_\tau, R)$ be a Γ-patient complete standard TSS in ready simulation format. For each term $t \in \mathbb{T}(\Sigma)$, closed substitution ρ, and $\varphi \in \mathbb{O}$:*

$$\rho(t) \models \varphi \ \Leftrightarrow \ \exists \psi \in t^{-1}(\varphi)\ \forall x \in var(t) :\ \rho(x) \models \psi(x)$$

3.3. Deriving the (rooted) branching bisimulation format

Definition 17 yields for every term t and every formula in the modal language $\mathbb{O}$ a set of decomposition mappings. Note that the modal language $\mathbb{O}_b$ ($\mathbb{O}_{rb}$), which characterises (rooted) branching bisimilarity, is a sublanguage of $\mathbb{O}$. In order to prove that (rooted) branching bisimilarity is a congruence for a TSS P, in view of Theorem 18 it therefore suffices to provide an argument that the decomposition mappings associated with formulas in $\mathbb{O}_b$ ($\mathbb{O}_{rb}$) assign formulas in $\mathbb{O}_b$ ($\mathbb{O}_{rb}$) to all variables. The congruence format should facilitate this argument. By carefully studying the syntactic shapes of the formulas assigned to variables by the decomposition mappings associated with a term t and a formula $\varphi \in \mathbb{O}_b$ ($\varphi \in \mathbb{O}_{rb}$), we can derive sufficient syntactic conditions on ruloids associated with P, which play a role in the definition of the decomposition mappings. These syntactic conditions on the ruloids, in turn, give rise to sufficient conditions on the rules of P.

Let us illustrate this approach by considering a Γ-patient standard TSS P in ready simulation format, a formula $\varphi \in \mathbb{O}_b$ of the shape $\langle\epsilon\rangle(\varphi_1\langle a\rangle\varphi_2)$, with $\varphi_1, \varphi_2 \in \mathbb{O}_b$, a term t, and a variable $x \in var(t)$. We first determine the general shape of the formula $\psi(x)$ assigned to x by a decomposition mapping $\psi \in t^{-1}(\varphi)$, and then formulate sufficient conditions on the rules of P that restrict the general shape in such a way that we can be certain that $\psi(x) \in \mathbb{O}_b$.

Since $\varphi = \langle\epsilon\rangle(\varphi_1\langle a\rangle\varphi_2)$, the decomposition mapping ψ satisfies clause 4 of Definition 17 and hence satisfies one of its two subclauses 4a or 4b. For the purpose of the explanation of the branching bisimulation format, it is enough to consider the case that φ satisfies subclause 4a.

If, on the one hand, x occurs Γ-liquid in t, then from clauses 1 and 3 it follows that there exist a P-ruloid $\frac{H}{t \xrightarrow{\alpha} u}$ and decomposition mappings $\psi_1 \in t^{-1}(\varphi_1)$ and $\psi_2 \in u^{-1}(\varphi_2)$ such that

$$\psi(x) = \langle\epsilon\rangle \left(\psi_1(x) \ \wedge \ \psi_2(x) \ \wedge \bigwedge_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\psi_2(y) \ \wedge \bigwedge_{x \xrightarrow{\gamma} \not\in H} \neg\langle\gamma\rangle\top \right) .$$

If we assume, inductively, that $\psi_1(x)$ and $\psi_2(y)$ are formulas in $\mathbb{O}_b$, then $\psi(x) \in \mathbb{O}_b$ if we can write the argument of $\langle\epsilon\rangle$ in $\psi(x)$ as $\varphi'\langle a\rangle\varphi''$ or as $\varphi'\langle\hat{\tau}\rangle\varphi''$. Clearly, this means that H can contain at most one positive premise $x \xrightarrow{\beta} y$, with $\beta \neq \tau$, and cannot contain a negative premise $x \xrightarrow{\gamma} \not\rightarrow$.

If, on the other hand, x does not occur Γ-liquid in t, then from clauses 1 and 3 it follows that there exist a P-ruloid $\frac{H}{t \xrightarrow{\alpha} u}$ and decomposition mappings $\psi_1 \in t^{-1}(\varphi_1)$ and $\psi_2 \in u^{-1}(\varphi_2)$ such that

$$\psi(x) = \psi_1(x) \ \wedge \psi_2(x) \ \wedge \bigwedge_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\psi_2(y) \ \wedge \bigwedge_{x \xrightarrow{\gamma} \not\in H} \neg\langle\gamma\rangle\top \ ,$$

and it is clear that x cannot occur at all in H.

The analysis above yields a rudimentary syntactic requirement for the P-ruloids $\frac{H}{t \xrightarrow{\alpha} u}$: *$H$ may not contain negative premises and, for all variables $x \in var(t)$, if x has no Γ-liquid occurrence in t, then it does not have a Γ-liquid occurrence in H either, and if x does have a Γ-liquid occurrence in t, then it can have at most one Γ-liquid occurrence of x in H, which must be in a positive premise $x \xrightarrow{\beta} y$ with $\beta \neq \tau$.*

As is shown in [15], the requirement derived above can be relaxed if Γ is defined as $\Lambda \cap \aleph$, where Λ and $\aleph$ are two auxiliary predicates. Intuitively, the predicate Λ marks arguments that contain processes that have started executing (but

may currently be unable to execute), while $\aleph$ marks arguments that contain processes that can execute immediately. For example, in process algebra, Λ and $\aleph$ hold for the arguments of the parallel composition $t_1 \| t_2$ and for the first argument of sequential composition $t_1 \cdot t_2$; they can contain processes that started to execute in the past, and these processes can continue their execution immediately. In absence of the empty process, Λ and $\aleph$ do not hold for the second argument of sequential composition; it contains a process that did not yet start to execute, and cannot execute immediately. If immediate termination is possible, i.e., in the presence of the empty process, then this argument becomes $\aleph$-liquid; cf. the sequencing operator example in Sect. 4.1. Λ does not hold and $\aleph$ holds for the arguments of alternative composition $t_1 + t_2$; they contain processes that did not yet start to execute, but that can start executing immediately.

Below, we recall the (rooted) branching bisimulation format proposed in [15]. Due to the use of the two predicates Λ and $\aleph$, the syntactic restrictions are technically more complicated than sketched above. We refer the reader to [15] for further explanations and examples.

Definition 19. Let $\aleph$ and Λ be predicates on $\{(f,i) \mid 1 \leq i \leq ar(f),\ \ f \in \Sigma\}$. A standard ntytt rule $r = \frac{H}{t \xrightarrow{\alpha} u}$ is *rooted branching bisimulation safe* w.r.t. $\aleph$ and Λ if it satisfies the following conditions.

1. Right-hand sides of positive premises occur only Λ-liquid in u.
2. If $x \in var(t)$ occurs only Λ-liquid in t, then x occurs only Λ-liquid in r.
3. If $x \in var(t)$ occurs only $\aleph$-frozen in t, then x occurs only $\aleph$-frozen in H.
4. If $x \in var(t)$ has exactly one $\aleph$-liquid occurrence in t and this occurrence is also Λ-liquid, then x has at most one $\aleph$-liquid occurrence in H and this occurrence must be in a positive premise. If, moreover, this premise is labelled τ, then r must be $\aleph \cap \Lambda$-patient.

A standard TSS is in *rooted branching bisimulation format* if it is in ready simulation format and, for some $\aleph$ and Λ, it is $\aleph \cap \Lambda$-patient and only contains rules that are rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ. It is in *branching bisimulation format* if moreover Λ is universal, i.e., $\Lambda(f,i)$ for all $f \in \Sigma$ and $i = 1, \ldots, ar(f)$.

4. Stability-respecting branching bisimilarity as a congruence

We present, in Sect. 4.1, congruence formats for stability-respecting branching bisimilarity and rooted stability-respecting branching bisimilarity, as relaxations of the congruence formats for branching bisimilarity and rooted branching bisimilarity, respectively. We illustrate the usefulness of the formats with applications to the priority operator and an operator for sequencing. In Sect. 4.2 we prove the correctness of the formats.

4.1. The (rooted) stability-respecting branching bisimulation format

We define when a standard ntytt rule is rooted stability-respecting branching bisimulation safe, and base the rooted stability-respecting branching bisimulation format on that notion. As with the branching bisimulation format, to define the stability-respecting branching bisimulation format we add one additional restriction to its rooted counterpart: Λ is universal. Our aim for the rest of this section will be to prove that the (rooted) stability-respecting branching bisimulation format guarantees that (rooted) stability-respecting branching bisimilarity is a congruence.

Definition 20. Let $\aleph$ and Λ be predicates on $\{(f,i) \mid 1 \leq i \leq ar(f),\ \ f \in \Sigma\}$. A standard ntytt rule $r = \frac{H}{t \xrightarrow{\alpha} u}$ is *rooted stability-respecting branching bisimulation safe* w.r.t. $\aleph$ and Λ if it satisfies the following conditions.

1. Right-hand sides of positive premises occur only Λ-liquid in u.
2. If $x \in var(t)$ occurs only Λ-liquid in t, then x occurs only Λ-liquid in r.
3. If $x \in var(t)$ occurs only $\aleph$-frozen in t, then x occurs only $\aleph$-frozen in H.
4. Suppose that x has exactly one $\aleph$-liquid occurrence in t, and that this occurrence is also Λ-liquid.
   (a) If x has an $\aleph$-liquid occurrence in a negative premise in H or more than one $\aleph$-liquid occurrence in the positive premises in H, then there is a premise $v \xrightarrow{\tau}\!\!\!\!\!/\ $ in H such that x occurs $\aleph$-liquid in v.
   (b) If there is a premise $w \xrightarrow{\tau} y$ in H and x occurs $\aleph$-liquid in w, then r is $\aleph \cap \Lambda$-patient.

Conditions 1–3 have been copied from Definition 19, and condition 4b is part of condition 4 in that definition. Condition 4a, however, establishes a relaxation of condition 4 in the definition of rooted branching bisimulation safeness, where it was required that x has at most one $\aleph$-liquid occurrence in H, which must be in a positive premise. Here, owing to stability, we can be more tolerant, as long as $x \xrightarrow{\tau}\!\!\!\!\!/\ $ can be derived. As a consequence of this relaxation we will see that the rule for the priority operator is rooted stability-respecting branching bisimulation safe, while it is not rooted branching bisimulation safe.

Definition 21. A standard TSS is in *rooted stability-respecting branching bisimulation format* if it is in ready simulation format and, for some $\aleph$ and Λ, it is $\aleph\cap\Lambda$-patient and only contains rules that are rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ.

This TSS is in *stability-respecting branching bisimulation format* if moreover Λ is universal.

Application to the priority operator. The *priority* operator [1] is a unary function the definition of which is based on an ordering $<$ on atomic actions. The term $\Theta(p)$ executes the transitions of the term p, with the restriction that a transition $p \xrightarrow{\alpha} q$ only gives rise to a transition $\Theta(p) \xrightarrow{\alpha} \Theta(q)$ if there does not exist a transition $p \xrightarrow{\beta} q'$ with $\beta > \alpha$. This intuition is captured by the rule for the priority operator below.

$$\frac{x \xrightarrow{\alpha} y \qquad x \xrightarrow{\beta}\!\!\!\!\!/\ \text{for all } \beta > \alpha}{\Theta(x) \xrightarrow{\alpha} \Theta(y)}$$

The priority operator does not preserve [rooted] branching bisimilarity (cf. [33, pp. 130–132]), as shown by the following example.

Example 22. Consider the following two LTSs:

Clearly $p \leftrightarrows_b q$. Note that on the other hand $p \not\leftrightarrows^s_b q$, because q is stable while p cannot perform a sequence of τ-transitions to a stable state.

Suppose that $a < b$. Let us try to extend the ordering $<$ such that $\Theta(p) \leftrightarrows_b \Theta(q)$. Since $\Theta(p)$ cannot execute the trace aa, we must declare $a < \tau$, to block this trace in $\Theta(q)$. But then $\Theta(p)$ can only execute an infinite τ-sequence while $\Theta(q)$ can execute a. Hence $\Theta(p) \not\leftrightarrows_b \Theta(q)$ for every ordering $<$ (with $a < b$).

Moreover, if p_0 and q_0 are processes with as only transitions $p_0 \xrightarrow{\tau} p$ and $q_0 \xrightarrow{\tau} q$, then $p_0 \leftrightarrows_{rb} q_0$, but $\Theta(p_0) \not\leftrightarrows_{rb} \Theta(q_0)$ for every ordering $<$ (with $a < b$).

So inevitably, as observed in [15], the rule for the priority operator is not in the rooted branching bisimulation format. Namely, the $\aleph\cap\Lambda$-liquid argument x in the source occurs $\aleph$-liquid in the negative premises, which violates the more restrictive condition 4 of the rooted branching bisimulation format.

We proceed to show that the rule for the priority operator does satisfy the relaxed condition 4 of Definition 20. To this end, first note that in view of the target $\Theta(y)$, by condition 1 of Definition 20, the argument of Θ must be chosen Λ-liquid. And in view of condition 3 of Definition 20, the argument of Θ must be chosen $\aleph$-liquid. Then the rule above is rooted stability-respecting branching bisimulation safe, if the following condition on the ordering on atomic actions is satisfied: if there is a β such that $\beta > \alpha$, then $\tau > \alpha$. Namely, this guarantees that condition 4a of Definition 20 is satisfied: if there is a negative premise $x \xrightarrow{\beta}\!\!\!\!\!/$, then there is also a negative premise $x \xrightarrow{\tau}\!\!\!\!\!/$. Moreover, note that then the rule for the priority operator with $\alpha = \tau$ constitutes a patience rule because there can be no β with $\beta > \tau$. Furthermore, since the argument of Θ is Λ-liquid, this operator is within the stability-respecting branching bisimulation format. Theorem 32 and Theorem 33, which are presented at the end of Sect. 4, will therefore imply the following congruence results.

Corollary 23. *$\leftrightarrows^s_b$ and $\leftrightarrows^s_{rb}$ are congruences for the priority operator.*

Application to sequencing. The binary *sequencing operator* ; is a variant of sequential composition that does not rely on a notion of successful termination. Intuitively, the process $p\,;q$ behaves as its left-hand side argument until that can no longer do any transitions; then it proceeds with its right-hand side argument. The rules below, which appeared e.g. in [4], formalise this behaviour:

$$\frac{x \xrightarrow{\alpha} x'}{x\,;y \xrightarrow{\alpha} x'\,;y} \qquad \frac{x \xrightarrow{\alpha}\!\!\!\!\!/\ \text{for all } \alpha \in A_\tau \qquad y \xrightarrow{\beta} y'}{x\,;y \xrightarrow{\beta} y'}$$

Sequencing does not preserve [rooted] branching bisimilarity, as shown by the following example.

Example 24. Consider three processes p, q, and r with $p \xrightarrow{\tau} p$ and $r \xrightarrow{a} \_$ while q cannot perform any transitions. Then $p \leftrightarrows_b q$, but $p\,;r \leftrightarrows p \not\leftrightarrows_b r \leftrightarrows q\,;r$. Note that $p \not\leftrightarrows^s_b q$ since q is stable while p cannot perform a sequence of τ-transitions to a stable state.

Using processes p_0 and q_0 as in Ex. 22 shows that also $\leftrightarrows_{rb}$ fails to be a congruence for sequencing.

We proceed to show that if both arguments of ; are chosen to be $\aleph$-liquid, and only the first argument of ; is chosen to be Λ-liquid, then both rules are rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ (see Definition 20):

1. The right-hand side x' of the positive premise in the first rule occurs Λ-liquid in $x'\,;y$.
2. In both rules, the variable x has only Λ-liquid occurrences.
3. In both rules, both variables x and y have $\aleph$-liquid occurrences in the source.
4. In both rules, only the variable x has exactly one $\aleph$-liquid occurrence in the source that is also Λ-liquid.
   (a) The variable x does not have more than one $\aleph$-liquid occurrence in the positive premises of the rules. It does have $\aleph$-liquid occurrences in negative premises of the second rule, but, since α ranges over all actions in A_τ, there is also a premise $x \xrightarrow{\tau}\!\!\!\!\!/\;$.
   (b) Clearly the first rule with $\alpha = \tau$, which has a premise $x \xrightarrow{\tau} x'$ with an $\aleph$-liquid occurrence of x, is $\aleph\cap\Lambda$-patient.

Theorem 33, which is presented at the end of Sect. 4, will therefore imply the following congruence result.

Corollary 25. *$\leftrightarrows^s_{rb}$ is a congruence for the sequencing operator.*

Note that we cannot take Λ to be universal, for then by condition 4b we would need the second rule for $\beta = \tau$ to be $\aleph\cap\Lambda$-patient. Yet, as is easy to check, $\leftrightarrows^s_b$ is a congruence for sequencing. This shows that our (unrooted) stability-respecting branching bisimulation format does not cover all relevant operators from the literature.

We argue that it is not straightforward to develop a congruence format for stability-preserving branching bisimilarity that includes the rules for the sequencing operator. The second rule for the sequencing operator is not a patience rule for two reasons: it has negative premises and the target is not of the form $x\,;y'$. Putting $x\,;y'$ as the target would not change the semantics in an essential way, so only the first reason is relevant. The following example shows that a mild generalisation of the notion of patience rule allowing some negative premises would not work.

Example 26. Consider a binary function symbol f defined by the following three rules:

$$\frac{x \xrightarrow{\tau} x' \qquad y \xrightarrow{\tau}\!\!\!\!\!/\;}{f(x,y) \xrightarrow{\tau} f(x',y)} \qquad \frac{x \xrightarrow{\tau}\!\!\!\!\!/\; \qquad y \xrightarrow{\tau} y'}{f(x,y) \xrightarrow{\tau} f(x,y')} \qquad \frac{x \xrightarrow{a} x'}{f(x,y) \xrightarrow{a} f(x',y)}$$

Similar to the second argument of the sequencing operator, the 'patience rules' for the two arguments of f both carry an additional negative premise. Making both arguments Λ- and $\aleph$-liquid, all requirements of the stability-respecting branching bisimulation format are met, including 4b if the first two rules are considered patience rules.

However, $\leftrightarrows^s_b$ is not a congruence for f. Suppose $p \xrightarrow{\tau} p' \xrightarrow{a} p''$ and $q \xrightarrow{\tau} q'$. Then clearly $p \leftrightarrows^s_b p'$, but $f(p,q) \not\leftrightarrows^s_b f(p',q)$ because $f(p,q)$ cannot perform any transition while $f(p',q)$ can perform an a-transition.

4.2. Correctness

To prove that (rooted) stability-respecting branching bisimilarity is indeed a congruence for every complete standard TSS P in the (rooted) stability-respecting branching bisimulation format, we use the modal decomposition method discussed in Sect. 3. It suffices to establish that the transformation to ruloids preserves the syntactic restrictions of the format, and that the format, in turn, ensures that decomposition mappings associated with a formula in $\mathbb{O}^s_b$ ($\mathbb{O}^s_{rb}$) assign to the variables again formulas in $\mathbb{O}^s_b$ ($\mathbb{O}^s_{rb}$).

Preservation of syntactic restrictions. The definition of modal decomposition is based on P-ruloids. Therefore we must verify that if P is in rooted stability-respecting branching bisimulation format, then the P-ruloids are rooted stability-respecting branching bisimulation safe (Proposition 30). The key part of the proof is to show that the syntactic restriction of decent rooted stability-respecting branching bisimulation safety is preserved under irredundant provability (Lemma 29).

In the proof of the preservation lemma below, rules with a negative conclusion will play an important role. Therefore the notion of rooted stability-respecting branching bisimulation safety needs to be extended to non-standard rules. The following definition coincides with the definition of rooted branching bisimulation safe for non-standard rules in [15].

Definition 27. An ntytt rule $r = \frac{H}{t \xrightarrow{\alpha}\!\!\!\!\!/\;}$ is *rooted stability-respecting branching bisimulation safe* w.r.t. $\aleph$ and Λ if it satisfies conditions 2 and 3 of Definition 20.

