Precongruence Formats with Lookahead through Modal Decomposition

Wan Fokkink[1] and Rob J. van Glabbeek[2]

1 **Vrije Universiteit Amsterdam, The Netherlands**
2 **Data61, CSIRO, Sydney, Australia; and**
University of New South Wales, Sydney, Australia

Abstract

Bloom, Fokkink & van Glabbeek (2004) presented a method to decompose formulas from Hennessy-Milner logic with regard to a structural operational semantics specification. A term in the corresponding process algebra satisfies a Hennessy-Milner formula if and only if its subterms satisfy certain formulas, obtained by decomposing the original formula. They used this decomposition method to derive congruence formats in the realm of structural operational semantics. In this paper it is shown how this framework can be extended to specifications that include bounded lookahead in their premises. This extension is used in the derivation of a congruence format for the partial trace preorder.

1998 ACM Subject Classification F.3.2 [Semantics of Programming Languages] Operational Semantics, F.3.1 [Specifying and Verifying and Reasoning about Programs] Specification Techniques, F.4.1 [Mathematical Logic] Modal Logic, Proof Theory

Keywords and phrases Structural Operational Semantics, Compositionality, Congruence, Modal Logic, Modal Decomposition, Lookahead

Digital Object Identifier 10.4230/LIPIcs.CSL.2017.25

1 Introduction

Structural operational semantics [24] provides specification languages with an interpretation. A transition system specification (TSS), consisting of an algebraic signature and a set of transition rules of the form $\frac{\text{premises}}{\text{conclusion}}$, generates a labelled transition system consisting of transitions between the closed terms over the signature. Transition rules may contain lookahead, meaning that the right-hand side of a premise may occur in the left-hand side of a premise. Lookahead appears for example in the structural operational semantics of a process algebra for process creation [2], an axiomatisation of the process algebra ACP_τ [17], timed LOTOS [22], the stochastic timed process algebra EMPA [3], a probabilistic bisimulation tester [9], and the synchronous programming language Esterel [23]. It also plays an important role in parsing algorithms for e.g. Java [10]. The usefulness of lookahead in formal semantics in the context of agent systems is advocated in [20].

Hennessy-Milner logic (HML) [19] is a dynamic logic to specify properties of a labelled transition system. Larsen [21] showed how to decompose formulas from HML with respect to a TSS in the De Simone format [25]. In [13] this decomposition method was extended to the tyft/tyxt format [18], which allows lookahead. As a step towards this end they used a transformation from [11] to derive so-called "P-ruloids" from a TSS P: transition rules in which the left-hand sides of premises are single variables. A rule has bounded lookahead if all chains of forward dependencies between its premises are finite. In [13] each ruloid with unbounded lookahead is replaced by an equivalent ruloid with bounded lookahead,

by endowing each infinite forward chain with an ordinal count-down. This step is needed because HML, even with infinite conjunctions, cannot capture unbounded lookahead.

An equivalence is a congruence for a given process algebra if it is preserved by all functions in the signature. This is an important property, notably to fit a semantics into an axiomatic framework. Different syntactic formats have been developed for TSSs, to guarantee this property for specific semantics, i.e. for specific behavioural equivalences or preorders. Most of these congruence formats, notably the De Simone format, GSOS [6] and the ready simulation format [15], disallow lookahead.

In [5] the decomposition method for HML is exploited to derive congruence formats, in the context of structural operational semantics, for a wide range of semantics. It takes the ready simulation format as starting point, so the obtained congruence formats are limited to transition rules that have no lookahead. With regard to their congruence format for partial trace semantics, the open question was posed whether the method can be extended to allow some form of lookahead. Doing this is the aim of the current paper.

The key idea in the decomposition method from [5] is that a congruence format $\mathfrak{F}$ for a semantics must ensure that the formulas in a modal characterisation of this semantics are always decomposed into formulas that are again in this modal characterisation. To obtain such a property one needs that if all rules of a TSS P are in $\mathfrak{F}$-format, then so are are all P-ruloids that are needed in the decomposition methods from [13]. However, the ruloids produced by the transformation from unbounded to bounded lookahead in [13] violate most congruence formats, including the partial trace format from [5]. (This is no surprise, as a partial trace format cannot allow unbounded lookahead; see Example 3.)