Lemma 28. *Let Q be an $\aleph\cap\Lambda$-patient TSS in decent ntyft format. If an ntytt rule $\frac{H}{t\xrightarrow{\tau}/}$ is provable from Q^+ and x occurs $\aleph\cap\Lambda$-liquid in t, then H has a premise $v\xrightarrow{\tau}/$ where x occurs $\aleph\cap\Lambda$-liquid in v.*

Proof. Recall from Sect. 3.1 that also the TSS Q^+ is in decent ntyft format. Let $\frac{H}{t\xrightarrow{\tau}/}$ be provable from Q^+, by means of a proof π. We apply structural induction with respect to π.

Induction basis: The case where π has only one node, marked "hypothesis", is trivial, as then H contains $t\xrightarrow{\tau}/$.

Induction step: Let r be the decent ntyft rule and σ the substitution used at the bottom of π. Let $f(x_1,\ldots,x_{ar(f)})\xrightarrow{\tau}/$ be the conclusion of r. Then $\sigma(f(x_1,\ldots,x_{ar(f)}))=t$. Since x occurs $\aleph\cap\Lambda$-liquid in t, it occurs $\aleph\cap\Lambda$-liquid in $\sigma(x_{i_0})$ where i_0 is an $\aleph\cap\Lambda$-liquid argument of f. In view of the patience rule for this argument of f and the construction of rules with a negative premise it follows that r has a premise $x_{i_0}\xrightarrow{\tau}/$. So a rule $\frac{H'}{\sigma(x_{i_0})\xrightarrow{\tau}/}$ is provable from Q^+ by means of a strict subproof of π, where $H'\subseteq H$. By induction H' contains a premise $v\xrightarrow{\tau}/$ where x occurs $\aleph\cap\Lambda$-liquid in v. □

Lemma 29. *Let P be an $\aleph\cap\Lambda$-patient TSS in decent ntyft format, in which each rule is rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ. Moreover, P is either a standard TSS or a TSS of the form Q^+ with Q an $\aleph\cap\Lambda$-patient standard TSS in decent ntyft format. Then each ntytt rule irredundantly provable from P is rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ.*

Proof. For all conditions of Definition 20 and Definition 27 except 4a, the preservation proof coincides with the corresponding proof for rooted branching bisimulation safeness in [15, Lem. 5—aliased 4.5]. Therefore we here focus only on condition 4a. Let an ntytt rule $\frac{H}{t\xrightarrow{\alpha}u}$ be irredundantly provable from P, by means of a proof π. We prove, using structural induction with respect to π, that this rule satisfies condition 4a of Definition 20.

Induction basis: Suppose π has only one node, marked "hypothesis". Then $\frac{H}{t\xrightarrow{\alpha}u}$ equals $\frac{t\xrightarrow{\alpha}u}{t\xrightarrow{\alpha}u}$ (so u is a variable). This rule trivially satisfies condition 4a of Definition 20.

Induction step: Let r be the decent ntyft rule and σ the substitution used at the bottom of π. By assumption, r is decent, ntyft, and rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ. Let r be of the form

$$\frac{\{v_k\xrightarrow{\beta_k}y_k\mid k\in K\}\cup\{w_\ell\xrightarrow{\gamma_\ell}/\mid\ell\in L\}}{f(x_1,\ldots,x_{ar(f)})\xrightarrow{\alpha}v}$$

Then $\sigma(f(x_1,\ldots,x_{ar(f)}))=t$ and $\sigma(v)=u$. Also, decent ntytt rules $r_k=\frac{H_k}{\sigma(v_k)\xrightarrow{\beta_k}\sigma(y_k)}$ for each $k\in K$ and $r_\ell=\frac{H_\ell}{\sigma(w_\ell)\xrightarrow{\gamma_\ell}/}$ for each $\ell\in L$ 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, they are rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ.

Suppose that x has exactly one $\aleph$-liquid occurrence in t, which is also Λ-liquid. Then there is an $i_0\in\{1,\ldots,ar(f)\}$ with $\aleph(f,i_0)$ and $\Lambda(f,i_0)$ such that x has exactly one $\aleph$-liquid occurrence in $\sigma(x_{i_0})$, which is also Λ-liquid. Furthermore, for each $i\in\{1,\ldots,ar(f)\}\backslash\{i_0\}$, $\neg\aleph(f,i)$ or x occurs only $\aleph$-frozen in $\sigma(x_i)$. Since the ntyft rule r is rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ, by condition 3 of Definition 20, if $\neg\aleph(f,i)$, then x_i occurs only $\aleph$-frozen in v_k for all $k\in K$, as well as in w_ℓ for all $\ell\in L$. We distinguish three possible cases, and argue each time that condition 4a of Definition 20 is satisfied.

Case 1: x_{i_0} has no $\aleph$-liquid occurrences in the premises of r. Then x has no $\aleph$-liquid occurrences in $\sigma(v_k)$ for $k\in K$ and $\sigma(w_\ell)$ for $\ell\in L$. So by condition 3 of Definition 20 and decency of the r_k and r_ℓ, x has no $\aleph$-liquid occurrences in H.

Case 2: x_{i_0} has exactly one $\aleph$-liquid occurrence in the premises of r, in v_{k_0} for some $k_0\in K$. By condition 2 of Definition 20 this occurrence is also Λ-liquid. Then x has exactly one $\aleph$-liquid occurrence in $\sigma(v_{k_0})$, and this occurrence is also Λ-liquid. So by condition 4a of Definition 20, if x has an $\aleph$-liquid occurrence in a negative premise in H_{k_0} or more than one in the positive premises in H_{k_0}, then there is a premise $w\xrightarrow{\tau}/$ in $H_{k_0}\subseteq H$ where x occurs $\aleph$-liquid in w. Furthermore, x has no $\aleph$-liquid occurrences in $\sigma(v_k)$ for $k\in K\setminus\{k_0\}$ and $\sigma(w_\ell)$ for $\ell\in L$. So by condition 3 of Definition 20, x has no $\aleph$-liquid occurrences in H_k for $k\in K\setminus\{k_0\}$ and H_ℓ for $\ell\in L$.

Case 3: x_{i_0} occurs $\aleph$-liquid in w_{ℓ_0} for some $\ell_0\in L$ or has more than one $\aleph$-liquid occurrence in the v_k for $k\in K$. Then, by condition 4a of Definition 20, x_{i_0} occurs $\aleph$-liquid in w_{ℓ_1} for some $\ell_1\in L$ with $\gamma_{\ell_1}=\tau$. By condition 2 of Definition 20 this occurrence is also Λ-liquid. It follows that x occurs $\aleph\cap\Lambda$-liquid in $\sigma(w_{\ell_1})$. In case P is a standard TSS the premise $w_\ell\xrightarrow{\gamma_\ell}/$ occurs in H, and otherwise, by Lemma 28, $H_{\ell_1}\subseteq H$ contains a premise $w\xrightarrow{\tau}/$ where x occurs $\aleph\cap\Lambda$-liquid in w. □

The following proposition can now be proved in the same way as the corresponding Prop. 2—aliased 4.6—for rooted branching bisimulation safeness in [15].

Proposition 30. *Let P be an $\aleph\cap\Lambda$-patient TSS in ready simulation format, in which each rule is rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ. Then each P-ruloid is rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ.*

Preservation of modal characterisations. Given a standard TSS in rooted stability-respecting branching bisimulation format, w.r.t. some $\aleph$ and Λ, Definition 17 yields decomposition mappings $\psi \in t^{-1}(\varphi)$, with $\Gamma := \aleph\cap\Lambda$. In this section we prove that if $\varphi \in \mathbb{O}_b^s$, then $\psi(x) \in \mathbb{O}_b^{s\equiv}$ if x occurs only Λ-liquid in t. (That is why in the stability-respecting branching bisimulation format, Λ must be universal.) Furthermore, we prove that if $\varphi \in \mathbb{O}_{rb}^s$, then $\psi(x) \in \mathbb{O}_{rb}^{s\equiv}$ for all variables x. From these preservation results we will deduce the promised congruence results for unrooted and rooted stability-respecting branching bisimilarity, respectively.

In [15] the following proposition was proved in two separate steps: Prop. 3 and Prop. 4—aliased 4.7 and 4.8—in that paper. This was possible since $\mathbb{O}_{rb}$ incorporates $\mathbb{O}_b$ but not vice versa. However, since there is a circular dependency between the definitions of $\mathbb{O}_b^s$ and $\mathbb{O}_{rb}^s$, here we need to prove the corresponding results for these modal characterisations simultaneously. The third part of the proposition below is merely a tool in proving the other parts.

Proposition 31. *Let P be an $\aleph\cap\Lambda$-patient standard TSS in ready simulation format, in which each rule is rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ.*

1. *For each term t and variable x that occurs only Λ-liquid in t:*
$$\varphi \in \mathbb{O}_b^s \;\Rightarrow\; \forall \psi \in t^{-1}(\varphi) : \psi(x) \in \mathbb{O}_b^{s\equiv}$$
2. *For each term t and variable x:*
$$\overline{\varphi} \in \mathbb{O}_{rb}^s \;\Rightarrow\; \forall \psi \in t^{-1}(\overline{\varphi}) : \psi(x) \in \mathbb{O}_{rb}^{s\equiv}$$
3. *For each term t and variable x that occurs only Λ-liquid and $\aleph$-frozen in t:*
$$\overline{\varphi} \in \mathbb{O}_{rb}^s \;\Rightarrow\; \forall \psi \in t^{-1}(\overline{\varphi}) : \psi(x) \in \mathbb{O}_b^{s\equiv}$$

Proof. We apply simultaneous induction on the structure of φ, resp. $\overline{\varphi}$, and the construction of ψ. We only treat the case where t is univariate. The case where t is not univariate can be dealt with in the same way as in the proofs of the corresponding Prop. 3 and Prop. 4—aliased 4.7 and 4.8—in [15]. Let $\psi \in t^{-1}(\varphi)$. If $x \notin var(t)$ then $\psi(x) \equiv \top \in \mathbb{O}_b^{s\equiv}$. So suppose x occurs exactly once in t.

We start with the first claim of the proposition. The cases where φ is of the form $\bigwedge_{i\in I}\varphi_i$ or $\neg\varphi'$ can be dealt with as in the proof of Prop. 3—aliased 4.7—in [15]. We therefore focus on the other cases.

- $\varphi = \langle\varepsilon\rangle(\varphi_1\langle\hat{\tau}\rangle\varphi_2)$ with $\varphi_1, \varphi_2 \in \mathbb{O}_b^s$. Let the occurrence of x in t be Λ-liquid. According to Definition 17.4 we can distinguish two cases.
  - Case 1: $\psi(x)$ is defined based on Definition 17.4a. Then $\psi(x) = \langle\varepsilon\rangle\chi(x)$ if x occurs $\aleph$-liquid in t, or $\psi(x) = \chi(x)$ if x occurs $\aleph$-frozen in t, for some $\chi \in t^{-1}(\varphi_1\langle\hat{\tau}\rangle\varphi_2)$. By Definition 17.1, $\chi(x) = \chi_1(x) \wedge \chi_2(x)$ with $\chi_1 \in t^{-1}(\varphi_1)$ and $\chi_2 \in t^{-1}(\langle\hat{\tau}\rangle\varphi_2)$. By induction on formula size, $\chi_1(x) \in \mathbb{O}_b^{s\equiv}$. For $\chi_2(x)$, according to Definition 17.5, we can distinguish three cases. Cases 1.1 and 1.2 where $\chi_2(x)$ is defined based on Definition 17.5a and Definition 17.5b, respectively, proceed in the same way is in the proof of [15, Prop. 3]. We focus on the third case.
  - Case 1.3: $\chi_2(x)$ is defined based on Definition 17.5c, employing an $\aleph\cap\Lambda$-impatient P-ruloid $\frac{H}{t\xrightarrow{\tau}u}$ and a $\xi \in u^{-1}(\varphi_2)$. So $\chi_2(x) = \xi(x) \wedge \bigwedge_{x\xrightarrow{\beta}y\in H}\langle\beta\rangle\xi(y) \wedge \bigwedge_{x\xrightarrow{\gamma}\not\in H}\neg\langle\gamma\rangle\top$. By Proposition 30, $\frac{H}{t\xrightarrow{\tau}u}$ is rooted stability-respecting branching bisimulation safe. Since the occurrence of x in t is Λ-liquid, by condition 2 of Definition 20, x occurs only Λ-liquid in u. Therefore, by induction on formula size, $\xi(x) \in \mathbb{O}_b^{s\equiv}$. Case 1.3.1 where the occurrence of x in t is $\aleph$-frozen still proceeds in the same way is in the proof of [15, Prop. 3]. We focus on the other case.
  - Case 1.3.2: The occurrence of x in t is $\aleph$-liquid. If H has at most one premise of the form $x\xrightarrow{\beta}y$, for which $\beta \neq \tau$, and none of the form $x\xrightarrow{\gamma}\not$, then still the proof proceeds in the same way is in the proof of [15, Prop. 3]. However, the more relaxed condition 4a of Definition 20 allows H to have more than one premise of the form $x\xrightarrow{\beta}y$ with $\beta \neq \tau$, or premises $x\xrightarrow{\gamma}\not$. Only, then H must also contain the premise $x\xrightarrow{\tau}\not$. Thus $\psi(x) \equiv \langle\varepsilon\rangle\big(\neg\langle\tau\rangle\top \wedge \chi_1(x) \wedge \xi(x) \wedge \bigwedge_{x\xrightarrow{b}y\in H}\langle b\rangle\xi(y) \wedge \bigwedge_{x\xrightarrow{c}\not\in H}\neg\langle c\rangle\top\big)$. By condition 1 of Definition 20 the right-hand sides y of positive premises in H occur only Λ-liquid in u, so by induction $\xi(y) \in \mathbb{O}_b^{s\equiv}$. Hence the conjuncts $\langle b\rangle\xi(y)$ are in $\mathbb{O}_{rb}^{s\equiv}$. It follows that $\psi(x) \in \mathbb{O}_b^{s\equiv}$.
  - Case 2: $\psi(x)$ is defined based on Definition 17.4b, employing an $\aleph\cap\Lambda$-impatient P-ruloid $\frac{H}{t\xrightarrow{\tau}u}$ and a $\chi \in u^{-1}(\langle\varepsilon\rangle(\varphi_1\langle\hat{\tau}\rangle\varphi_2))$. By Proposition 30, $\frac{H}{t\xrightarrow{\tau}u}$ is rooted stability-respecting branching bisimulation safe. Since the occurrence of x in t is Λ-liquid, by condition 2 of Definition 20, x occurs only Λ-liquid in u. Therefore, by

induction on the construction of ψ, $\chi(x) \in \mathbb{O}_b^{\equiv}$. Case 2.1 where the occurrence of x in t is $\aleph$-frozen proceeds in the same way is in the proof of [15, Prop. 3]. We focus on the other case.

Case 2.2: The occurrence of x in t is $\aleph$-liquid. Then we have $\psi(x) = \langle\varepsilon\rangle\big(\chi(x) \wedge \bigwedge_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\chi(y) \wedge \bigwedge_{x \xrightarrow{\gamma} \not\rightarrow \in H} \neg\langle\gamma\rangle\top\big)$.

If H has at most one premise of the form $x \xrightarrow{\beta} y$, for which $\beta \neq \tau$, and none of the form $x \xrightarrow{\gamma} \not\rightarrow$, then still the proof proceeds in the same way is in the proof of [15, Prop. 3]. However, the more relaxed condition 4a of Definition 20 allows H to have more than one premise of the form $x \xrightarrow{\beta} y$ with $\beta \neq \tau$, or premises $x \xrightarrow{\gamma} \not\rightarrow$. Only, then H must also contain the premise $x \xrightarrow{\tau} \not\rightarrow$. Thus $\psi(x) \equiv \langle\varepsilon\rangle\big(\neg\langle\tau\rangle\top \wedge \chi(x) \wedge \bigwedge_{x \xrightarrow{b} y \in H} \langle b\rangle\chi(y) \wedge \bigwedge_{x \xrightarrow{c} \not\rightarrow \in H} \neg\langle c\rangle\top\big)$. By condition 1 of Definition 20 the right-hand sides y of positive premises in H occur only Λ-liquid in u, so by induction $\xi(y) \in \mathbb{O}_b^{s\equiv}$. Hence the conjuncts $\langle b\rangle\chi(y)$ are in $\mathbb{O}_{rb}^{s\equiv}$. It follows that $\psi(x) \in \mathbb{O}_b^{s\equiv}$.

The proof of the case $\varphi = \langle\varepsilon\rangle(\varphi_1\langle a\rangle\varphi_2)$ in $\mathbb{O}_b$ from [15, Prop. 3] needs to be adapted in a similar fashion as the case $\varphi = \langle\varepsilon\rangle(\varphi_1\langle\hat{\tau}\rangle\varphi_2)$. We take the liberty to omit this adaptation here, and continue with the additional clause in the modal characterisation of $\mathbb{O}_b^s$, compared to $\mathbb{O}_b$.

- $\varphi = \langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \overline{\varphi})$ with $\overline{\varphi} \in \mathbb{O}_{rb}^s$. Let the occurrence of x in t be Λ-liquid. According to Definition 17.4 we can distinguish two cases.

  Case 1: $\psi(x)$ is defined based on Definition 17.4a. Then $\psi(x) = \langle\varepsilon\rangle\chi(x)$ if x occurs $\aleph$-liquid in t, or $\psi(x) = \chi(x)$ if x occurs $\aleph$-frozen in t, for some $\chi \in t^{-1}(\neg\langle\tau\rangle\top \wedge \overline{\varphi})$. By Definition 17.1, $\chi(x) = \chi_1(x) \wedge \chi_2(x)$ with $\chi_1 \in t^{-1}(\neg\langle\tau\rangle\top)$ and $\chi_2 \in t^{-1}(\overline{\varphi})$. By Definition 17.2 there is a function $h : t^{-1}(\langle\tau\rangle\top) \to var(t)$ such that $\chi_1(x) = \wedge_{\xi \in h^{-1}(x)} \neg\xi(x)$.

  Case 1.1: x occurs $\aleph$-liquid in t. By Definition 17.3, for each $\xi \in h^{-1}(x)$, $\xi(x)$ is of the form $\bigwedge_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\top \wedge \bigwedge_{x \xrightarrow{\gamma} \not\rightarrow \in H} \neg\langle\gamma\rangle\top$ for some P-ruloid $\frac{H}{t \xrightarrow{\tau} u}$. Note that such formulas are in $\mathbb{O}_{rb}^s$. Moreover, by induction on formula size, $\chi_2(x) \in \mathbb{O}_{rb}^{s\equiv}$. Since the occurrence of x in t is $\aleph\cap\Lambda$-liquid, there is an $\aleph\cap\Lambda$-patient ruloid $\frac{x \xrightarrow{\tau} y}{t \xrightarrow{\tau} t'}$. This gives rise to a $\xi \in h^{-1}(x)$ such that $\xi(x) = \neg\langle\tau\rangle\top$. Concluding, $\psi(x) = \langle\varepsilon\rangle(\chi_1(x) \wedge \chi_2(x))$ is of the form $\langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \overline{\varphi}')$ for some $\overline{\varphi}' \in \mathbb{O}_{rb}^s$. So $\psi(x) \in \mathbb{O}_b^s$.

  Case 1.2: x occurs $\aleph$-frozen in t. Then by condition 3 of Definition 20, x does not occur in H. Since moreover $\omega(x) \equiv \top$ for each $\omega \in t^{-1}(\top)$, by Definition 17.3, $\xi(x) \equiv \top$ for each $\xi \in h^{-1}(x)$. So either $\chi_1(x) \equiv \neg\top$ if $h^{-1}(x)$ is non-empty or $\chi_1(x) \equiv \top$ if $h^{-1}(x)$ is empty. In the first case $\psi(x) \equiv \neg\top$, so then we are done. In the second case, by induction on formula size, using the third claim of this proposition, $\psi(x) \equiv \chi_2(x) \in \mathbb{O}_b^{s\equiv}$.

  Case 2: $\psi(x)$ is defined based on Definition 17.4b. This case proceeds in the same way as case 2 of $\varphi = \langle\varepsilon\rangle(\varphi_1\langle\hat{\tau}\rangle\varphi_2)$.

This completes the proof of the first claim of the proposition. We continue with the second claim of the proposition. The cases where $\overline{\varphi}$ is of the form $\bigwedge_{i\in I}\overline{\varphi}_i$ or $\neg\overline{\varphi}'$ or $\langle\alpha\rangle\varphi$ can be dealt with as in the proof of Prop. 4 in [15]. We focus on the only other case.

- $\overline{\varphi} \in \mathbb{O}_b^s$. The cases where $\overline{\varphi}$ is of the form $\bigwedge_{i\in I}\varphi_i$ or $\neg\varphi'$ or $\langle\varepsilon\rangle(\varphi_1\langle\hat{\tau}\rangle\varphi_2)$ or $\langle\varepsilon\rangle(\varphi_1\langle a\rangle\varphi_2)$ can be dealt with as in the proof of Prop. 4 in [15]. We focus on the only new case here.

∗ $\overline{\varphi} = \langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \overline{\varphi}')$ with $\overline{\varphi}' \in \mathbb{O}_{rb}^s$. If the occurrence of x in t is Λ-liquid, then we already proved in the corresponding case for the first claim of the proposition that $\psi(x) \in \mathbb{O}_b^{s\equiv} \subset \mathbb{O}_{rb}^{s\equiv}$. So we can assume that this occurrence is Λ-frozen. According to Definition 17.4 we can distinguish two cases.

  Case 1: $\psi(x)$ is defined based on Definition 17.4a. Then, since x occurs Λ-frozen in t, $\psi(x) = \chi(x)$ for some $\chi \in t^{-1}(\neg\langle\tau\rangle\top \wedge \overline{\varphi}')$. By Definition 17.1, $\chi(x) = \chi_1(x) \wedge \chi_2(x)$ with $\chi_1 \in t^{-1}(\neg\langle\tau\rangle\top)$ and $\chi_2 \in t^{-1}(\overline{\varphi}')$. By induction on formula size, $\chi_1(x) \in \mathbb{O}_{rb}^{s\equiv}$ and $\chi_2(x) \in \mathbb{O}_{rb}^{s\equiv}$. Hence, $\psi(x) = \chi_1(x) \wedge \chi_2(x)$ is in $\mathbb{O}_{rb}^{s\equiv}$.

  Case 2: $\psi(x)$ is defined based on Definition 17.4b, using an $\aleph\cap\Lambda$-impatient P-ruloid $\frac{H}{t \xrightarrow{\tau} u}$ and a $\chi \in u^{-1}(\langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \overline{\varphi}'))$. As the occurrence of x in t is Λ-frozen, $\psi(x) = \chi(x) \wedge \bigwedge_{x \xrightarrow{\beta} y \in H} \langle\beta\rangle\chi(y) \wedge \bigwedge_{x \xrightarrow{\gamma} \not\rightarrow \in H} \neg\langle\gamma\rangle\top$. By induction on the construction of ψ, $\chi(x) \in \mathbb{O}_{rb}^{\equiv}$. Moreover, by condition 1 of Definition 20 the y occur only Λ-liquid in u, so we proved before that the $\chi(y)$ are in $\mathbb{O}_b^{s\equiv}$. Hence $\psi(x) \in \mathbb{O}_{rb}^{s\equiv}$.

This completes the proof of the second claim of the proposition. We finish with the last claim. The cases where $\overline{\varphi}$ is of the form $\bigwedge_{i\in I}\overline{\varphi}_i$ or $\neg\overline{\varphi}'$ can be dealt with as in the proof of Prop. 3 in [15], and the case $\overline{\varphi} = \varphi \in \mathbb{O}_b$ follows immediately from the first claim. We therefore focus on the remaining case: $\overline{\varphi} = \langle\alpha\rangle\varphi$ with $\varphi \in \mathbb{O}_b$.

$\psi(x)$ is defined based on Definition 17.3, for some P-ruloid $\frac{H}{t \xrightarrow{\alpha} u}$ and $\chi \in u^{-1}(\varphi)$. Since the occurrence of x in t is $\aleph$-frozen, by condition 3 of Definition 20, x does not occur in H. Hence, $\psi(x) = \chi(x)$. By Proposition 30, $\frac{H}{t \xrightarrow{\alpha} u}$ is rooted

stability-respecting branching bisimulation safe. Since the occurrence of x in t is Λ-liquid, by condition 2 of Definition 20, x occurs only Λ-liquid in u. Therefore, by the first claim of this proposition, $\chi(x) \in \mathbb{O}_b^{\equiv}$. □

Now the promised congruence results for $\leftrightarrows_b^s$ and $\leftrightarrows_{rb}^s$ can be proved in the same way as their counterparts for $\leftrightarrows_b$ and $\leftrightarrows_{rb}$ in [15].

Theorem 32. *If P is a complete standard TSS in stability-respecting branching bisimulation format, then $\leftrightarrows_b^s$ is a congruence for P.* □

Theorem 33. *If P is a complete standard TSS in rooted stability-respecting branching bisimulation format, then $\leftrightarrows_{rb}^s$ is a congruence for P.* □

5. Divergence-preserving branching bisimilarity as a congruence

To obtain a modal characterisation of (weakly) divergence-preserving branching bisimilarity, a modality that captures divergence needs to be added to the modal logic. A modal characterisation of divergence-preserving branching bisimilarity would be obtained by adding a unary modality Δ to the modal logic for branching bisimilarity, with the interpretation $p \models \Delta\varphi$ if there is an infinite trace $p = p_0 \xrightarrow{\tau} p_1 \xrightarrow{\tau} p_2 \xrightarrow{\tau} \cdots$ such that $p_i \models \varphi$ for all $i \in \mathbb{N}$.

Modal formulas of the form $\Delta\varphi$, which capture divergence, elude the inductive decomposition method from Definition 17, because they ask for the existence of an infinite sequence of τ-transitions. The following example shows that the decomposition method does not readily extend to modalities $\Delta\varphi$.

Example 34. Let $A = \{a_i, \overline{a_i} \mid i \in \mathbb{N}\}$ and let $A_\iota = A \cup \{\iota\}$. The interleaving parallel composition operator $\|$ is usually defined by the following two rules, where α ranges over $A_\iota \cup \{\tau\}$:

$$\frac{x_1 \xrightarrow{\alpha} y}{x_1 \| x_2 \xrightarrow{\alpha} y \| x_2} \qquad \frac{x_2 \xrightarrow{\alpha} y}{x_1 \| x_2 \xrightarrow{\alpha} x_1 \| y}$$

We extend the operational semantics of this operator with communication rules, for all $i \in \mathbb{N}$:

$$\frac{x_1 \xrightarrow{a_i} y_1 \quad x_2 \xrightarrow{\overline{a_i}} y_2}{x_1 \| x_2 \xrightarrow{\iota} y_1 \| y_2} \qquad \frac{x_1 \xrightarrow{\overline{a_i}} y_1 \quad x_2 \xrightarrow{a_i} y_2}{x_1 \| x_2 \xrightarrow{\iota} y_1 \| y_2}$$

Consider the following two collections of processes p_i, q_i for $i \in \mathbb{N}$. We define $p_i \xrightarrow{\tau} p_{i+1}$ and $p_i \xrightarrow{a_i} \mathbf{0}$ and $q_i \xrightarrow{\tau} q_{i+1}$ and $q_i \xrightarrow{\overline{a_i}} \mathbf{0}$ for all $i \in \mathbb{N}$. We have $p_0 \| q_0 \models \Delta(\langle\varepsilon\rangle\langle\iota\rangle\top)$. There is no obvious way to decompose this modal property of $p_0 \| q_0$ into modal properties of its arguments p_0 and q_0.

Rather than modifying the decomposition method in order to derive new congruence formats for (weakly) divergence-preserving branching bisimilarity, we shall argue in this section that the stability-respecting branching bisimilarity format is also a congruence format for weakly divergence-preserving branching bisimilarity and divergence-preserving branching bisimilarity, and similarly for their rooted variants.

Suppose that P is a Γ-patient TSS (for some predicate Γ) on which $\leftrightarrows_b^s$ is a congruence, and suppose that we wish to show that $\leftrightarrows_b^{\Delta\top}$ is a congruence on P too. Then, using that $\leftrightarrows_b^{\Delta\top} \subseteq \leftrightarrows_b^s$, it suffices to argue that, for all $p_1, \ldots, p_n$ and $q_1, \ldots, q_n$ such that $p_i \leftrightarrows_b^{\Delta\top} q_i$ $(1 \leq i \leq n)$, we have that $f(p_1, \ldots, p_n) \leftrightarrows_b^s f(q_1, \ldots, q_n)$ implies $f(p_1, \ldots, p_n) \leftrightarrows_b^{\Delta\top} f(q_1, \ldots, q_n)$. The feature that distinguishes $\leftrightarrows_b^{\Delta\top}$ from $\leftrightarrows_b^s$ is that $\leftrightarrows_b^{\Delta\top}$ preserves divergences whereas $\leftrightarrows_b^s$ does not. Now suppose that $f(p_1, \ldots, p_n)$ admits a divergence. Then we can distinguish two cases:

1. the process $f(p_1, \ldots, p_n)$ directly inherits this divergence from one of its arguments through a patience rule, say, p_i admits a divergence and the ith argument of f is Γ-liquid; or
2. none of the p_i for i a Γ-liquid argument of f admits a divergence, but somehow there is an interplay between f and its arguments $p_1, \ldots, p_n$ that facilitates the divergence.

We want to argue that in both cases $f(q_1, \ldots, q_n)$ necessarily admits a divergence as well. In the first case, using that $p_i \leftrightarrows_b^{\Delta\top} q_i$, also q_i admits a divergence. So by the patience rule for the ith argument of f, $f(q_1, \ldots, q_n)$ admits a divergence. It hence suffices to reduce the second case to the first. The following example roughly illustrates how such a reduction can be achieved.

Example 35. Let $A = \{a, \overline{a}\}$ and consider the parallel composition operator $|$ of CCS for this set of actions A defined by the following four rules (with α ranging over A_τ):

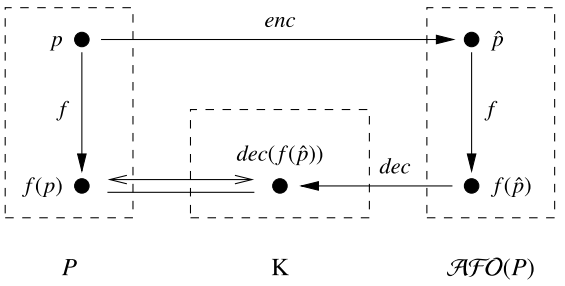

Fig. 2. Proof idea.

$$\frac{x_1 \xrightarrow{\alpha} y}{x_1 \mid x_2 \xrightarrow{\alpha} y \mid x_2} \qquad \frac{x_2 \xrightarrow{\alpha} y}{x_1 \mid x_2 \xrightarrow{\alpha} x_1 \mid y}$$

$$\frac{x_1 \xrightarrow{a} y_1 \quad x_2 \xrightarrow{\bar{a}} y_2}{x_1 \mid x_2 \xrightarrow{\tau} y_1 \mid y_2} \qquad \frac{x_1 \xrightarrow{\bar{a}} y_1 \quad x_2 \xrightarrow{a} y_2}{x_1 \mid x_2 \xrightarrow{\tau} y_1 \mid y_2}$$

Note that the operational semantics of $\mid$ includes patience rules for both arguments (specific instances of the two left-most rules), but also rules that may lead to divergences not directly inherited from arguments. For instance, if p_1 is the process with just one transition $p_1 \xrightarrow{a} p_1$ and p_2 is the process with just one transition $p_2 \xrightarrow{\bar{a}} p_2$, then $p_1 \mid p_2$ admits a divergence, while p_1 and p_2 do not admit divergences.

Now also consider the parallel composition operator $\parallel$ defined in Ex. 34 in combination with a unary abstraction operator τ_ι defined by the following two rules, where α ranges over A_τ (i.e., $\alpha \neq \iota$):

$$\frac{x \xrightarrow{\alpha} y}{\tau_\iota(x) \xrightarrow{\alpha} \tau_\iota(y)} \qquad \frac{x \xrightarrow{\iota} y}{\tau_\iota(x) \xrightarrow{\tau} \tau_\iota(y)}$$

It is not hard to see that in a TSS P that includes the CCS parallel composition $\mid$, the parallel composition operator $\parallel$ as defined in Ex. 34 and the unary abstraction operator τ_ι, we have, for all processes p_1 and p_2, that $p_1 \mid p_2 \leftrightarroweq \tau_\iota(p_1 \parallel p_2)$. Hence, since $\leftrightarroweq \subseteq \leftrightarroweq_b^{\Delta\top}$, in order to show that $\leftrightarroweq_b^{\Delta\top}$ is a congruence for $\mid$ it suffices to establish that it is a congruence for $\parallel$ and that it is a congruence for τ_ι. Namely, then $p_1 \leftrightarroweq_b^{\Delta\top} q_1$ and $p_2 \leftrightarroweq_b^{\Delta\top} q_2$ yield $p_1 \mid p_2 \leftrightarroweq \tau_\iota(p_1 \parallel p_2) \leftrightarroweq_b^{\Delta\top} \tau_\iota(q_1 \parallel q_2) \leftrightarroweq q_1 \mid q_2$. Moreover, it is well-known and easy to see that τ_ι is compositional for a wide range of behavioural equivalences, including the ones mentioned in Sect. 2.1. Since all operational rules for $\parallel$ with a τ-labelled conclusion are patience rules, only case 1 of $p_1 \parallel p_2$ admitting a divergence can apply.

The idea discussed in the example above can be generalised: every operator f can be expressed as a combination of τ_ι and an *abstraction-free variant* f' of f, for which all rules with a τ-labelled conclusion are patience rules. The main ingredient of our proof that on every TSS P in the stability-respecting branching bisimulation format both $\leftrightarroweq_b^{\Delta\top}$ and $\leftrightarroweq_b^{\Delta}$ are congruences, and similarly for the rooted case, is a transformation that allows us to establish compositionality of an operator f by establishing the compositionality of its abstraction-free variant.

Although the idea is fairly simple, the formal technicalities are quite involved. It is therefore convenient to first formulate, in Sect. 5.1, sufficient conditions on a presupposed transformation on TSSs and associated encoding and decoding functions for them to be suitable for lifting a congruence format for some behavioural equivalence to a finer equivalence. Instead of dealing with specific equivalences $\leftrightarroweq_b^{\Delta}$ and $\leftrightarroweq_b^{s}$, we work with parametric equivalences $\sim$ and $\approx$ where $\sim$ is finer than $\approx$; they will later be instantiated with $\leftrightarroweq_b^{\Delta}$ and $\leftrightarroweq_b^{s}$. This allows a reuse of our work with $\leftrightarroweq_b^{\Delta\top}$ and $\leftrightarroweq_b^{s}$ in the roles of $\sim$ and $\approx$, as well as with $\leftrightarroweq_{rb}^{\Delta\top}$ and $\leftrightarroweq_{rb}^{\Delta}$ in the role of $\sim$ and $\leftrightarroweq_{rb}^{s}$ in the role of $\approx$. In Sect. 5.2 we introduce the machinery needed in Sect. 5.1 and show that it has the required properties, except for those that depend on the choice of $\sim$ and $\approx$. Finally, in Sect. 5.3 we apply our framework to derive congruence formats for $\leftrightarroweq_b^{\Delta\top}$, $\leftrightarroweq_b^{\Delta}$, $\leftrightarroweq_{rb}^{\Delta\top}$ and $\leftrightarroweq_{rb}^{\Delta}$.

5.1. A general framework for lifting congruence formats to finer equivalences

Our general proof idea is illustrated in Fig. 2. Here P (on the left) is a TSS in our congruence format for $\approx$. We want to show that on P also $\sim$ is a congruence. So consider an operator f, for simplicity depicted as unary. Given two processes p and q in P (closed terms in its signature) with $p \sim q$, we need to show that $f(p) \sim f(q)$. Fig. 2 shows a roundabout trajectory from p to $f(p)$. Imagine a similar trajectory from q to $f(q)$—not depicted in Fig. 2, but hovering above the page.

First we apply a transformation $\mathcal{AFO}$ on P. The TSS $\mathcal{AFO}(P)$ will be *abstraction-free* in the sense that it only allows patience rules and rules without premises to carry a conclusion with the label τ. Moreover, the transformation $\mathcal{AFO}$ needs to be such that on $\mathcal{AFO}(P)$ the behavioural equivalences $\sim$ and $\approx$ coincide, and to realise this property it will introduce

oracle transitions that reveal some pertinent information on the behaviour of a process, such as whether it can diverge. We ensure that $\sim$-equivalence is preserved under the addition of these oracle transitions.

The transformation $\mathcal{AFO}$ will modify the rules of P, but we still want to find for each closed term p of P a faithful representant of p inside $\mathcal{AFO}(P)$. To this end we introduce for each closed term p of P a constant $\hat{p}$ in $\mathcal{AFO}(P)$, in such a way that $p \sim q$ implies $\hat{p} \sim \hat{q}$.[1] Note that by including these processes as constants, added oracle transition $\hat{p} \xrightarrow{\omega} \surd$ are decent ntyft rules without premises, which therefore do not compromise the $\approx$-congruence format that $\mathcal{AFO}(P)$ needs to satisfy. Each n-ary operator f of P remains an n-ary operator f of $\mathcal{AFO}(P)$. Since $\sim \subseteq \approx$ we have $\hat{p} \approx \hat{q}$. We argue that if P is within our congruence format for $\approx$, then the TSS $\mathcal{AFO}(P)$ is also within this congruence format, and conclude from $\hat{p} \approx \hat{q}$ that $f(\hat{p}) \approx f(\hat{q})$. An important result, deferred to Sect. 5.3, is that on $\mathcal{AFO}(P)$ the equivalences $\sim$ and $\approx$ coincide. Hence $f(\hat{p}) \sim f(\hat{q})$.

Finally, we decode the processes $f(\hat{p})$ and $f(\hat{q})$, aiming to return to the LTS generated by P, but actually ending up in another LTS K. Our decoding function *dec* exactly undoes the effects of the encoding *enc*, which sent p to $\hat{p}$, so that $dec(f(\hat{p}))$ is strongly bisimilar with $f(p)$. A crucial property of the function *dec*, also deferred to Sect. 5.3, is that it is compositional for $\sim$, meaning that from $f(\hat{p}) \sim f(\hat{q})$ we may conclude $dec(f(\hat{p})) \sim dec(f(\hat{q}))$. By imposing the requirement that $\sim$ contains strong bisimilarity, this implies that $f(p) \sim f(q)$.

We now formalise this proof idea. An LTS G is called *disjoint* from a complete TSS P if it is disjoint from the LTS G_P associated with P. In that case $P \uplus G$ denotes the union of G_P and G (cf. Definition 4).