Lookahead is intrinsically difficult because it establishes an indirect, transitive relation between variables in (proofs using) transition rules. We introduce the notion of a "general" P-ruloid, which intuitively means that different occurrences of the same variable in the rule are linked to each other through its proof; i.e., after renaming some but not all occurrences of a variable x to another variable y, the resulting rule is no longer provable (by means of that proof). We show that the decomposition method from [13] can be restricted to general ruloids. Next we show that general ruloids preserve bounded lookahead. This opens the door to deriving congruence formats that allow bounded lookahead.

As a concrete example we extend the partial trace format from [5] with bounded lookahead. We prove that the resulting partial trace format is preserved by general ruloids, which implies that it is a congruence format for the partial trace preorder. This answers the open question from [5].

Concluding, this paper develops machinery to cope with bounded lookahead in the context of structural operational semantics and modal logic, and applies it to obtain a congruence format for the partial trace preorder. We conjecture that the same machinery also makes it possible to develop congruence formats that allow bounded lookahead for decorated trace and ready simulation semantics, which would provide a positive answer to another open question in [5].

The set-up of the paper is as follows. Section 2 contains technical preliminaries and defines a congruence format for the partial trace preorder that allows bounded lookahead. Section 3 presents the notion of a structured proof, in which different variables are collapsed only if this is required by the proof. A rule is called general if it can be derived by a structured proof. It is shown that each derivable rule is the substitution instance of a general rule. In Section 4 and 5 it is argued that the syntactic restrictions of our partial trace format are inherited by general rules. In Section 6 the developed machinery is used to prove that our partial trace format guarantees that the partial trace preorder induced by a TSS is a congruence. The appendix contains omitted proofs.

2 Preliminaries

This section presents the basic notions of partial trace semantics, modal logic and structural operational semantics, as well as the decomposition method from [13]. It also contains a process algebra for process creation from the literature, which exemplifies the concepts from structural operational semantics and serves as an application of our partial trace format that allows lookahead.

2.1 Hennessy-Milner logic

A *labelled transition system (LTS)* is a pair $(\mathbb{P}, \rightarrow)$ with $\mathbb{P}$ a set of *processes* and $\rightarrow \subseteq \mathbb{P} \times A \times \mathbb{P}$ a *transition relation*, for a set A of *actions*. We call each $(p, a, q) \in \rightarrow$ a *transition*, and write it as $p \xrightarrow{a} q$. A sequence $a_1 \cdots a_n \in A^*$ is a *(partial) trace* of a $p \in \mathbb{P}$ if $p \xrightarrow{a_1} \cdots \xrightarrow{a_n} p'$ for some $p' \in \mathbb{P}$. We write $p \sqsubseteq_T q$ if the set of partial traces of p is included in that of q.

Properties of processes can be formulated in modal logic. Hennessy-Milner logic (HML) [19] characterises the bisimulation equivalence relation on processes. The set $\mathbb{O}$ of *HML formulas* is defined by the BNF grammar $\varphi ::= \bigwedge_{i \in I} \varphi_i \mid \langle a \rangle \varphi \mid \neg \varphi$ where a ranges over A and I is any index set. The satisfaction relation $\models \subseteq \mathbb{P} \times \mathbb{O}$ is defined as usual. In particular, $p \models \langle a \rangle \varphi$ iff $p \xrightarrow{a} p'$ and $p' \models \varphi$ for some $p' \in \mathbb{P}$. We use $\varphi_1 \wedge \varphi_2$ and $\top$ as abbreviations of $\bigwedge_{i \in \{1,2\}} \varphi_i$ and $\bigwedge_{i \in \emptyset} \varphi_i$, respectively. We write $\varphi \equiv \varphi'$ if $p \models \varphi \Leftrightarrow p \models \varphi'$ for each process p in each LTS.

The set $\mathbb{O}_T$ of *partial trace observations* is defined by the BNF grammar $\varphi ::= \top \mid \langle a \rangle \varphi$ where a ranges over A. Given an LTS $(\mathbb{P}, \rightarrow)$, let $\mathcal{O}_T(p)$ denote $\{\varphi \in \mathbb{O}_T \mid p \models \varphi\}$. The class $\mathbb{O}_T^{\equiv}$ denotes the closure of $\mathbb{O}_T$ under $\equiv$.