Theorem 36. *Let $\sim$ and $\approx$ be behavioural equivalences on LTSs, with $\leftrightarrows \subseteq \sim \subseteq \approx$. Let $\mathfrak{F}$ be a congruence format for $\approx$, included in the decent ntyft format, and let $\mathcal{AFO}$ be an operation on standard TSSs, where for each TSS $P = (\Sigma, Act, R)$ the signature $\hat{\Sigma}$ of $\mathcal{AFO}(P)$ contains Σ enriched by a fresh constant $\hat{p}$ for each closed term p in $T(\Sigma)$, such that, for each complete standard TSS P in decent ntyft format:*

1. *also $\mathcal{AFO}(P)$ is a complete standard TSS,*
2. *if P is in $\mathfrak{F}$-format then so is $\mathcal{AFO}(P)$,*
3. *$p \sim_P q \;\Rightarrow\; \hat{p} \sim_{\mathcal{AFO}(P)} \hat{q}$,*
4. *$\sim_{\mathcal{AFO}(P)}$ and $\approx_{\mathcal{AFO}(P)}$ coincide, and*

there is an LTS $\mathrm{K} = (\mathbb{P}_\mathrm{K}, Act_\mathrm{K}, \rightarrow_\mathrm{K})$, disjoint from P, as well as a function $dec : T(\hat{\Sigma}) \to \mathbb{P}_\mathrm{K}$ such that:

5. *$p \sim_{\mathcal{AFO}(P)} q \;\Rightarrow\; dec(p) \sim_\mathrm{K} dec(q)$, and*
6. *$f(p_1, \ldots, p_n) \leftrightarrows_{P \uplus \mathrm{K}} dec(f(\hat{p}_1, \ldots, \hat{p}_n))$ for any n-ary $f \in \Sigma$ and $p_1, \ldots, p_n \in T(\Sigma)$.*

Then $\mathfrak{F}$ is also a congruence format for $\sim$.

Proof. Let $P = (\Sigma, Act, R)$ be a complete standard TSS in $\mathfrak{F}$-format. We will show that $\sim_P$ is a congruence for P. So let $f \in \Sigma$ be an n-ary function symbol, and let $p_i, q_i \in \mathrm{T}(\Sigma)$ with $p_i \sim_P q_i$ for $i = 1, \ldots, n$. We need to show that $f(p_1, \ldots, p_n) \sim_P f(q_1, \ldots, q_n)$.

By requirements 1 and 2, $\mathcal{AFO}(P)$ is a complete standard TSS in $\mathfrak{F}$-format; hence $\approx_{\mathcal{AFO}(P)}$ is a congruence for $\mathcal{AFO}(P)$. By requirement 3 $\hat{p}_i \sim_{\mathcal{AFO}(P)} \hat{q}_i$ for $i = 1, \ldots, n$. By requirement 4 also $\sim_{\mathcal{AFO}(P)}$ is a congruence for $\mathcal{AFO}(P)$. So $f(\hat{p}_1, \ldots, \hat{p}_n) \sim_{\mathcal{AFO}(P)} f(\hat{q}_1, \ldots, \hat{q}_n)$. Hence, by requirement 5,

$$dec(f(\hat{p}_1, \ldots, \hat{p}_n)) \sim_\mathrm{K} dec(f(\hat{q}_1, \ldots, \hat{q}_n)).$$

Therefore, by two applications of requirement 6, and the definition of a behavioural equivalence, $f(p_1, \ldots, p_n) \sim_{P \uplus \mathrm{K}} f(q_1, \ldots, q_n)$ and so $f(p_1, \ldots, p_n) \sim_P f(q_1, \ldots, q_n)$. □

5.2. Abstraction-freeness

In this section we introduce the machinery needed for Theorem 36, namely the conversion $\mathcal{AFO}$ on TSSs and the function *dec* into the LTS K. We also establish requirements 1 and 6 of Theorem 36, leaving 2–5 to the applications of Theorem 36 in Sect. 5.3 for specific instances of $\sim$ and $\approx$. Since we are only interested in TSSs with τ-transitions, we here take the set of actions Act used in Sect. 5.1 to be A_τ.

5.2.1. The conversion $\mathcal{AFO}$

Let again Γ denote a predicate that marks the (Γ-liquid) arguments of function symbols. In [14] we called a standard TSS *abstraction-free* w.r.t. Γ if only its Γ-patience rules carry the label τ in their conclusion. Here we use a slightly more

[1] In all our applications we have $p \sim q$ iff $\hat{p} \sim \hat{q}$, but our general framework does not require this.

liberal definition of abstraction-freeness that also allows rules that have no premises, and a conclusion of the form $c \xrightarrow{\tau} d$ for constants c and d.

The following conversion turns any Γ-patient standard TSS $P = (\Sigma, A_\tau, R)$ into a Γ-patient and abstraction-free TSS $\mathcal{AFO}_\Gamma^{O,\zeta}(P)$. It is parameterised by the choice of a fresh set of actions O, so $O \cap A_\tau = \emptyset$, and a partial function $\zeta : \mathrm{T}(\Sigma) \rightharpoonup O$, which we call an *oracle*. The choice of O and ζ varies for different applications of Theorem 36. This choice will be made in Sect. 5.3 in such a way that requirements 3 and 4 of Theorem 36 are met, for specific instances of $\sim$ and $\approx$.

Definition 37. Given a Γ-patient standard TSS $P = (\Sigma, A_\tau, R)$. Let $\hat{\Sigma}$ be the signature Σ, enriched with a fresh constant $\surd$ and a fresh constant $\hat{p}$ for each closed term $p \in \mathrm{T}(\Sigma)$. Pick a fresh action $\iota \notin A_\tau \cup O$. We define the TSS $\mathcal{AFO}_\Gamma^{O,\zeta}(P)$ as $(\hat{\Sigma}, A_\tau \cup O \cup \{\iota\}, R')$ where the rules in R' are obtained from the rules in R as follows:

1. R_1 is obtained from R by adding for each rule r and each non-empty subset S of positive τ-premises of r, a copy of r with as only difference that the labels τ in the premises in S are replaced by ι;
2. R_2 is obtained from R_1 by replacing, in every rule that has a conclusion with the label τ and is not a Γ-patience rule, the τ-label in the conclusion by ι;
3. R_3 is obtained from R_2 by adding the premise $v \xrightarrow{\iota}\!\!\!\!\!\!/\;$ to each rule with a negative τ-premise $v \xrightarrow{\tau}\!\!\!\!\!\!/\;$;
4. R_4 is obtained from R_3 by the addition of a rule $\hat{p} \xrightarrow{\alpha} \hat{q}$ for each transition $p \xrightarrow{\alpha} q$ *ws*-provable from P;
5. R_5 is obtained from R_4 by the addition of a rule $\hat{p} \xrightarrow{\zeta(p)} \surd$ for each $p \in \mathrm{T}(\Sigma)$ with $\zeta(p)$ defined;
6. R' adds to R_5 a rule $\dfrac{x_k \xrightarrow{\omega} y}{f(x_1, \ldots, x_n) \xrightarrow{\omega} y}$ for each $\omega \in O$, $f \in \Sigma$ and argument k with $\Gamma(f, k)$.

Step 2 above makes the resulting TSS abstraction-free by renaming τ-labels in conclusions of non-patience rules into ι. To ensure that still the same transitions are derived, modulo the conversion of some τ into ι-labels, step 1 above allows positive premises labelled ι to be used instead of τ in all rules, and step 3 achieves the same purpose for negative premises. These three steps result in a Γ-patient and abstraction-free TSS that could be called $\mathcal{AF}_\Gamma(P)$.

Conceptually, steps 1, 2 and 3 on the one hand and steps 4 and 5 on the other hand are independent. Listing steps 1, 2 and 3 before steps 4 and 5 makes it evident that the transformations 1–3 do not apply to the rules added by 4 and 5.

For convenience in proofs, and to better explain steps 4 and 5 of Definition 37, we consider an auxiliary LTS $\mathrm{G} = (\mathbb{P}_\mathrm{G}, A_\tau, \rightarrow_\mathrm{G})$ with $\mathbb{P}_\mathrm{G} = \{\hat{p} \mid p \in \mathrm{T}(\Sigma)\}$ and $\hat{p} \xrightarrow{\alpha}_\mathrm{G} \hat{q}$ iff $P \vdash_{ws} p \xrightarrow{\alpha} q$, and an auxiliary LTS $\mathrm{H} = (\mathbb{P}_\mathrm{H}, A_\tau \cup O, \rightarrow_\mathrm{H})$ with $\mathbb{P}_\mathrm{H} = \mathbb{P}_\mathrm{G} \cup \{\surd\}$ and $\rightarrow_\mathrm{H} := \rightarrow_\mathrm{G} \cup \{\hat{p} \xrightarrow{\zeta(p)} \surd \mid \zeta(p) \text{ defined}\}$. The LTS G is simply a disjoint copy of the LTS generated by P.

Lemma 38. *If $p \sim_P q$ for some $p, q \in T(\Sigma)$, then $\hat{p} \sim_\mathrm{G} \hat{q}$.*

Proof. We have $p \leftrightarrows_{P \uplus \mathrm{G}} \hat{p}$ for each $p \in \mathrm{T}(\Sigma)$, because the relation $\{(p, \hat{p}), (\hat{p}, p) \mid p \in \mathrm{T}(\Sigma)\}$ is a strong bisimulation. So $\hat{p} \leftrightarrows_{P \uplus \mathrm{G}} p \sim_{P \uplus \mathrm{G}} q \leftrightarrows_{P \uplus \mathrm{G}} \hat{q}$, using the definition of a behavioural equivalence, and thus $\hat{p} \sim_{P \uplus \mathrm{G}} \hat{q}$, using that $\leftrightarrows \subseteq \sim$. Hence $\hat{p} \sim_\mathrm{G} \hat{q}$, again by the definition of a behavioural equivalence. □

The LTS H adds *oracle transitions* to G. The idea is that $\zeta(p)$ is particular for the $\sim$-equivalence class of $p \in \mathrm{T}(\Sigma)$, which on the one hand ensures that $\hat{p} \sim_\mathrm{H} \hat{q}$ iff $\hat{p} \sim_\mathrm{G} \hat{q}$, and on the other hand enforces that $\sim$ and $\approx$ coincide on H. Namely, if $\hat{p} \approx_\mathrm{H} \hat{q}$ then the oracle action of $\hat{p}$ can be matched by $\hat{q}$ (and vice versa), which implies $\hat{p} \sim_\mathrm{H} \hat{q}$.

Example 39. Take $\sim$ to be weakly divergence-preserving branching bisimilarity, and $\approx$ to be stability-respecting branching bisimilarity. Let us say that a process p in an LTS is *divergent* if there exists an infinite sequence of processes $(p_k)_{k \in \mathbb{N}}$ such that $p = p_0$ and $p_k \xrightarrow{\tau} p_{k+1}$ for all $k \in \mathbb{N}$, i.e. if $p \models \Delta\top$. Take $O = \{\Delta\top\}$ and let $\zeta(p) = \Delta\top$ iff p is divergent. Thus in H all divergent states of G have a fresh outgoing transition labelled $\Delta\top$.

With this definition of H, we have $\hat{p} \sim_\mathrm{H} \hat{q}$ iff $\hat{p} \sim_\mathrm{G} \hat{q}$: any weakly divergence-preserving branching bisimulation $\mathcal{B}$ on G relates divergent states with divergent states only, and thus is also a weakly divergence-preserving branching bisimulation on H (when adding $\surd\ \mathcal{B}\ \surd$). Furthermore, as we will show in Proposition 54, due to the construction of H, $\hat{p} \sim_\mathrm{H} \hat{q}$ iff $\hat{p} \approx_\mathrm{H} \hat{q}$.

Steps 4 and 5 of Definition 37 incorporate the entire LTS H into $\mathcal{AF}_\Gamma(P)$: each state appears as a constant and each transition appears as a rule without premises. The operators from Σ can now be applied to arguments of the form $\hat{p}$. Finally, step 6 lets any term $f(x_1, \ldots, x_n)$ inherit the oracle transitions from its Γ-liquid arguments. Steps 4, 5 and 6 preserve abstraction-freeness; for step 4 this uses the relaxed definition of abstraction-freeness that allows to incorporate τ-transitions between constants as rules without premises.

Example 40. Let P have the three rules

$$\frac{x_1 \xrightarrow{\tau} y}{g(x_1,x_2,x_3) \xrightarrow{\tau} g(y,x_2,x_3)} \qquad \frac{x_1 \xrightarrow{a} y_1 \quad x_1 \xrightarrow{\tau} y_2 \quad x_3 \xrightarrow{\tau} y_3}{g(x_1,x_2,x_3) \xrightarrow{\tau} x_2}$$

$$\frac{x_2 \xrightarrow{\tau} y \quad x_3 \xrightarrow{\tau\!\!\!/}}{g(x_1,x_2,x_3) \xrightarrow{a} y}$$

where $\Gamma(g,1)$. Then $\mathcal{AFO}_\Gamma^{O,\zeta}(P)$ has the nine rules

$$\frac{x_1 \xrightarrow{\tau} y}{g(x_1,x_2,x_3) \xrightarrow{\tau} g(y,x_2,x_3)} \qquad \frac{x_1 \xrightarrow{\iota} y}{g(x_1,x_2,x_3) \xrightarrow{\iota} g(y,x_2,x_3)}$$

$$\frac{x_1 \xrightarrow{a} y_1 \quad x_1 \xrightarrow{\tau} y_2 \quad x_3 \xrightarrow{\tau} y_3}{g(x_1,x_2,x_3) \xrightarrow{\iota} x_2} \qquad \frac{x_1 \xrightarrow{a} y_1 \quad x_1 \xrightarrow{\iota} y_2 \quad x_3 \xrightarrow{\iota} y_3}{g(x_1,x_2,x_3) \xrightarrow{\iota} x_2}$$

$$\frac{x_1 \xrightarrow{a} y_1 \quad x_1 \xrightarrow{\iota} y_2 \quad x_3 \xrightarrow{\iota} y_3}{g(x_1,x_2,x_3) \xrightarrow{\iota} x_2}$$

$$\frac{x_2 \xrightarrow{\tau} y \quad x_3 \xrightarrow{\tau\!\!\!/} \quad x_3 \xrightarrow{\iota\!\!\!/}}{g(x_1,x_2,x_3) \xrightarrow{a} y} \qquad \frac{x_2 \xrightarrow{\iota} y \quad x_3 \xrightarrow{\tau\!\!\!/} \quad x_3 \xrightarrow{\iota\!\!\!/}}{g(x_1,x_2,x_3) \xrightarrow{a} y}$$

$$\frac{x_1 \xrightarrow{\omega} y}{g(x_1,x_2,x_3) \xrightarrow{\omega} y} \quad (\omega \in O)$$

By step 1, the first and third rule of P spawn one copy and the second rule of P spawns three copies in which part or all of the τ-labels of premises are renamed into ι. By step 2, in each rule that has a τ-label in the conclusion, except the patience rule for the first argument of g, this label is renamed into ι. By step 3, in the third rule and its copy, a negative ι-premise is added. By step 6, an ω-rule is added for the first argument of g. Since there are no closed terms in this example, steps 4 and 5 are void.

Clearly, for any Γ-patient standard TSS P, the standard TSS $\mathcal{AFO}_\Gamma^{O,\zeta}(P)$ is Γ-patient and abstraction-free w.r.t. Γ. Henceforth, the superscripts O and ζ are dropped and $\mathcal{AFO}(P)$ denotes $\mathcal{AFO}_\Gamma(P)$ for the largest predicate Γ for which P is Γ-patient. The signature of $\mathcal{AFO}(P)$ contains the signature of P enriched by a fresh constant $\hat{p}$ for each closed term p in P, as required in Theorem 36.

Lemma 41. *Let P be a complete standard TSS in ntyft format. If $\hat{p} \sim_{\mathrm{H}} \hat{q}$ for some $p,q \in T(\Sigma)$, then $\hat{p} \sim_{\mathcal{AFO}(P)} \hat{q}$.*

Proof. Since $\mathcal{AFO}(P)$ has no rules whose source is a variable, the only derivable transitions of the form $\hat{p} \xrightarrow{\alpha} q^*$ with $\alpha \in A_\tau \cup O \cup \{\iota\}$ are the ones from H, with q^* of the form $\hat{q}$ or $\surd$. For this reason any process $\hat{p}$ in H is strongly bisimilar to the process $\hat{p}$ in $\mathcal{AFO}(P)$. Using this, the lemma follows just as Lemma 38. □

As an immediate consequence of Lemma 38 and Lemma 41 we have the following corollary.

Corollary 42. *Requirement 3 of Theorem 36 is met if $\hat{p} \sim_{\mathrm{G}} \hat{q}$ implies $\hat{p} \sim_{\mathrm{H}} \hat{q}$.* □

The inference $\hat{p} \sim_{\mathrm{G}} \hat{q} \Rightarrow \hat{p} \sim_{\mathrm{H}} \hat{q}$ depends on the choice of O and ζ, and is deferred to Sect. 5.3.

The oracle inheritance rules in $\mathcal{AFO}_\Gamma(P)$—introduced in step 6 of Definition 37—ensure that a closed term $f(p_1,\ldots,p_n)$ has an outgoing ω-transition, for $\omega \in O$, iff one of its Γ-liquid arguments p_k has such a transition. Ultimately, all such oracle transitions stem from H. Using that in $\mathcal{AFO}_\Gamma(P)$ any term $p \in \mathrm{T}(\hat{\Sigma})$ can uniquely (up to renaming of variables) be written as $\rho(t)$ with $t \in \mathbb{T}(\Sigma)$ and $\rho : var(t) \to \mathbb{P}_{\mathrm{H}}$, this observation can be phrased as follows.

Observation 43. *Let $t \in \mathbb{T}(\Sigma)$ and $\rho : var(t) \to \mathbb{P}_{\mathrm{H}}$; then $\mathcal{AFO}_\Gamma(P) \vdash_{ws} \rho(t) \xrightarrow{\omega}$ iff t has a Γ-liquid occurrence of a variable x with $\rho(x) \xrightarrow{\omega}_{\mathrm{H}}$.* □

We now compare provability in a standard TSS P with provability in its abstraction-free counterpart $\mathcal{AFO}(P)$, in order to verify requirements 1 and 6 of Theorem 36. The most laborious part in this comparison lays in step 4 of Definition 37. Therefore we deal with this step separately from the other five. Hence, we define for any standard TSS $P = (\Sigma, A_\tau, R)$ a TSS $\mathcal{G}(P) = (\hat{\Sigma}, A_\tau, R'')$, which constitutes an intermediate between P and $\mathcal{AFO}(P)$. It is built by only applying step 4 from the construction of $\mathcal{AFO}(P)$, with R in the role of R_3.

5.2.2. Comparison of P and $\mathcal{G}(P)$

We restrict our analysis to standard TSSs $P = (\Sigma, A_\tau, R)$ in decent ntyft format. For $p \in \mathrm{T}(\hat{\Sigma})$, let $\check{p} \in \mathrm{T}(\Sigma)$ be obtained from p by replacing every subterm $\hat{q}$ in p by q. Likewise, for N a set of closed negative literals, $\check{N} := \{\check{p} \stackrel{\alpha}{\not\longrightarrow} \mid (p \stackrel{\alpha}{\not\longrightarrow}) \in N\}$.

When we are merely interested in proving, from P or $\mathcal{G}(P)$, literals of the form $p \stackrel{\alpha}{\not\longrightarrow}$ or $p \stackrel{\alpha}{\longrightarrow}$, possibly under some hypotheses, where the target of a positive literal is existentially quantified, we can equivalently use a version of P resp. $\mathcal{G}(P)$ in which all right-hand sides of the premises and conclusions of rules have been dropped. Here we use that P and $\mathcal{G}(P)$ are in decent ntyft format. This we do below.

Lemma 44. *Let $P = (\Sigma, A_\tau, R)$ be a standard TSS in decent ntyft format. Let $\alpha \in A_\tau$ and $p \in T(\hat{\Sigma})$.*

- *If $\mathcal{G}(P) \vdash_{irr} \dfrac{N}{p \stackrel{\alpha}{\longrightarrow}}$, with N a set of closed negative literals, then $P \vdash_{irr} \dfrac{\check{N} \cup N'}{\check{p} \stackrel{\alpha}{\longrightarrow}}$ for a set N' of closed negative literals λ with $P \vdash_{ws} \lambda$.*
- *If $P \vdash_{irr} \dfrac{N'}{\check{p} \stackrel{\alpha}{\longrightarrow}}$, with N' a set of closed negative literals, then either $P \not\vdash_{ws} \lambda$ for a literal $\lambda \in N'$, or $\mathcal{G}(P) \vdash_{irr} \dfrac{N}{p \stackrel{\alpha}{\longrightarrow}}$ for some N with $\check{N} \subseteq N'$.*

Proof. For the first statement we apply induction on the irredundant proof π of $\frac{N}{p \stackrel{\alpha}{\longrightarrow}}$ from $\mathcal{G}(P)$.

First assume that p has the form $\hat{q}$, so that $\check{p} = q$. Since $\mathcal{G}(P)$ has no rules whose source is a variable, the last step of π must be an application of a rule $\hat{q} \stackrel{\alpha}{\longrightarrow} \hat{r}$, introduced in step 4 of the construction of $\mathcal{G}(P)$, and $N = \emptyset$. It follows that $P \vdash_{ws} q \stackrel{\alpha}{\longrightarrow} r$. Let N' be the unique set of negative literals occurring in the well-supported proof of $q \stackrel{\alpha}{\longrightarrow} r$ from P that have no negative literals below them. Then $P \vdash_{irr} \frac{N'}{q \stackrel{\alpha}{\longrightarrow} r}$ and each literal $\lambda \in N'$ is *ws*-provable from P.

Alternatively, $p = f(p_1, \ldots, p_n)$ with $f \in \Sigma$. Then the last step of π must be an application of a decent ntyft rule $\frac{H}{f(x_1, \ldots, x_n) \stackrel{\alpha}{\longrightarrow}}$ in P and the substitution $\sigma : \{x_1, \ldots, x_n\} \to \mathrm{T}(\hat{\Sigma})$ with $\sigma(x_i) = p_i$ for $i = 1, \ldots, n$. Let $\sigma' : \{x_1, \ldots, x_n\} \to \mathrm{T}(\Sigma)$ be the substitution with $\sigma'(x_i) = \check{p}_i$ for $i = 1, \ldots, n$. With every positive premise $\mu = (t \stackrel{\gamma}{\longrightarrow})$ in H we may associate a set of closed negative literals $N_\mu \subseteq N$ such that $\mathcal{G}(P) \vdash_{irr} \frac{N_\mu}{\sigma(t) \stackrel{\gamma}{\longrightarrow}}$ and such that N is the union of all N_μ. So, by induction, $P \vdash_{irr} \frac{\check{N}_\mu \cup N'_\mu}{\sigma'(t) \stackrel{\gamma}{\longrightarrow}}$ for a set N'_μ of closed negative literals λ with $P \vdash_{ws} \lambda$. Let N' be the union of all the N'_μ. Note that $\check{N}$ is the union of all the $\check{N}_\mu$ and the literals $\sigma'(u) \stackrel{\gamma}{\not\longrightarrow}$ with $(u \stackrel{\gamma}{\not\longrightarrow}) \in H$. It follows that $P \vdash_{irr} \frac{\check{N} \cup N'}{\check{p} \stackrel{\alpha}{\longrightarrow}}$.

For the second statement assume $P \vdash_{irr} \frac{N'}{\check{p} \stackrel{\alpha}{\longrightarrow}}$ and $P \vdash_{ws} \lambda$ for all $\lambda \in N'$. Then $P \vdash_{ws} \check{p} \stackrel{\alpha}{\longrightarrow}$. We show that $\mathcal{G}(P) \vdash_{irr} \frac{N}{p \stackrel{\alpha}{\longrightarrow}}$ for some N with $\check{N} \subseteq N'$, by induction on the irredundant proof π of $\frac{N'}{\check{p} \stackrel{\alpha}{\longrightarrow}}$ from P. First assume that p has the form $\hat{q}$, so that $\check{p} = q$. Then $P \vdash_{ws} q \stackrel{\alpha}{\longrightarrow}$, so $\mathcal{G}(P) \vdash_{irr} \hat{q} \stackrel{\alpha}{\longrightarrow}$, meeting the requirement of the lemma.

Alternatively, $p = f(p_1, \ldots, p_n)$ with $f \in \Sigma$, so that $\check{p} = f(\check{p}_1, \ldots, \check{p}_n)$. Then the last step of π must be an application of a decent ntyft rule $\frac{H}{f(x_1, \ldots, x_n) \stackrel{\alpha}{\longrightarrow}}$ and the substitution $\sigma' : \{x_1, \ldots, x_n\} \to \mathrm{T}(\Sigma)$ with $\sigma'(x_i) = \check{p}_i$ for $i = 1, \ldots, n$. Let $\sigma : \{x_1, \ldots, x_n\} \to \mathrm{T}(\hat{\Sigma})$ be the substitution with $\sigma(x_i) = p_i$ for $i = 1, \ldots, n$. For each positive premise $\mu = (t \stackrel{\gamma}{\longrightarrow})$ in H we have $P \vdash_{irr} \frac{N'_\mu}{\sigma'(t) \stackrel{\gamma}{\longrightarrow}}$ for some $N'_\mu \subseteq N'$ and $P \vdash_{ws} \lambda$ for all $\lambda \in N'_\mu$, and thus, by induction, $\mathcal{G}(P) \vdash_{irr} \frac{N_\mu}{\sigma(t) \stackrel{\gamma}{\longrightarrow}}$ for some N_μ with $\check{N}_\mu \subseteq N'_\mu$. Let N be the union of all the N_μ, and the literals $\sigma(u) \stackrel{\gamma}{\not\longrightarrow}$ with $(u \stackrel{\gamma}{\not\longrightarrow}) \in H$. Then $\check{N} \subseteq N'$. It follows that $\mathcal{G}(P) \vdash_{irr} \frac{N}{p \stackrel{\alpha}{\longrightarrow}}$. □

Proposition 45. *Let $P = (\Sigma, A_\tau, R)$ be a complete standard TSS in decent ntyft format, $\alpha \in A_\tau$ and $p \in T(\hat{\Sigma})$.*

- *$P \vdash_{ws} \check{p} \stackrel{\alpha}{\longrightarrow} q^* \;\Leftrightarrow\; \exists q \in T(\hat{\Sigma}).\, q^* = \check{q} \wedge \mathcal{G}(P) \vdash_{ws} p \stackrel{\alpha}{\longrightarrow} q$.*
- *$P \vdash_{ws} \check{p} \stackrel{\alpha}{\not\longrightarrow} \;\Leftrightarrow\; \mathcal{G}(P) \vdash_{ws} p \stackrel{\alpha}{\not\longrightarrow}$.*

Proof. "⇒": We prove both statements by simultaneous induction on the well-founded proof π from $\check{p} \stackrel{\alpha}{\longrightarrow} q^*$ or $\check{p} \stackrel{\alpha}{\not\longrightarrow}$ from P. First consider the case $P \vdash_{ws} \check{p} \stackrel{\alpha}{\longrightarrow} q^*$.

Assume that p has the form $\hat{r}$, so that $\check{p} = r$. It follows immediately from step 4 of the construction of $\mathcal{G}(P)$ that $\mathcal{G}(P) \vdash_{ws} \hat{r} \stackrel{\alpha}{\longrightarrow} \hat{q}^*$. So take $q := \hat{q}^*$.

Alternatively, $p = f(p_1, \ldots, p_n)$ with $f \in \Sigma$, so that $\check{p} = f(\check{p}_1, \ldots, \check{p}_n)$. Then the last step of π must be an application of a decent ntyft rule $\frac{H}{f(x_1, \ldots, x_n) \stackrel{\alpha}{\longrightarrow} t}$ and a substitution $\sigma' : V \rightharpoonup \mathrm{T}(\Sigma)$ with $\sigma'(x_i) = \check{p}_i$ for $i = 1, \ldots, n$ and $\sigma'(t) = q^*$. Let

$\sigma'' : \{x_1, \ldots, x_n\} \to \mathrm{T}(\hat{\Sigma})$ be the substitution with $\sigma''(x_i) = p_i$ for $i = 1, \ldots, n$. For each negative premise $u \xrightarrow{\gamma}\!\!\!\!\!/\;$ in H we have $P \vdash_{ws} \sigma'(u) \xrightarrow{\gamma}\!\!\!\!\!/\;$, and thus, by induction, $\mathcal{G}(P) \vdash_{ws} \sigma''(u) \xrightarrow{\gamma}\!\!\!\!\!/\;$. Likewise, for each positive premise $\mu = (u \xrightarrow{\gamma} y_\mu)$ in H, $P \vdash_{ws} \sigma'(u) \xrightarrow{\gamma} \sigma'(y_\mu)$, and thus, by induction, $\mathcal{G}(P) \vdash_{ws} \sigma''(u) \xrightarrow{\gamma} r_\mu$ for some r_μ with $\check{r}_\mu = \sigma'(y_\mu)$. Let $\sigma : V \rightharpoonup \mathrm{T}(\hat{\Sigma})$ be a substitution with $\sigma(x_i) = p_i$ for $i = 1, \ldots, n$ and $\sigma(y_\mu) = r_\mu$ for each positive premise μ in H. Then $\mathcal{G}(P) \vdash_{ws} \sigma(\mu)$ for each premise μ in H, and thus $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\alpha} \sigma(t)$. Take $q := \sigma(t)$. Then $\check{q} = \sigma'(t) = q^*$.

Next consider the case $P \vdash_{ws} \check{p} \xrightarrow{\alpha}\!\!\!\!\!/\;$. Suppose that $\frac{N}{p \xrightarrow{\alpha}}$ is irredundantly provable from $\mathcal{G}(P)$. Then, by Lemma 44, $P \vdash_{irr} \frac{\check{N} \cup N'}{\check{p} \xrightarrow{\alpha}}$ for a set N' of closed negative literals λ with $P \vdash_{ws} \lambda$. By Definition 11 $\check{N} \cup N'$ contains a literal $r' \xrightarrow{\gamma}\!\!\!\!\!/\;$ such that $r' \xrightarrow{\gamma}$ is *ws*-provable from P by means of a strict subproof of π. By the consistency of $\vdash_{ws}$, $(r' \xrightarrow{\gamma}\!\!\!\!\!/\;) \notin N'$. Hence N contains a literal $r \xrightarrow{\gamma}\!\!\!\!\!/\;$ with $\check{r} = r'$. By induction $\mathcal{G}(P) \vdash_{ws} r \xrightarrow{\gamma}$, and this literal denies a literal in N. From this it follows that $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\alpha}\!\!\!\!\!/\;$.

"$\Leftarrow$": We prove both statements by simultaneous induction on the well-founded proof π of $p \xrightarrow{\alpha} q$ or $p \xrightarrow{\alpha}\!\!\!\!\!/\;$ from $\mathcal{G}(P)$. First consider the case $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\alpha} q$.

Assume that p has the form $\hat{r}$, so that $\check{p} = r$. Since $\mathcal{G}(P)$ has no rules whose source is a variable, the last step of π must be an application of a rule $\hat{r} \xrightarrow{\alpha} \hat{s}$, introduced in step 4 of the construction of $\mathcal{G}(P)$. It follows that $q = \hat{s}$, so $\check{q} = s$, and $P \vdash_{ws} r \xrightarrow{\alpha} s$.

Alternatively, $p = f(p_1, \ldots, p_n)$ with $f \in \Sigma$, so that $\check{p} = f(\check{p}_1, \ldots, \check{p}_n)$. Then the last step of π must be an application of a decent ntyft rule $\frac{H}{f(x_1, \ldots, x_n) \xrightarrow{\alpha} t}$ and a substitution $\sigma : V \rightharpoonup \mathrm{T}(\hat{\Sigma})$ with $\sigma(x_i) = p_i$ for $i = 1, \ldots, n$ and $\sigma(t) = q$. Let $\sigma' : V \rightharpoonup \mathrm{T}(\Sigma)$ be the substitution with $\sigma'(x) = \check{\sigma}(x)$ for each $x \in V$ such that $\sigma(x)$ is defined, and undefined otherwise. For each premise μ in H we have $\mathcal{G}(P) \vdash_{ws} \sigma(\mu)$, by means of a subproof of π, and thus, by induction, $P \vdash_{ws} \sigma'(\mu)$. It follows that $P \vdash_{ws} \check{p} \xrightarrow{\alpha} \check{q}$.

Finally, consider the case $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\alpha}\!\!\!\!\!/\;$. Suppose that $\frac{N'}{\check{p} \xrightarrow{\alpha}}$ is irredundantly provable from P. Then, by Lemma 44, either $P \nvdash_{ws} \lambda$ for a literal $\lambda \in N'$, or $\mathcal{G}(P) \vdash_{irr} \frac{N}{p \xrightarrow{\alpha}}$ for some N with $\check{N} \subseteq N'$. In the first case, by the completeness of P, $P \vdash_{ws} \mu$ for a literal μ denying λ, and we are done. So assume the second case. By Definition 11 N contains a literal $r \xrightarrow{\gamma}\!\!\!\!\!/\;$ such that $r \xrightarrow{\gamma}$ is *ws*-provable from $\mathcal{G}(P)$ by means of a strict subproof of π. By induction $P \vdash_{ws} \check{r} \xrightarrow{\gamma}$, and this literal denies a literal in N'. From this it follows that $P \vdash_{ws} \check{p} \xrightarrow{\alpha}\!\!\!\!\!/\;$. □

Corollary 46. *Let $P = (\Sigma, A_\tau, R)$ be a complete standard TSS in decent ntyft format. Then also $\mathcal{G}(P)$ is complete.*

Proof. Let $p \in \mathrm{T}(\hat{\Sigma})$ and $\alpha \in A_\tau$. Since P is complete, either $P \vdash_{ws} \check{p} \xrightarrow{\alpha}$ or $P \vdash_{ws} \check{p} \xrightarrow{\alpha}\!\!\!\!\!/\;$. Consequently, by Proposition 45, either $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\alpha}$ or $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\alpha}\!\!\!\!\!/\;$. □

5.2.3. Comparison of $\mathcal{G}(P)$ and $\mathcal{AFO}(P)$

$\mathcal{G}(P)$ and $\mathcal{AFO}(P)$ differ on the account of rules with premises and conclusions labelled ι and $\omega \in O$. We note that actions $\omega \in O$ play no role in the next lemma and proposition.

Lemma 47. *Let $P = (\Sigma, A_\tau, R)$ be a standard TSS in ntyft format. Let a range over A and p, q over $T(\hat{\Sigma})$.*

- *$\mathcal{AFO}(P) \vdash_{irr} \frac{N}{p \xrightarrow{a} q}$ for a set of closed negative literals N iff there is a set of closed negative literals N' such that $\mathcal{G}(P) \vdash_{irr} \frac{N'}{p \xrightarrow{a} q}$ and $N = \{s \xrightarrow{a}\!\!\!\!\!/\; \mid (s \xrightarrow{a}\!\!\!\!\!/\;) \in N'\} \cup \{t \xrightarrow{\tau}\!\!\!\!\!/\;,\ \ t \xrightarrow{\iota}\!\!\!\!\!/\; \mid (t \xrightarrow{\tau}\!\!\!\!\!/\;) \in N'\}$.*
- *$\mathcal{AFO}(P) \vdash_{irr} \frac{N}{p \xrightarrow{\alpha} q}$, with $\alpha = \tau$ or $\alpha = \iota$, for a set of closed negative literals N, iff there is an N' such that $\mathcal{G}(P) \vdash_{irr} \frac{N'}{p \xrightarrow{\tau} q}$ and $N = \{s \xrightarrow{a}\!\!\!\!\!/\; \mid (s \xrightarrow{a}\!\!\!\!\!/\;) \in N'\} \cup \{t \xrightarrow{\tau}\!\!\!\!\!/\;,\ \ t \xrightarrow{\iota}\!\!\!\!\!/\; \mid (t \xrightarrow{\tau}\!\!\!\!\!/\;) \in N'\}$.*

Proof. "If": Both statements follow by a straightforward simultaneous induction on irredundant provability from $\mathcal{G}(P)$.

"Only if": Both statements follow by a straightforward simultaneous induction on irredundant provability from $\mathcal{AFO}(P)$. □

Proposition 48. *Let $P = (\Sigma, A_\tau, R)$ be a standard TSS in ntyft format. Let a range over A and p, q over $T(\hat{\Sigma})$.*

- *$\mathcal{G}(P) \vdash_{ws} p \xrightarrow{a} q \;\Leftrightarrow\; \mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a} q$.*
- *$\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\tau} q \;\Leftrightarrow\; (\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{\tau} q \vee\ \mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{\iota} q)$.*
- *$\mathcal{G}(P) \vdash_{ws} p \xrightarrow{a}\!\!\!\!\!/\; \;\Leftrightarrow\; \mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a}\!\!\!\!\!/\;$.*
- *$\mathcal{G}(P) \vdash_{ws} p \xrightarrow{\tau}\!\!\!\!\!/\; \Leftrightarrow\; (\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{\tau}\!\!\!\!\!/\; \wedge\ \mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{\iota}\!\!\!\!\!/\;)$.*

Proof. Using Lemma 47, the lemma follows by two straightforward inductions on well-supported provability, one for $\Rightarrow$, covering all four claims, and one for $\Leftarrow$. □

Corollary 49. *Let P be a standard TSS in decent ntyft format. If P is complete, then so is $\mathcal{AFO}(P)$.*

Proof. By Corollary 46, $\mathcal{G}(P)$ is complete. Hence, for each $p \in \mathrm{T}(\hat{\Sigma})$ and $a \in A$, either $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{a}\!\!\!\!\!/$ or $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{a}$. So by Proposition 48, either $\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a}\!\!\!\!\!/$ or $\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a}$.

Each $p \in \mathrm{T}(\hat{\Sigma})$ can be written as $\rho(t)$ with $t \in \mathbb{T}(\Sigma)$ and $\rho : var(t) \to \mathbb{P}_{\mathrm{H}}$. Let Γ be the largest predicate for which P is Γ-patient. Now $\mathcal{AFO}_\Gamma(P) \vdash_{ws} \rho(t) \xrightarrow{\tau}$ if t has a Γ-liquid argument x for which $\rho(x) \xrightarrow{\tau}_{\mathrm{H}}$. Otherwise, $\mathcal{AFO}_\Gamma(P) \vdash_{ws} \rho(t) \xrightarrow{\tau}\!\!\!\!\!/$.

Likewise, for any $\omega \in O$, $\mathcal{AFO}_\Gamma(P) \vdash_{ws} \rho(t) \xrightarrow{\omega}$ if t has a Γ-liquid argument x for which $\rho(x) \xrightarrow{\omega}_{\mathrm{H}}$. Otherwise, $\mathcal{AFO}_\Gamma(P) \vdash_{ws} \rho(t) \xrightarrow{\omega}\!\!\!\!\!/$.

We say that a closed negative literal $p \xrightarrow{\alpha}\!\!\!\!\!/$ is *true* if $\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a}\!\!\!\!\!/$, *false* if $\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a}$, and *ambiguous* if neither applies. Above we proved that there are no ambiguous literals labelled $a \in A$ or τ or $\omega \in O$. It remains to show that there are no ambiguous literals labelled ι. Towards a contradiction, assume that $p \xrightarrow{\iota}\!\!\!\!\!/$ is ambiguous.