▶ **Proposition 1** ([16]). *$p \sqsubseteq_T q \Leftrightarrow \mathcal{O}_T(p) \subseteq \mathcal{O}_T(q)$.*

2.2 Transition system specifications

V is an infinite set of variables with $|V| \geq |A|$. Let $var(S)$ denote the set of variables that occur in a syntactic object S. We say that S is *closed* if $var(S) = \emptyset$. A *signature* is a set Σ of function symbols $f \notin V$, with $|\Sigma| \leq |V|$, equipped with an arity function $ar : \Sigma \rightarrow \mathbb{N}$. The set $\mathbb{T}(\Sigma)$ of *terms* over a signature Σ is defined recursively by: $V \subseteq \mathbb{T}(\Sigma)$, and if $f \in \Sigma$ and $t_1, \ldots, t_{ar(f)} \in \mathbb{T}(\Sigma)$ then $f(t_1, \ldots, t_{ar(f)}) \in \mathbb{T}(\Sigma)$. Let $\mathrm{T}(\Sigma)$ denote the set of closed terms over Σ.

A Σ-*substitution* σ is a function from V to $\mathbb{T}(\Sigma)$. Let $\sigma(S)$ denote the syntactic object obtained from S by replacing each occurrence of all $x \in V$ in S by $\sigma(x)$. A Σ-substitution σ is *closed* if $\sigma(x) \in \mathrm{T}(\Sigma)$ for all $x \in V$.

A Σ-*literal* (or *transition*) is an expression $t \xrightarrow{a} t'$ with $t, t' \in \mathbb{T}(\Sigma)$ and $a \in A$. A *transition rule* is of the form $\frac{H}{\alpha}$ with H a set of Σ-literals (the *premises* of the rule) and α a Σ-literal (the *conclusion*). The left- and right-hand side of α are called the *source* and *target* of the rule. A *transition system specification (TSS)* is a pair (Σ, R) with R a collection of transition rules over Σ. The purpose of a TSS (Σ, R) is to specify an LTS $(\mathrm{T}(\Sigma), \rightarrow)$ with as processes the closed terms over Σ and as transition relation a set of closed literals $\rightarrow \subseteq \mathrm{T}(\Sigma) \times A \times \mathrm{T}(\Sigma)$.

Let $P = (\Sigma, R)$ be a TSS. An *irredundant proof* from P of a transition rule $\frac{H}{\alpha}$ is a well-founded, upwardly branching tree in which the nodes are labelled by Σ-literals and some of the leaves are marked "hypothesis", such that: the root is labelled by α, H is the set of labels of the hypotheses, and if β is the label of a node q which is not a hypothesis and K is the set of labels of the nodes directly above q, then $\frac{K}{\beta}$ is a substitution instance of a rule

in R. A *proof* from P of $\frac{K}{\alpha}$ is an irredundant proof from P of $\frac{H}{\alpha}$ with $H \subseteq K$. We note that if a leaf in a proof from P is not a hypothesis, it is a substitution instance of a rule $\frac{\emptyset}{\beta}$ in R.

The proof of $\frac{H}{\alpha}$ is called irredundant because H must equal (instead of include) the set of labels of the hypotheses. Rules that are derived with an irredundant proof may inherit certain syntactic structure from the transition rules in the TSS from which they are derived; in classic proofs this syntactic structure is usually lost, because arbitrary literals can be added as premises of derived rules. Irredundancy of proofs is essential in applications of our decomposition method to derive congruence formats for TSSs [5].

2.3 Syntactic restrictions on transition rules

We present some definitions for transition rules; the majority stems from [18]. A rule is *univariate* if its source does not contain multiple occurrences of the same variable. In a *tytt rule*, the right-hand sides of premises are distinct variables that do not occur in the source. A univariate tytt rule is *tyxt* if its source is a variable, and *tyft* if its source contains exactly one function symbol. A tytt rule is *xytt* if the left-hand sides of its premises are variables. An *xyft rule* is a tyft rule that is also an xytt rule.