There must exist a closed term q and set of closed negative literals N such that $\mathcal{AFO}(P) \vdash_{irr} \frac{N}{p \xrightarrow{\iota} q}$ and no literal in N is false. For if there were no such q and N, the literal $p \xrightarrow{\iota}\!\!\!\!\!/$ would be true by Definition 11. Moreover, N must contain an ambiguous literal $r \xrightarrow{\iota}\!\!\!\!\!/$, for if all literals in N would be true, then the literal $p \xrightarrow{\iota}\!\!\!\!\!/$ would be false by Definition 11. By Lemma 47 there is a set of closed negative literals N' such that $\mathcal{G}(P) \vdash_{irr} \frac{N'}{p \xrightarrow{\tau} q}$ and $N = \{s \xrightarrow{a}\!\!\!\!\!/ \mid (s \xrightarrow{a}\!\!\!\!\!/) \in N'\} \cup \{t \xrightarrow{\tau}\!\!\!\!\!/,\ \ t \xrightarrow{\iota}\!\!\!\!\!/ \mid (t \xrightarrow{\tau}\!\!\!\!\!/) \in N'\}$. So N contains both $r \xrightarrow{\iota}\!\!\!\!\!/$ and $r \xrightarrow{\tau}\!\!\!\!\!/$. If $\mathcal{G}(P) \vdash_{ws} r \xrightarrow{\tau}\!\!\!\!\!/$ then $r \xrightarrow{\iota}\!\!\!\!\!/$ would be true by Proposition 48. Hence $\mathcal{G}(P) \vdash_{ws} r \xrightarrow{\tau}$, by the completeness of $\mathcal{G}(P)$. So by Proposition 48, $\mathcal{AFO}(P) \vdash_{ws} r \xrightarrow{\tau}$ or $\mathcal{AFO}(P) \vdash_{ws} r \xrightarrow{\iota}$. It follows that one of the literals $r \xrightarrow{\iota}\!\!\!\!\!/$ or $r \xrightarrow{\tau}\!\!\!\!\!/$ must be false, contradicting the absence of false literals in N. □

5.2.4. Comparison of $\mathcal{AFO}(P)$ and K

The LTS $\mathrm{K} = (\mathbb{P}_{\mathrm{K}}, A_\tau \cup O \cup \{\iota\}, \to_{\mathrm{K}})$ has as states $\mathbb{P}_{\mathrm{K}} = \{dec(p) \mid p \in \mathrm{T}(\hat{\Sigma})\}$, and the transitions are the ones generated by the following two rules:

$$\frac{x \xrightarrow{\alpha} y}{dec(x) \xrightarrow{\alpha} dec(y)} \qquad \frac{x \xrightarrow{\iota} y}{dec(x) \xrightarrow{\tau} dec(y)}$$

where α ranges over A_τ. The operator $dec : \mathrm{T}(\hat{\Sigma}) \to \mathbb{P}_{\mathrm{K}}$ simply sends any process $p \in \mathrm{T}(\hat{\Sigma})$ to the state $dec(p) \in \mathbb{P}_{\mathrm{K}}$. Thus, dec erases all transitions with labels from O and renames labels ι into τ. All other transitions are preserved.

Proposition 50. *Let $P = (\Sigma, A_\tau, R)$ be a standard TSS. Let a range over A and p, q over $T(\hat{\Sigma})$.*

- $dec(p) \xrightarrow{a}_{\mathrm{K}} q^* \ \Leftrightarrow\ \exists q \in T(\hat{\Sigma}).\, q^* = dec(q) \wedge \mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a} q$.
- $dec(p) \xrightarrow{\tau}_{\mathrm{K}} q^* \ \Leftrightarrow\ \exists q \in T(\hat{\Sigma}).\, q^* = dec(q) \wedge \begin{pmatrix} \mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{\tau} q \\ \vee\ \mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{\iota} q) \end{pmatrix}$.

Proof. Straightforward. □

5.2.5. Verifying requirements 1 and 6 of Theorem 36

That requirement 1 of Theorem 36 is met is immediate by Corollary 49.

We end this section by verifying requirement 6 of Theorem 36. Intuitively, the behaviour of a process $f(p_1, \ldots, p_n)$ in P is the same as that of the process $f(\hat{p}_1, \ldots, \hat{p}_n)$ in $\mathcal{AFO}(P)$, except that some τ-transitions of the former are turned into ι-transitions of the latter process, and that some oracle transitions may have been added in the latter. Since any rule in $\mathcal{AFO}(P)$ with a conclusion labelled by $A_\tau \cup \{\iota\}$ has positive and negative premises with labels from $A_\tau \cup \{\iota\}$ only, these oracle transitions have no influence on the derivation of any transitions from $\mathcal{AFO}(P)$ with labels in $A_\tau \cup \{\iota\}$. The operator dec removes all oracle transitions and renames ι into τ, thereby returning the behaviour of $f(\hat{p}_1, \ldots, \hat{p}_n)$ to match that of $f(p_1, \ldots, p_n)$ exactly.

Proposition 51. *Let P be a complete standard TSS in decent ntyft format. Then $f(p_1, \ldots, p_n) \leftrightarrows_{P \uplus \mathrm{K}} dec(f(\hat{p}_1, \ldots, \hat{p}_n))$ for any n-ary $f \in \Sigma$ and $p_1, \ldots, p_n \in T(\Sigma)$.*

Proof. It suffices to show that the relation $\{(\check{p}, dec(p) \mid p \in \mathrm{T}(\hat{\Sigma})\}$ is a strong bisimulation.

So let $p \in \mathrm{T}(\hat{\Sigma})$. Suppose that $dec(p) \xrightarrow{a}_{\mathrm{K}} q^*$, with $a \in A$. Then, by Proposition 50, there is a $q \in \mathrm{T}(\hat{\Sigma})$ with $q^* = dec(q)$ and $\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a} q$. Hence, by Proposition 48, $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{a} q$. So, by Proposition 45, $P \vdash_{ws} \check{p} \xrightarrow{a} \check{q}$, which had to be shown.

The case that $dec(p) \xrightarrow{\tau}_{\mathrm{K}} q^*$ proceeds in the same way.

Now suppose that $P \vdash_{ws} \check{p} \xrightarrow{a} q^*$, with $a \in A$. Then, by Proposition 45, there is a $q \in \mathrm{T}(\hat{\Sigma})$ with $q^* = \check{q}$ and $\mathcal{G}(P) \vdash_{ws} p \xrightarrow{a} q$. Hence, by Proposition 48, $\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{a} q$. So, by Proposition 50, $dec(p) \xrightarrow{a}_{\mathrm{K}} dec(q)$, which had to be shown. Again the case $P \vdash_{ws} \check{p} \xrightarrow{\tau} q^*$ proceeds in the same way. □

5.3. Application of the general framework to divergence-preserving semantics

We now apply Theorem 36 to derive that the stability-respecting branching bisimulation format and its rooted variant are congruence formats for $\leftrightarrows_b^{\Delta\top}$ and $\leftrightarrows_b^{\Delta}$, and $\leftrightarrows_{rb}^{\Delta\top}$ and $\leftrightarrows_{rb}^{\Delta}$, respectively.

As congruence format $\mathfrak{F}$ in Theorem 36, take the (rooted) stability-respecting branching bisimulation format intersected with the decent ntyft format. If a TSS P is in this format, there exist predicates $\aleph$ and Λ such that P is $\aleph\cap\Lambda$-patient and only contains rules that are rooted stability-respecting branching bisimulation safe w.r.t. $\aleph$ and Λ. Although there may be some freedom in the choice of $\aleph$ and Λ, the predicate $\aleph\cap\Lambda$ is completely determined by P. Namely, each $\aleph\cap\Lambda$-liquid argument of an operator symbol f must have a patience rule, and conversely, each argument of f that has a patience rule must be $\aleph\cap\Lambda$-liquid, by conditions 1 and 3 of Definition 20. Hence there can be no ambiguity in the choice of Γ in $\mathcal{AFO}_\Gamma(P)$.

It is straightforward to check that the conversion $\mathcal{AFO}$ on standard TSSs defined in Sect. 5.2 preserves the (rooted) stability-respecting branching bisimulation format, and thus satisfies requirement 2 of Theorem 36. Here it is important that in step 6 of Definition 37 a term $f(x_1,\ldots,x_n)$ inherits oracle transitions only from its Γ-liquid arguments—else condition 3 of Definition 20 would be violated. Furthermore, condition 4a of Definition 20 is preserved because premises $v \not\xrightarrow{\tau}$ are kept in place in step 3 of Definition 37; and condition 4b of Definition 20 is preserved because the transformation in Definition 37 does not introduce new positive premises with the label τ.

5.3.1. A congruence format for weakly divergence-preserving branching bisimilarity

We first apply Theorem 36 with $\leftrightarrows_b^{\Delta\top}$ and $\leftrightarrows_b^{s}$ in the roles of $\sim$ and $\approx$. In the construction of the LTS H out of the LTS G (cf. Sect. 5.2) we take $O = \{\Delta\top\}$ and let $\zeta(p) = \Delta\top$ iff p is divergent. As observed in Ex. 39, $\hat{p} \sim_{\mathrm{H}} \hat{q}$ iff $\hat{p} \sim_{\mathrm{G}} \hat{q}$. Hence, with Corollary 42, requirement 3 of Theorem 36 is satisfied. We proceed to show that also requirement 4 is satisfied.

Lemma 52. *Let $t \in \mathbb{T}(\Sigma)$ and $\rho : var(t) \to \mathbb{P}_{\mathrm{H}}$. In $\mathcal{AFO}_\Gamma(P)$, $\rho(t)$ is divergent iff t has a Γ-liquid occurrence of a variable x such that $\rho(x)$ is divergent.*

Proof. For "only if", suppose $\rho(t)$ is divergent, i.e., there is an infinite sequence of τ-transitions from $\rho(t)$. Since $\mathcal{AFO}_\Gamma(P)$ is abstraction-free, each of these transitions must originate from a τ-transition from a process $\rho(x)$, where x occurs Γ-liquid in t. One of the variables x that occurs Γ-liquid in t must contribute infinitely many of these transitions, so that $\rho(x)$ is divergent.

"If" follows immediately from the fact that the TSS $\mathcal{AFO}_\Gamma(P)$ is Γ-patient. □

Lemma 53. *Let $p \in T(\hat{\Sigma})$. Then $\mathcal{AFO}_\Gamma(P) \vdash_{ws} p \xrightarrow{\Delta\top}$ iff p is divergent.*

Proof. In $\mathcal{AFO}_\Gamma(P)$ any term $p \in \mathrm{T}(\hat{\Sigma})$ can be written as $\rho(t)$ with $t \in \mathbb{T}(\Sigma)$ and $\rho : var(t) \to \mathbb{P}_{\mathrm{H}}$.

Suppose $\rho(t)$ is divergent. Then t has a Γ-liquid occurrence of a variable x such that $\rho(x)$ is divergent, by Lemma 52. Hence $\rho(x) \xrightarrow{\Delta\top}_{\mathrm{H}}$ by the construction of H. Thus $\mathcal{AFO}_\Gamma(P) \vdash_{ws} \rho(t) \xrightarrow{\Delta\top}$, by Obs. 43. The other direction proceeds likewise. □

The following proposition states that requirement 4 of Theorem 36 is satisfied indeed.

Proposition 54. *On $\mathcal{AFO}(P)$ the equivalences $\leftrightarrows_b^{\Delta\top}$ and $\leftrightarrows_b^{s}$ coincide.*

Proof. It suffices to show that the relation $\leftrightarrows_b^{s}$ on $\mathcal{AFO}(P)$ is a weakly divergence-preserving branching bisimulation. By definition it is a (stability-respecting) branching bisimulation. Hence it remains to show that it is weakly divergence-preserving. To this end, it suffices to show that if p is divergent and $p \leftrightarrows_b^{s} q$ then also q is divergent.

Suppose p is divergent. Then by Lemma 53 $\mathcal{AFO}(P) \vdash_{ws} p \xrightarrow{\Delta\top}$. So $\mathcal{AFO}(P) \vdash_{ws} q \xRightarrow{\varepsilon} q' \xrightarrow{\Delta\top}$ for some q', by the definition of a branching bisimulation. So q' is divergent, by Lemma 53, and thus q is divergent. □

The following example shows that Proposition 54 would not hold if we had skipped step 6 of Definition 37, inheriting oracle transitions for Γ-liquid arguments, or if we had not used the oracle transitions at all.

Example 55. Let $p \in \mathrm{T}(\Sigma)$ be deadlock (a process without outgoing transitions) and $q \in \mathrm{T}(\Sigma)$ a process having only a τ-transition to itself, and a τ-transition to p. Then in the LTS G we have $\hat{p} \leftrightarrows_{b\,\mathrm{G}}^{s} \hat{q}$ but $\hat{p} \not\leftrightarrows_{b\,\mathrm{G}}^{\Delta\top} \hat{q}$. After translation to H,

the processes $\hat{p}$ and $\hat{q}$ are distinguished by means of oracle transitions, so that we have $\hat{p} \not\leftrightarrows^{s}_{b\ \mathrm{H}} \hat{q}$ and $\hat{p} \not\leftrightarrows^{\Delta\top}_{b\ \mathrm{H}} \hat{q}$. Now let Σ feature a unary operator f with as only rule $\frac{x \xrightarrow{\tau} y}{f(x) \xrightarrow{\tau} f(y)}$. If oracle transitions would not be inherited in $\mathcal{AFO}(P)$, then we would have $f(\hat{p}) \leftrightarrows^{s}_{b\,\mathcal{AFO}(P)} f(\hat{q})$ but $f(\hat{p}) \not\leftrightarrows^{\Delta\top}_{b\,\mathcal{AFO}(P)} f(\hat{q})$.

We now verify requirement 5 of Theorem 36.

Proposition 56. $p \leftrightarrows^{\Delta\top}_{b\mathcal{AFO}(P)} q \;\Rightarrow\; dec(p) \leftrightarrows^{\Delta\top}_{b\mathrm{K}} dec(q)$.

Proof. Define the relation $\mathcal{B}$ on the states of K by

$$dec(p)\ \mathcal{B}\ dec(q) \;\Leftrightarrow\; p \leftrightarrows^{\Delta\top}_{b} q\ .$$

It suffices to show that $\mathcal{B}$ is a weakly divergence-preserving branching bisimulation.

- Suppose $p \leftrightarrows^{\Delta\top}_{b} q$ and $dec(p) \xrightarrow{\alpha} p^\dagger$. By the semantics of *dec* there are two possibilities.
  - Case 1: $p \xrightarrow{\alpha} p'$ for some p' with $p^\dagger = dec(p')$. Since $p \leftrightarrows^{\Delta\top}_{b} q$, either $\alpha = \tau$ and $p' \leftrightarrows^{\Delta\top}_{b} q$, or $q \stackrel{\varepsilon}{\Longrightarrow} q' \xrightarrow{\alpha} q''$ for some q' and q'' with $p \leftrightarrows^{\Delta\top}_{b} q'$ and $p' \leftrightarrows^{\Delta\top}_{b} q''$. So either $\alpha = \tau$ and $dec(p')\ \mathcal{B}\ dec(q)$, or $dec(q) \stackrel{\varepsilon}{\Longrightarrow} dec(q') \xrightarrow{\alpha} dec(q'')$ with $dec(p)\ \mathcal{B}\ dec(q')$ and $dec(p')\ \mathcal{B}\ dec(q'')$.
  - Case 2: $\alpha = \tau$, and $p \xrightarrow{\iota} p'$ for some p' with $p^\dagger = dec(p')$. Since $p \leftrightarrows^{\Delta\top}_{b} q$, $q \stackrel{\varepsilon}{\Longrightarrow} q' \xrightarrow{\iota} q''$ for some q' and q'' with $p \leftrightarrows^{\Delta\top}_{b} q'$ and $p' \leftrightarrows^{\Delta\top}_{b} q''$. So $dec(q) \stackrel{\varepsilon}{\Longrightarrow} dec(q') \xrightarrow{\tau} dec(q'')$ with $dec(p)\ \mathcal{B}\ dec(q')$ and $dec(p')\ \mathcal{B}\ dec(q'')$.
- Suppose $p \leftrightarrows^{\Delta\top}_{b} q$ and there is an infinite sequence $(p^\dagger_k)_{k\in\mathbb{N}}$ such that $dec(p) = p^\dagger_0$, $p^\dagger_k \xrightarrow{\tau} p^\dagger_{k+1}$ and $p^\dagger_k\ \mathcal{B}\ dec(q)$ for all $k \in \mathbb{N}$. Then there is an infinite sequence $(p_k)_{k\in\mathbb{N}}$ such that $p_0 = p$ and, for all $k \in \mathbb{N}$, $p^\dagger_k = dec(p_k)$, $p_k \leftrightarrows^{\Delta\top}_{b} q$ and either $p_k \xrightarrow{\tau} p_{k+1}$ or $p_k \xrightarrow{\iota} p_{k+1}$. We distinguish two cases.
  - Case 1: For infinitely many of the k we have $p_k \xrightarrow{\iota} p_{k+1}$. Then there is an infinite sequence $(p'_j)_{j\in\mathbb{N}}$ such that $p'_0 = p_0$ and $p'_j \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\iota} p'_{j+1}$ for all $j \in \mathbb{N}$. Since $p \leftrightarrows_b q$, there must be an infinite sequence $(q'_j)_{j\in\mathbb{N}}$ such that $q = q'_0$, $q'_j \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\iota} q'_{j+1}$ and $p'_j \leftrightarrows_b q'_j$ for all $j \in \mathbb{N}$. It follows that $dec(q) = dec(q'_0)$ and $dec(p'_j) \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} dec(p'_{j+1})$ for all $j \in \mathbb{N}$. In other words, there exists an infinite sequence $(q^\dagger_\ell)_{\ell\in\mathbb{N}}$ such that $dec(q) = q^\dagger_0$ and $q^\dagger_\ell \xrightarrow{\tau} q^\dagger_{\ell+1}$ for all $\ell \in \mathbb{N}$.
  - Case 2: There is an $n \in \mathbb{N}$ such that $p_k \xrightarrow{\tau} p_{k+1}$ for all $k \geq n$. Since $p \leftrightarrows^{\Delta\top}_{b} q$, there must be a finite sequence $(q_k)_{k=0}^{n}$ such that $q = q_0$ and, for all $0 \leq k < n$, either $q_k \stackrel{\varepsilon}{\Longrightarrow} q_{k+1}$ or $q_k \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\iota} q_{k+1}$, and $p_{k+1} \leftrightarrows^{\Delta\top}_{b} q_{k+1}$. Moreover, since $p_n \leftrightarrows^{\Delta\top}_{b} q_n$, and $p_k \leftrightarrows^{\Delta\top}_{b} q \leftrightarrows^{\Delta\top}_{b} p_n \leftrightarrows^{\Delta\top}_{b} q_n$ for each $k \geq n$, there must be an infinite sequence $(q_\ell)_{\ell>n}$ such that $q_\ell \xrightarrow{\tau} q_{\ell+1}$ for all $\ell \geq n$. It follows that $dec(q) = dec(q_0) \stackrel{\varepsilon}{\Longrightarrow} dec(q_n)$ and $dec(q_\ell) \xrightarrow{\tau} dec(q_{\ell+1})$ for all $\ell \geq n$. □

Corollary 57. *$\mathfrak{F}$ is a congruence format for $\leftrightarrows^{\Delta\top}_{b}$.*

As indicated in Sect. 3.1, each standard TSS P in ready simulation format can be converted to a TSS P' in decent ntyft format. In [6] it is shown that this transformation preserves the set of *ws*-provable closed literals, hence $\sim$ is a congruence for P iff P' is. It is not hard to check that if P is in (rooted) stability-respecting branching bisimulation format then so is P'. Thus we obtain the following congruence theorem.

Theorem 58. *If P is a complete standard TSS in stability-respecting branching bisimulation format, then $\leftrightarrows^{\Delta\top}_{b}$ is a congruence for P.* □

5.3.2. A congruence format for rooted weakly divergence-preserving branching bisimilarity

We now apply Theorem 36 with $\leftrightarrows^{\Delta\top}_{rb}$ and $\leftrightarrows^{s}_{rb}$ in the roles of $\sim$ and $\approx$. The choice of O and ζ in the construction of H is the same as above.

Again requirement 3 of Theorem 36 is satisfied: since any two processes $g, h \in \mathbb{P}_{\mathrm{G}}$ are $\leftrightarrows^{\Delta\top}_{b}$-equivalent in G iff they are $\leftrightarrows^{\Delta\top}_{b}$-equivalent in H, it follows immediately from Definition 2 that any two processes $g, h \in \mathbb{P}_{\mathrm{G}}$ are $\leftrightarrows^{\Delta\top}_{rb}$-equivalent in G iff they are $\leftrightarrows^{\Delta\top}_{rb}$-equivalent in H.

Requirement 4 of Theorem 36 is also satisfied: since on $\mathcal{AFO}(P)$ the equivalences $\leftrightarrows^{\Delta\top}_{b}$ and $\leftrightarrows^{s}_{b}$ coincide, it follows immediately from Definition 2 that also $\leftrightarrows^{\Delta\top}_{rb}$ and $\leftrightarrows^{s}_{rb}$ coincide.

To check requirement 5 of Theorem 36 define the relation $\mathcal{R}$ on the states of K by

$$dec(p)\ \mathcal{R}\ dec(q) \;\Leftrightarrow\; p \leftrightarrows^{\Delta\top}_{rb} q\ .$$

With Definition 2 it is not hard to check that $\mathcal{R}$ is a rooted weakly divergence-preserving branching bisimulation. Hence requirement 5 is satisfied.

Corollary 59. *Rooted $\mathfrak{F}$ is a congruence format for $\leftrightarroweq_{rb}^{\Delta\top}$.*

Exactly as above, this yields the following congruence theorem.

Theorem 60. *Let P be a complete standard TSS in rooted stability-respecting branching bisimulation format. Then $\leftrightarroweq_{rb}^{\Delta\top}$ is a congruence for P.* □

5.3.3. A congruence format for divergence-preserving branching bisimilarity

Next, we apply Theorem 36 with $\leftrightarroweq_b^{\Delta}$ and $\leftrightarroweq_b^{s}$ in the roles of $\sim$ and $\approx$. In the construction of the LTS H, we let O contain a unique name for each $\leftrightarroweq_b^{\Delta}$-equivalence class of processes in G, and let $\zeta(p)$ be the name of the $\leftrightarroweq_b^{\Delta}$-equivalence class of $\hat{p} \in \mathbb{P}_G$, for any $p \in T(\Sigma)$. Thus in H all states of G have a fresh outgoing transition, labelled with the name of its $\sim$-equivalence class in G.

With this definition of H, requirement 3 of Theorem 36 is satisfied: any divergence-preserving branching bisimulation $\mathcal{B}$ on G relates states in the same $\leftrightarroweq_b^{\Delta}$-equivalence class only, and thus is also a divergence-preserving branching bisimulation on H (when adding $\surd\,\mathcal{B}\,\surd$).

We proceed to show that also requirement 4 is satisfied. A few lemmas are needed.

Lemma 61. *[18] Condition (D) in Definition 1 can be replaced by the following equivalent condition:*

(D′) *if $p\ \mathcal{B}\ q$ and there is an infinite sequence of processes $(p_k)_{k\in\mathbb{N}}$ such that $p = p_0$, $p_k \xrightarrow{\tau} p_{k+1}$ and $p_k\ \mathcal{B}\ q$ for all $k \in \mathbb{N}$, then there is a process q' such that $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} q'$ and $p_k\ \mathcal{B}\ q'$ for some $k \in \mathbb{N}$.*

That is, the resulting definition also yields the relation $\leftrightarroweq_b^{\Delta}$.

Furthermore, we will employ the following property of $\leftrightarroweq_b^{\Delta}$.

Lemma 62. *[18] If $p_0 \xrightarrow{\tau} p_1 \xrightarrow{\tau} \cdots \xrightarrow{\tau} p_n$ and $p_0 \leftrightarroweq_b^{\Delta} p_n$, then $p_0 \leftrightarroweq_b^{\Delta} p_i$ for all $i = 0, \ldots, n$.*

The following proposition tells that requirement 4 of Theorem 36 is satisfied indeed.

Proposition 63. *On $\mathcal{AFO}(P)$ the equivalences $\leftrightarroweq_b^{\Delta}$ and $\leftrightarroweq_b^{s}$ coincide.*

Proof. It suffices to show that the relation $\leftrightarroweq_b^{s}$ on $\mathcal{AFO}(P)$ is a divergence-preserving branching bisimulation. By definition it is a (stability-respecting) branching bisimulation. Hence it remains to show that it is divergence-preserving. By Lemma 61 it suffices to show that it satisfies condition (D′). So assume that $p \leftrightarroweq_b^{s} q$ and there is an infinite sequence of processes $(p_k)_{k\in\mathbb{N}}$ such that $p = p_0$, $p_k \xrightarrow{\tau} p_{k+1}$ and $p_k \leftrightarroweq_b^{s} q$ for all $k \in \mathbb{N}$. We need to find a process q' such that $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} q'$ and $p_k \leftrightarroweq_b^{s} q'$ for some $k \in \mathbb{N}$. Towards a contradiction, assume that there is no q' with $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} q'$ and $p_k \leftrightarroweq_b^{s} q'$ for some $k \in \mathbb{N}$.

Let $p = \rho(t)$ and $q = \rho(u)$ for univariate $t, u \in \mathbb{T}(\Sigma)$ with $var(t) \cap var(u) = \emptyset$ and $\rho{:}var(t) \cup var(u) \to \mathbb{P}_H$. Since $\mathcal{AFO}(P)$ is abstraction-free, all τ-transitions in the infinite sequence $(p_k \xrightarrow{\tau} p_{k+1})_{k\in\mathbb{N}}$ originate, through patience rules, from τ-transitions of the processes $\rho(x)$ for variables x occurring Γ-liquid in t. Let L be the set of variables x that occur Γ-liquid in t. Then $\rho(x) = h_0^x \xrightarrow{\tau}_H h_1^x \xrightarrow{\tau}_H h_2^x \xrightarrow{\tau}_H \cdots$ for each $x \in L$, and for at least one $x \in L$—call it z—this sequence is infinitely long. For each $k \in \mathbb{N}$ and $x \in L$, let $k_x \leq k$ keep track of how many of the τ-transitions in the sequence $p_0 \xrightarrow{\tau} \cdots \xrightarrow{\tau} p_k$ originate from $\rho(x)$. Hence $p_k = \rho_k(t)$ where $\rho_k(x) = h_{k_x}^x$ for each $x \in L$ and $\rho_k(x) = \rho(x)$ for each $x \in var(t) \setminus L$. Since, by assumption, the sequence of τ-transitions originating from $\rho(z)$ is infinite, for each $\ell \in \mathbb{N}$ there exists a $k \in \mathbb{N}$ such that $k_z = \ell$.

Claim: There is a $j \in \mathbb{N}$ such that $h_n^z \leftrightarroweq_b^{\Delta} h_{j_z}^z$ for all $n \geq j_z$.

Proof of claim: Suppose there is no such j. Then, using Lemma 62, the processes h_ℓ^z for $\ell \in \mathbb{N}$ belong to infinitely many $\leftrightarroweq_b^{\Delta}$-equivalence classes. So there are infinitely many actions $\omega \in O$ such that $\exists \ell.\, h_\ell^z \xrightarrow{\omega}_H$. For each of those actions ω, $\mathcal{AFO}(P) \vdash_{ws} p_k \xrightarrow{\omega}$ for some $k \in \mathbb{N}$, by Obs. 43, and hence $\mathcal{AFO}(P) \vdash_{ws} q \stackrel{\varepsilon}{\Longrightarrow} q'' \xrightarrow{\omega}$ for some q'' with $p_k \leftrightarroweq_b^{s} q''$, since $p_k \leftrightarroweq_b^{s} q$. As we have assumed that there is no q' with $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} q'$ and $p_k \leftrightarroweq_b^{s} q'$ for some $k \in \mathbb{N}$, we have $q'' = q$, and thus $\mathcal{AFO}(P) \vdash_{ws} q \xrightarrow{\omega}$. However, according to Obs. 43 there can only be finitely many actions $\omega \in O$ with $\mathcal{AFO}(P) \vdash_{ws} q \xrightarrow{\omega}$, namely one for each variable occurring Γ-liquid in u. □

Let ω be the name of the $\leftrightarrows_b^\Delta$-equivalence class of the process $\rho_j(z) = h^z_{j_z}$. Then $\rho_j(z) \xrightarrow{\omega}_{\mathrm{H}}$, and consequently $\mathcal{AFO}(P) \vdash_{ws} \rho_j(t) \xrightarrow{\omega}$ by Obs. 43. Since $\rho_j(t) \leftrightarrows_b^s q$, it follows that $\mathcal{AFO}(P) \vdash_{ws} q \stackrel{\varepsilon}{\Longrightarrow} q'' \xrightarrow{\omega}$ for some process q'' with $\rho_j(t) \leftrightarrows_b^s q''$. As we assumed that there is no q' with $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} q'$ and $p_k \leftrightarrows_b^s q'$ for some $k \in \mathbb{N}$, we have $q'' = q$, and thus $\mathcal{AFO}(P) \vdash_{ws} q = \rho(u) \xrightarrow{\omega}$. So by Obs. 43 u has a Γ-liquid occurrence of a variable y with $\rho(y) \xrightarrow{\omega}_{\mathrm{H}}$. It follows that in G we have $h^z_{j_z} = \rho_j(z) \leftrightarrows_b^\Delta \rho(y)$. Thus, by Lemma 61, there exists a $g \in \mathbb{P}_{\mathrm{G}}$ with $\rho(y) \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} g$ and $h^z_\ell \leftrightarrows_b^\Delta g$ for some $\ell \geq j_z$. Since the TSS $\mathcal{AFO}(P)$ is Γ-patient, we have $q = \rho(u) \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} \rho'(u)$ for a substitution ρ' with $\rho'(y) = g$ and $\rho'(x) = \rho(x)$ for all variables $x \neq y$. As, using the claim, $\rho(y) \leftrightarrows_b^s \rho_j(z) = h^z_{j_z} \leftrightarrows_b^s h^z_\ell \leftrightarrows_b^s g = \rho'(y)$, and $\leftrightarrows_b^s$ is a congruence on $\mathcal{AFO}(P)$, it follows that $\rho(u) \leftrightarrows_b^s \rho'(u)$. Since $p \leftrightarrows_b^s q = \rho(u) \leftrightarrows_b^s \rho'(u)$, it suffices to take $q' := \rho'(u)$, so that $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} q'$ and $p \leftrightarrows_b^s q'$. This contradicts our assumption that no such q' exists. □

We now verify requirement 5 of Theorem 36.

Proposition 64. $p \leftrightarrows_{b\ \mathcal{AFO}(P)}^{\Delta} q \;\Rightarrow\; dec(p) \leftrightarrows_{b\ \mathrm{K}}^{\Delta} dec(q)$.

Proof. Define the relation $\mathcal{B}$ on the states of K by

$$dec(p)\ \mathcal{B}\ dec(q) \;\Leftrightarrow\; p \leftrightarrows_b^\Delta q\,.$$

By Lemma 61 it suffices to show that $\mathcal{B}$ is a branching bisimulation satisfying condition (D′). That it is a branching bisimulation follows exactly as in the proof of Proposition 56. To show that it satisfies (D′), suppose $p \leftrightarrows_b^\Delta q$ and there is an infinite sequence $(p_k^\dagger)_{k\in\mathbb{N}}$ such that $dec(p) = p_0^\dagger$, $p_k^\dagger \xrightarrow{\tau} p_{k+1}^\dagger$ and $p_k^\dagger\ \mathcal{B}\ dec(q)$ for all $k \in \mathbb{N}$. Then there is an infinite sequence $(p_k)_{k\in\mathbb{N}}$ such that $p_0 = p$ and, for all $k \in \mathbb{N}$, $p_k^\dagger = dec(p_k)$, $p_k \leftrightarrows_b^\Delta q$ and either $p_k \xrightarrow{\tau} p_{k+1}$ or $p_k \xrightarrow{\iota} p_{k+1}$. We distinguish two cases.

Case 1: There is a $k \in \mathbb{N}$ with $p_k \xrightarrow{\iota} p_{k+1}$—consider the first such k. Then $p \stackrel{\varepsilon}{\Longrightarrow} p_k \xrightarrow{\iota} p_{k+1}$. Since $p \leftrightarrows_b^\Delta q$, there is a q' with $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\iota} q'$ and $p_{k+1} \leftrightarrows_b^\Delta q'$. Hence there is a $dec(q')$ with $dec(q) \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} dec(q')$ and $dec(p_{k+1})\ \mathcal{B}\ dec(q')$.

Case 2: There is no such k. Then, since $\leftrightarrows_b^\Delta$ satisfies (D′), there is a q' such that $q \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} q'$ and $p_k \leftrightarrows_b^\Delta q'$ for some $k \in \mathbb{N}$. It follows that there is a $dec(q')$ such that $dec(q) \stackrel{\varepsilon}{\Longrightarrow} \xrightarrow{\tau} dec(q')$ and $dec(p_{k+1})\ \mathcal{B}\ dec(q')$. □

Corollary 65. *$\mathfrak{F}$ is a congruence format for $\leftrightarrows_b^\Delta$.*

Exactly as above, this yields the following congruence theorem.

Theorem 66. *If P is a complete standard TSS in stability-respecting branching bisimulation format, then $\leftrightarrows_b^\Delta$ is a congruence for P.* □

5.3.4. A congruence format for rooted divergence-preserving branching bisimilarity

We now apply Theorem 36 with $\leftrightarrows_{rb}^\Delta$ and $\leftrightarrows_{rb}^s$ in the roles of $\sim$ and $\approx$. The choice of O and ζ in the construction of H is the same as in Sect. 5.3.3.

Again requirement 3 of Theorem 36 is satisfied: since any two processes $g, h \in \mathbb{P}_{\mathrm{G}}$ are $\leftrightarrows_b^\Delta$-equivalent in G iff they are $\leftrightarrows_b^\Delta$-equivalent in H, if follows immediately from Definition 2 that any two processes $g, h \in \mathbb{P}_{\mathrm{G}}$ are $\leftrightarrows_{rb}^\Delta$-equivalent in G iff they are $\leftrightarrows_{rb}^\Delta$-equivalent in H.

Requirement 4 of Theorem 36 is also satisfied: since on $\mathcal{AFO}(P)$ the equivalences $\leftrightarrows_b^\Delta$ and $\leftrightarrows_b^s$ coincide, it follows immediately from Definition 2 that also $\leftrightarrows_{rb}^\Delta$ and $\leftrightarrows_{rb}^s$ coincide.

To check requirement 5 of Theorem 36 define the relation $\mathcal{R}$ on the states of K by

$$dec(p)\ \mathcal{R}\ dec(q) \;\Leftrightarrow\; p \leftrightarrows_{rb}^\Delta q\,.$$

With Definition 2 it is not hard to check that $\mathcal{R}$ is a rooted weakly divergence-preserving branching bisimulation. Hence requirement 5 is satisfied.

Corollary 67. *Rooted $\mathfrak{F}$ is a congruence format for $\leftrightarrows_{rb}^\Delta$.*

Exactly as above, this yields the following congruence theorem.

Theorem 68. *Let P be a complete standard TSS in rooted stability-respecting branching bisimulation format. Then $\leftrightarrows_{rb}^\Delta$ is a congruence for P.* □

6. Related work

Ulidowski [30–32] proposed congruence formats, inside GSOS [7], for weak semantics that take into account non-divergence, called *convergence* in [16]. In [30] he introduces the *ISOS* format, and shows that the weak convergent *refusal simulation* preorder is a precongruence for all TSSs in the ISOS format. The GSOS format—in our terminology the *decent nxyft* format—allows only decent ntyft rules with variables as the left-hand sides of premises. The ISOS format is contained in the intersection of the GSOS format and our stability-preserving branching bisimulation format. Its additional restriction is that no variable may occur multiple times as the left-hand side of a positive premise, or both as the left-hand side of a positive premise and in the conclusion of a rule.

In [31,32] he employs *Ordered SOS* (OSOS) TSSs [26]. An OSOS TSS allows no negative premises, but includes priorities between rules: $r < r'$ means that r can only be applied if r' cannot. An OSOS specification can be seen as, or translated into, a GSOS specification with negative premises. Each rule r with exactly one higher-priority rule $r' > r$ is replaced by a number of rules, one for each (positive) premise of r'; in the copy of r, this premise is negated. For a rule r with multiple higher-priority rules r', this replacement is carried out for each such r'.

The `ebo` and `bbo` formats from [31] target unrooted convergent *delay* and branching bisimulation equivalence, respectively. The `bbo` format is more liberal than the `ebo` format, which in turn is more liberal than the ISOS format. The `rebo` and `rbbo` formats from [32] target rooted convergent delay and branching bisimulation equivalence, respectively. These rooted formats are more liberal than their unrooted counterparts, and the `rbbo` format is more liberal than the `rebo` format.

If patience rules are not allowed to have a lower priority than other rules, then the `(r)bbo` format, upon translation from OSOS to GSOS, can be seen as a subformat of our (rooted) stability-respecting branching bisimulation format. The basic idea is that in the `rbbo` format all arguments of so-called τ-preserving function symbols [32], which are the only ones allowed to occur in targets, are declared Λ-liquid; in the `bbo` format, all arguments of function symbols are declared Λ-liquid. Moreover, all arguments of function symbols that occur as the left-hand side of a positive premise are declared $\aleph$-liquid. Patience rules are in the `(r)bbo` format however, under strict conditions, allowed to be dominated by other rules, which in our setting gives rise to patience rules with negative premises. This is outside the realm of our rooted stability-respecting branching bisimulation format. On the other hand, the TSSs of the process algebra $\mathrm{BPA}_{\varepsilon\delta\tau}$, the binary Kleene star and deadlock testing (see [11,12]), for which rooted convergent branching bisimulation equivalence is a congruence, are outside the `rbbo` format but within the rooted stability-respecting branching bisimulation format.

7. Conclusions

We showed how the method from [15] for deriving congruence formats through modal decomposition can be applied to weak semantics that are stability-respecting. We used (rooted and unrooted) stability-respecting branching bisimulation equivalence as a notable example. Moreover, we developed a general method for lifting congruence formats from a weak semantics to a finer semantics, and used it to show that congruence formats for stability-respecting branching bisimulation equivalence are also congruence formats for their divergence-preserving counterparts. This research provides a deeper insight into the link between modal logic and congruence formats, and strengthens the framework from [15] for the derivation of congruence formats for weak semantics.

Almost every weak semantics has stability-respecting and divergence-preserving variants. Such variants have been studied most widely in the literature for branching, η-, delay and weak bisimulation equivalence, but they have for instance also been considered for decorated trace semantics, such as *exhibited behaviour equivalence* [29], the *generalised failure preorder* [24], *refusal equivalence* [27], and *copy+refusal equivalence* [30]. We expect that the methods developed in this paper, combined with the methods from [15,12], can serve as a cornerstone in the generation of congruence formats for such semantics. In particular, we conjecture that this will straightforwardly yield congruence formats for stability-respecting and divergence-preserving variants of η-, delay and weak bisimulation equivalence.

Admittedly, the whole story is quite technical and intricate. Partly this is because we build on a rich body of earlier work in the realm of structural operational semantics: the notions of well-supported proofs and complete TSSs from [17]; the ntyft/ntyxt format from [21,8]; the transformation to ruloids, which for the main part goes back to [10]; and the work on modal decomposition and congruence formats from [6]. In spite of these technicalities, we have arrived at a relatively simple framework for the derivation of congruence formats for weak semantics. Namely, for this one only needs to: (1) provide a modal characterisation of the weak semantics under consideration; (2) study the class of modal formulas that result from decomposing this modal characterisation, and formulate syntactic restrictions on TSSs to bring this class of modal formulas within the original modal characterisation; and (3) check that these syntactic restrictions are preserved under the transformation to ruloids. Steps (2) and (3) are very similar in structure for different weak semantics, as exemplified by the way we obtained a congruence format for stability-respecting branching bisimulation equivalence. And the resulting congruence formats tend to be more liberal and elegant than existing congruence formats in the literature.