The *dependency graph* of a rule with premises $\{t_i \xrightarrow{a_i} t'_i \mid i \in I\}$ is a directed graph with as edges $\{\langle x, y\rangle \mid x \in var(t_i) \text{ and } y \in var(t'_i) \text{ for some } i \in I\}$. A rule is *well-founded* if each backward chain of edges in its dependency graph is finite. It has *lookahead* if there is a variable in the right-hand side of a premise that also occurs in the left-hand side of a premise; the lookahead is *bounded* if each forward chain of edges in the dependency graph is finite.

A variable in a rule is *free* if it occurs in neither the source nor the right-hand sides of premises of this rule. A rule is *pure* if it is well-founded and does not contain free variables.

Each combination of syntactic restrictions on transition rules induces a corresponding syntactic format of the same name for TSSs. For example, a TSS is in *pure tyft/tyxt format* iff it contains only pure tyft and tyxt rules.

2.4 Partial trace format

A preorder is a *precongruence* if, for all functions f in the signature, $p_i \sqsubseteq q_i$ for all $i = 1, \ldots, ar(f)$ implies $f(p_1, \ldots, p_{ar(f)}) \sqsubseteq f(q_1, \ldots, q_{ar(f)})$. Likewise, an equivalence is a *congruence* if it is preserved by all functions in the signature. Here we extend the precongruence format for the partial trace preorder from [5] with bounded lookahead, answering an open question from [5].

Let Λ be a predicate on $\{(f, i) \mid f \in \Sigma, 1 \leq i \leq ar(f)\}$; intuitively it marks those arguments of function symbols that may contain processes that have started running. For example, the first argument of the sequential composition operator from process algebra is typically marked by Λ, but the second argument of this operator generally is not (cf. the process algebra APC in Section 2.5). If $\Lambda(f, i)$, then the argument i of f is called Λ-*liquid*; else it is Λ-*frozen*. An occurrence of a variable x in a term $t \in \mathbb{T}(\Sigma)$ is *[at a]* Λ-*liquid [position]* if either $t = x$, or $t = f(t_1, \ldots, t_{ar(f)})$ and the occurrence of x is Λ-liquid in t_i for some liquid argument i of f. A variable in a tytt rule over Σ is Λ-*floating* if either it occurs as the right-hand side of a premise, or it occurs exactly once in the source, at a Λ-liquid position.

▶ **Definition 2.** Let Λ be a predicate on the arguments of function symbols. A tytt rule is Λ-*partial trace safe* if:

- it has bounded lookahead, and
- each Λ-floating variable has at most one occurrence in total in the left-hand sides of the premises and in the target; this occurrence must be at a Λ-liquid position.

A TSS is in *partial trace format* if it is in tyft/tyxt format and its rules are Λ-partial trace safe with respect to some Λ.

This format extends the partial trace format from [5] in allowing bounded lookahead. By abuse of terminology we reuse the format name from [5] for our more liberal format.

▶ Remark. If a TSS is in partial trace format, then there is a smallest predicate Λ for which all its rules are Λ-partial trace safe; it is *generated* by the second condition above.

This paper develops the machinery to prove that the partial trace preorder induced by a TSS in partial trace format is a precongruence (see Corollary 37). The next example shows that the partial trace format cannot allow unbounded lookahead.

▶ **Example 3.** Let $p \xrightarrow{a} r_i$ and $r_{i+1} \xrightarrow{a} r_i$ for all $i \in \mathbb{Z}_{\geq 0}$, and $q \xrightarrow{a} q$. Clearly $q \sqsubseteq_T p$, as the partial traces of both processes are a^i for all $i \in \mathbb{Z}_{\geq 0}$. Let the unary function symbol f be defined by the xyft rule $\frac{\{x_i \xrightarrow{a} x_{i+1} | i \in \mathbb{Z}_{\geq 0}\}}{f(x_0) \xrightarrow{b} \mathbf{0}}$. where $\mathbf{0}$ is a constant. Then $f(q) \not\sqsubseteq_T f(p)$, because $f(q) \xrightarrow{b} \mathbf{0}$ while $f(p)$ cannot perform a b-transition.

The next example shows that the partial trace format cannot allow multiple occurrences of a Λ-floating variable in left-hand sides of premises.