Our intention is to carve out congruence formats for all weak semantics in the spectrum from [16] that have reasonable congruence properties. At first we expected that the current third instalment would allow us to do so. However, it turns out that convergent weak semantics as considered in for instance [31,32,34] still need extra work. The modal characterisations of

these semantics are three-valued [16], which requires an extension of the modal decomposition technique to a three-valued setting.

Declaration of Competing Interest

We wish to confirm that there are no known conflicts of interest associated with this publication and that, apart from the regular academic salaries paid to the authors, there has been no significant financial support for this work that could have influenced its outcome.

We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the (alphabetical) order in which the authors are listed has been approved by all of us.

We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

Appendix A. Modal characterisations

We prove Theorem 8, which states that $\mathbb{O}^s_b$ is a modal characterisation of $p \leftrightarrows^s_b q$, and $\mathbb{O}^s_{rb}$ of $p \leftrightarrows^s_{rb} q$. So we need to prove, given an LTS $(\mathbb{P}, Act, \rightarrow)$, that $p \leftrightarrows^s_b q \Leftrightarrow p \sim_{\mathbb{O}^s_b} q$ and $p \leftrightarrows^s_{rb} q \Leftrightarrow p \sim_{\mathbb{O}^s_{rb}} q$ for all $p, q \in \mathbb{P}$.

Proof. ($\Rightarrow$) We prove by structural induction on φ, resp. $\overline{\varphi}$, that $p \leftrightarrows^s_b q \wedge p \models \varphi \Rightarrow q \models \varphi$ for all $\varphi \in \mathbb{O}^s_b$, and $p \leftrightarrows^s_{rb} q \wedge p \models \overline{\varphi} \Rightarrow q \models \overline{\varphi}$ for all $\overline{\varphi} \in \mathbb{O}^s_{rb}$, The converse implications ($q \models \varphi \Rightarrow p \models \varphi$ and $q \models \overline{\varphi} \Rightarrow p \models \overline{\varphi}$) follow by symmetry.

- $\varphi = \bigwedge_{i\in I} \varphi_i$. Then $p \models \varphi_i$ for all $i \in I$. By induction $q \models \varphi_i$ for all $i \in I$, so $q \models \bigwedge_{i\in I} \varphi_i$.
- $\varphi = \neg\varphi'$. Then $p \not\models \varphi'$. By induction $q \not\models \varphi'$, so $q \models \neg\varphi'$.
- $\varphi = \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$. Then for some n there are $p_0, \ldots, p_n \in \mathbb{P}$ with $p_0 = p$, $p_i \xrightarrow{\tau} p_{i+1}$ for $i \in \{0, \ldots, n-1\}$, and $p_n \models \varphi_1\langle\hat{\tau}\rangle\varphi_2$. We apply induction on n.
  - $\boldsymbol{n = 0}$ Then $p \models \varphi_1$, so by induction on formula size, $q \models \varphi_1$. Furthermore, either (1) $p \models \varphi_2$ or (2) there is a $p' \in \mathbb{P}$ with $p \xrightarrow{\tau} p'$ and $p' \models \varphi_2$. In case (1), by induction on formula size, $q \models \varphi_2$, so $q \models \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$. In case (2), since $p \leftrightarrows^s_b q$, by Definition 1 either (2.1) $p' \leftrightarrows^s_b q$ or (2.2) $q \stackrel{\varepsilon}{\Longrightarrow} q' \xrightarrow{\tau} q''$ with $p \leftrightarrows^s_b q'$ and $p' \leftrightarrows^s_b q''$. In case (2.1), by induction on formula size, $q \models \varphi_2$. In case (2.2), by induction on formula size, $q' \models \varphi_1$ and $q'' \models \varphi_2$. In both cases, $q \models \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$.
  - $\boldsymbol{n > 0}$ Since $p \xrightarrow{\tau} p_1$, and $p \leftrightarrows^s_b q$, according to Definition 1 there are two possibilities.
    (a) Either $p_1 \leftrightarrows^s_b q$. Since $p_1 \models \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$, by induction on n, $q \models \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$.
    (b) Or $q \stackrel{\varepsilon}{\Longrightarrow} q' \xrightarrow{\tau} q''$ with $p_1 \leftrightarrows^s_b q''$. Since $p_1 \models \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$, by induction on n, $q'' \models \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$. Hence $q \models \langle\varepsilon\rangle\varphi_1\langle\hat{\tau}\rangle\varphi_2$.
- $\varphi = \langle\varepsilon\rangle\varphi_1\langle a\rangle\varphi_2$. Then for some n there are $p_0, \ldots, p_n \in \mathbb{P}$ with $p_0 = p$, $p_i \xrightarrow{\tau} p_{i+1}$ for $i \in \{0, \ldots, n-1\}$, and $p_n \models \varphi_1\langle a\rangle\varphi_2$. We apply induction on n.
  - $\boldsymbol{n = 0}$ Then $p \models \varphi_1$, and there is a $p' \in \mathbb{P}$ with $p \xrightarrow{a} p'$ and $p' \models \varphi_2$. Since $p \leftrightarrows^s_b q$, by Definition 1 $q \stackrel{\varepsilon}{\Longrightarrow} q' \xrightarrow{a} q''$ with $p \leftrightarrows^s_b q'$ and $p' \leftrightarrows^s_b q''$. By induction on formula size, $q' \models \varphi_1$ and $q'' \models \varphi_2$. Hence $q \models \langle\varepsilon\rangle\varphi_1\langle a\rangle\varphi_2$.
  - $\boldsymbol{n > 0}$ This case goes exactly as the case $n > 0$ above.
- $\varphi = \langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \overline{\varphi})$ with $\overline{\varphi} \in \mathbb{O}^s_{rb}$. Then for some n there are $p_0, \ldots, p_n \in \mathbb{P}$ with $p_0 = p$, $p_i \xrightarrow{\tau} p_{i+1}$ for $i \in \{0, \ldots, n-1\}$, and $p_n \models \neg\langle\tau\rangle\top \wedge \overline{\varphi}$. We apply induction on n.
  - $\boldsymbol{n = 0}$ Then $p \models \overline{\varphi}$ and $p \xrightarrow{\tau}\!\!\!\!\!/\ $. Since p and q are related by a *stability-respecting* branching bisimulation, there is a q' with $q \stackrel{\varepsilon}{\Longrightarrow} q' \xrightarrow{\tau}\!\!\!\!\!/\ $ and $p \leftrightarrows^s_b q'$. Because p and q' are both stable, $p \leftrightarrows^s_b q'$ implies $p \leftrightarrows^s_{rb} q'$. So by induction on formula size, $q' \models \overline{\varphi}$. Hence $q \models \langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \overline{\varphi})$.
  - $\boldsymbol{n > 0}$ This case goes exactly as the case $n > 0$ above.
- $\overline{\varphi} = \bigwedge_{i\in I} \overline{\varphi}_i$. Then $p \models \overline{\varphi}_i$ for all $i \in I$. By induction $q \models \overline{\varphi}_i$ for all $i \in I$, so $q \models \bigwedge_{i\in I} \overline{\varphi}_i$.
- $\overline{\varphi} = \neg\overline{\varphi}'$. Then $p \not\models \overline{\varphi}'$. By induction $q \not\models \overline{\varphi}'$, so $q \models \neg\overline{\varphi}'$.
- $\overline{\varphi} = \langle\alpha\rangle\varphi$ with $\varphi \in \mathbb{O}^s_b$. Then $p \xrightarrow{\alpha} p'$ for some p' with $p \models \varphi$. By Definition 2, $q \xrightarrow{\alpha} q'$ for some q' with $p' \leftrightarrows^s_b q'$. So by induction $q' \models \varphi$, and hence $q \models \langle\alpha\rangle\varphi$.
- $\overline{\varphi} = \varphi \in \mathbb{O}^s_b$. Since $p \leftrightarrows^s_{rb} q$ implies $p \leftrightarrows^s_b q$, we obtain $q \models \varphi$ by the cases treated above.

($\Leftarrow$) We first prove that $\sim_{\mathbb{O}^s_b}$ is a branching bisimulation. The relation is clearly symmetric. Let $p \sim_{\mathbb{O}^s_b} q$. Suppose $p \xrightarrow{\alpha} p'$. If $\alpha = \tau$ and $p' \sim_{\mathbb{O}^s_b} q$, then the first condition of Definition 1 is fulfilled. So we can assume that either (i) $\alpha \neq \tau$ or (ii) $p' \not\sim_{\mathbb{O}^s_b} q$. We define two sets:

$$Q' = \{q' \in \mathbb{P} \mid q \stackrel{\varepsilon}{\Longrightarrow} q' \wedge p \not\sim_{\mathbb{O}^s_b} q'\}$$
$$Q'' = \{q'' \in \mathbb{P} \mid \exists q' \in \mathbb{P} : q \stackrel{\varepsilon}{\Longrightarrow} q' \xrightarrow{\alpha} q'' \wedge p' \not\sim_{\mathbb{O}^s_b} q''\}$$

For each $q' \in Q'$, let $\varphi_{q'}$ be a formula in $\mathbb{O}_b^s$ such that $p \models \varphi_{q'}$ and $q' \not\models \varphi_{q'}$. (Such a formula always exists because $\mathbb{O}_b^s$ is closed under negation $\neg$.) We define

$$\varphi = \bigwedge_{q' \in Q'} \varphi_{q'}$$

Similarly, for each $q'' \in Q''$, let $\psi_{q''}$ be a formula in $\mathbb{O}_b^s$ such that $p' \models \psi_{q''}$ and $q'' \not\models \psi_{q''}$. We define

$$\psi = \bigwedge_{q'' \in Q''} \psi_{q''}$$

Clearly, $\varphi, \psi \in \mathbb{O}_b^s$, $p \models \varphi$ and $p' \models \psi$. We distinguish two cases.

1. $\alpha \neq \tau$. Since $p \models \langle\varepsilon\rangle\varphi\langle\alpha\rangle\psi \in \mathbb{O}_b^s$ and $p \sim_{\mathbb{O}_b^s} q$, also $q \models \langle\varepsilon\rangle\varphi\langle\alpha\rangle\psi$. Hence $q \stackrel{\varepsilon}{\Longrightarrow} q' \stackrel{\alpha}{\longrightarrow} q''$ with $q' \models \varphi$ and $q'' \models \psi$. By the definition of φ and ψ it follows that $p \sim_{\mathbb{O}_b^s} q'$ and $p' \sim_{\mathbb{O}_b^s} q''$.
2. $\alpha = \tau$ and $p' \not\sim_{\mathbb{O}_b^s} q$. Let $\tilde{\varphi} \in \mathbb{O}_b^s$ such that $p' \models \tilde{\varphi}$ and $p, q \not\models \tilde{\varphi}$. Since $p \models \langle\varepsilon\rangle\varphi\langle\hat{\tau}\rangle(\tilde{\varphi} \wedge \psi) \in \mathbb{O}_b^s$ and $p \sim_{\mathbb{O}_b^s} q$, also $q \models \langle\varepsilon\rangle\varphi\langle\hat{\tau}\rangle(\tilde{\varphi} \wedge \psi)$. So $q \stackrel{\varepsilon}{\Longrightarrow} q'$ with $q' \models \varphi\langle\hat{\tau}\rangle(\tilde{\varphi} \wedge \psi)$. By definition of φ it follows that $p \sim_{\mathbb{O}_b^s} q'$. Thus $q' \not\models \tilde{\varphi}$, so $q' \stackrel{\tau}{\longrightarrow} q''$ with $q'' \models \tilde{\varphi} \wedge \psi$. By the definition of ψ it follows that $p' \sim_{\mathbb{O}_b^s} q''$.

Both cases imply that the first condition of Definition 1 is fulfilled, i.e. that $\sim_{\mathbb{O}_b^s}$ is a branching bisimulation. Next we show that $\sim_{\mathbb{O}_b^s}$ is stability-respecting. Let $p \sim_{\mathbb{O}_b^s} q$ and $p \stackrel{\tau}{\not\longrightarrow}$. Define Q' and $\varphi \in \mathbb{O}_b^s \subseteq \mathbb{O}_{rb}^s$ as above. Then $p \models \langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \varphi)$, and thus also $q \models \langle\varepsilon\rangle(\neg\langle\tau\rangle\top \wedge \varphi)$. Hence $q \stackrel{\varepsilon}{\Longrightarrow} q'$ for some q' with $q' \models \neg\langle\tau\rangle\top \wedge \varphi$. So $q \stackrel{\tau}{\not\longrightarrow}$ and by the definition of φ it follows that $p \sim_{\mathbb{O}_b^s} q'$, which had to be shown.

Finally we show that $\sim_{\mathbb{O}_{rb}^s}$ is a rooted stability-respecting branching bisimulation. Let $p \sim_{\mathbb{O}_{rb}^s} q$ and $p \stackrel{\alpha}{\longrightarrow} p'$. Let $Q_\alpha = \{q' \in \mathbb{P} \mid q \stackrel{\alpha}{\longrightarrow} q' \wedge p' \not\sim_{\mathbb{O}_b^s} q'\}$. For each $q' \in Q'$, let $\chi_{q'}$ be a formula in $\mathbb{O}_b^s$ such that $p' \models \chi_{q'}$ and $q' \not\models \chi_{q'}$. We define $\chi = \bigwedge_{q' \in Q'} \chi_{q'}$. Since $p \models \langle\alpha\rangle\chi$ we have $q \models \langle\alpha\rangle\chi$, so $q \stackrel{\alpha}{\longrightarrow} q'$ for some q' with $q' \models \chi$. By the definition of χ it follows that $p' \sim_{\mathbb{O}_b^s} q'$, which had to be shown. □

References

[1] J.C.M. Baeten, J.A. Bergstra, J.W. Klop, Syntax and defining equations for an interrupt mechanism in process algebra, Fundam. Inform. 9 (2) (1986) 127–167.
[2] J.C.M. Baeten, B. Luttik, F. Yang, Sequential composition in the presence of intermediate termination (extended abstract), in: Proc. EXPRESS/SOS'17, in: EPTCS, vol. 255, 2017, pp. 1–17.
[3] T. Basten, Branching bisimulation is an equivalence indeed!, Inf. Process. Lett. 58 (3) (1996) 141–147.
[4] B. Bloom, When is partial trace equivalence adequate?, Form. Asp. Comput. 6 (1994) 317–338.
[5] B. Bloom, Structural operational semantics for weak bisimulations, Theor. Comput. Sci. 146 (1/2) (1995) 25–68.
[6] B. Bloom, W.J. Fokkink, R.J. van Glabbeek, Precongruence formats for decorated trace semantics, ACM Trans. Comput. Log. 5 (1) (2004) 26–78.
[7] B. Bloom, S. Istrail, A.R. Meyer, Bisimulation can't be traced, J. ACM 42 (1) (1995) 232–268.
[8] R.N. Bol, J.F. Groote, The meaning of negative premises in transition system specifications, J. ACM 43 (5) (1996) 863–914.
[9] W.J. Fokkink, Rooted branching bisimulation as a congruence, J. Comput. Syst. Sci. 60 (1) (2000) 13–37.
[10] W.J. Fokkink, R.J. van Glabbeek, Ntyft/ntyxt rules reduce to ntree rules, Inf. Comput. 126 (1) (1996) 1–10.
[11] W.J. Fokkink, R.J. van Glabbeek, Divide and congruence II: delay and weak bisimilarity, in: Proc. LICS'16, ACM/IEEE, 2016, pp. 778–787.
[12] W.J. Fokkink, R.J. van Glabbeek, Divide and congruence II: from decomposition of modal formulas to preservation of delay and weak bisimilarity, Inf. Comput. 257 (2017) 79–113.
[13] W.J. Fokkink, R.J. van Glabbeek, B. Luttik, Divide and congruence III: stability & divergence, in: Proc. CONCUR'17, LIPIcs 85, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017, pp. 15:1–15:16.
[14] W.J. Fokkink, R.J. van Glabbeek, P. de Wind, Divide and congruence: from decomposition of modalities to preservation of branching bisimulation, in: Proc. FMCO'05, in: LNCS, vol. 4111, Springer, 2006, pp. 195–218.
[15] W.J. Fokkink, R.J. van Glabbeek, P. de Wind, Divide and congruence: from decomposition of modal formulas to preservation of branching and η-bisimilarity, Inf. Comput. 214 (2012) 59–85.
[16] R.J. van Glabbeek, The linear time-branching time spectrum II: the semantics of sequential systems with silent moves, in: Proc. CONCUR'93, in: LNCS, vol. 715, Springer, 1993, pp. 66–81.
[17] R.J. van Glabbeek, The meaning of negative premises in transition system specifications II, J. Log. Algebraic Program. 60/61 (2004) 229–258.
[18] R.J. van Glabbeek, B. Luttik, N. Trčka, Branching bisimilarity with explicit divergence, Fundam. Inform. 93 (4) (2009) 371–392.
[19] R.J. van Glabbeek, B. Luttik, N. Trčka, Computation tree logic with deadlock detection, Log. Methods Comput. Sci. 5 (4) (2009) 5.
[20] R.J. van Glabbeek, W.P. Weijland, Branching time and abstraction in bisimulation semantics, J. ACM 43 (3) (1996) 555–600.
[21] J.F. Groote, Transition system specifications with negative premises, Theor. Comput. Sci. 118 (2) (1993) 263–299.
[22] J.F. Groote, F.W. Vaandrager, Structured operational semantics and bisimulation as a congruence, Inf. Comput. 100 (2) (1992) 202–260.
[23] M. Hennessy, R. Milner, Algebraic laws for non-determinism and concurrency, J. ACM 32 (1) (1985) 137–161.
[24] R. Langerak, A testing theory for LOTOS using deadlock detection, in: E. Brinksma, G. Scollo, C.A. Vissers (Eds.), Proceedings 9th IFIP WG6.1 International Symposium on Protocol Specification, Testing, and Verification, Enschede, The Netherlands, 1989.
[25] R. Milner, Communication and Concurrency, Prentice-Hall, 1989.
[26] M.R. Mousavi, I. Phillips, M.A. Reniers, I. Ulidowski, Semantics and expressiveness of ordered SOS, Inf. Comput. 207 (2009) 85–119.

[27] I. Phillips, Refusal testing, Theor. Comput. Sci. 50 (1987) 241–284.
[28] G.D. Plotkin, A structural approach to operational semantics, J. Log. Algebraic Program. 60/61 (2004) 17–139, Originally appeared in 1981.
[29] L. Pomello, Some equivalence notions for concurrent systems – an overview, in: G. Rozenberg (Ed.), Advances in Petri Nets 1985, in: LNCS, vol. 222, Springer, 1986, pp. 381–400.
[30] I. Ulidowski, Equivalences on observable processes, in: Proc. LICS'92, IEEE, 1992, pp. 148–159.
[31] I. Ulidowski, I. Phillips, Ordered SOS rules and process languages for branching and eager bisimulations, Inf. Comput. 178 (2002) 180–213.
[32] I. Ulidowski, S. Yuen, Process languages for rooted eager bisimulation, in: Proc. CONCUR'00, in: LNCS, vol. 1877, Springer, 2000, pp. 275–289.
[33] F.W. Vaandrager, Algebraic Techniques for Concurrency and Their Application, PhD Thesis, University of Amsterdam, 1990.
[34] D. Walker, Bisimulation and divergence, Inf. Comput. 85 (1990) 202–241.