▶ **Example 4.** Let $p \xrightarrow{a} p'$, $p \xrightarrow{a} p''$, $p' \xrightarrow{b} \mathbf{0}$, and $p'' \xrightarrow{c} \mathbf{0}$. Moreover, let $q \xrightarrow{a} q'$, $q' \xrightarrow{b} \mathbf{0}$, and $q' \xrightarrow{c} \mathbf{0}$. Clearly $q \sqsubseteq_T p$, as the completed traces of both processes are ab and ac. Let the unary function symbol f be defined by the xyft rule $\frac{x \xrightarrow{a} y \quad y \xrightarrow{b} z \quad y \xrightarrow{c} z'}{f(x) \xrightarrow{d} \mathbf{0}}$. Then $f(q) \not\sqsubseteq_T f(p)$, because $f(q) \xrightarrow{d} \mathbf{0}$ while $f(p)$ cannot perform a d-transition.

Since $p \sqsubseteq_T q$ allows that $p \not\xrightarrow{a}$ (i.e., p cannot perform any a-transitions) while $q \xrightarrow{a} q'$, clearly the partial trace format cannot contain so-called negative premises $t \not\xrightarrow{a}$; cf. [5, Example 13 – aliased 11.2]. (For partial trace *equivalence*, Λ-frozen variables can be allowed to occur in negative premises; see [5, Theorem 9 – aliased 11.3].) Moreover, negative premises do not combine well with lookahead; cf. [13, Example 7 – aliased 3.15]. For these reasons the current paper focuses on so-called positive TSSs that do not contain negative premises.

2.5 Application: Algebra for process creation

Baeten & Vaandrager [2] defined a process algebra APC for process creation. Its structural operational semantics contains one transition rule with lookahead. From our congruence format it follows immediately that the partial trace preorder is a precongruence for APC. For simplicity only part of its syntax and semantics is presented here; some auxiliary operators needed for the axiomatisation and the encapsulation operator are omitted.

APC contains the following constants: actions a from a set A, successful termination ε, and deadlock δ. The rules are:

$$\frac{}{a \xrightarrow{a} \varepsilon} \qquad \frac{}{\varepsilon \xrightarrow{\surd} \delta}$$

Let e range over $A \cup \{\surd\}$. The two rules for the binary alternative composition operator $+$ are:

$$\frac{x_1 \xrightarrow{e} y_1}{x_1 + x_2 \xrightarrow{e} y_1} \qquad \frac{x_2 \xrightarrow{e} y_2}{x_1 + x_2 \xrightarrow{e} y_2}$$

The asymmetric binary parallel composition $\between$ assumes a partially defined communication function $| : A \times A \to A$; it passes through a termination signal $\surd$ from the right side only.

$$\frac{x_1 \xrightarrow{a} y_1}{x_1 \between x_2 \xrightarrow{a} y_1 \between x_2} \qquad \frac{x_2 \xrightarrow{e} y_2}{x_1 \between x_2 \xrightarrow{e} x_1 \between y_2} \qquad \frac{x_1 \xrightarrow{a} y_1 \quad x_2 \xrightarrow{b} y_2 \quad a|b = c}{x_1 \between x_2 \xrightarrow{c} y_1 \between y_2}$$

The second and third rule rule for the binary sequential composition operator $\cdot$ below are unusual: they spawn a parallel component. The last rule contains lookahead.

$$\frac{x_1 \xrightarrow{a} y_1}{x_1 \cdot x_2 \xrightarrow{a} y_1 \cdot x_2} \qquad \frac{x_1 \xrightarrow{\surd} y_1 \quad x_2 \xrightarrow{e} y_2}{x_1 \cdot x_2 \xrightarrow{e} y_1 \between y_2} \qquad \frac{x_1 \xrightarrow{\surd} y_1 \quad y_1 \xrightarrow{a} z_1 \quad x_2 \xrightarrow{b} y_2 \quad a|b=c}{x_1 \cdot x_2 \xrightarrow{c} z_1 \between y_2}$$

Finally, the unary operator **new**, originating from [1], creates a process that can be put in parallel with an existing process, in view of the peculiar semantics of sequential composition.

$$\frac{}{\mathbf{new}(x) \xrightarrow{\surd} x \cdot \delta} \qquad \frac{x \xrightarrow{a} y}{\mathbf{new}(x) \xrightarrow{a} \mathbf{new}(y)}$$

These are all tyft rules, because in each rule the source contains a single function symbol and the variables in the source and in the right-hand sides of the premises are all distinct. Furthermore, these rules clearly all have bounded lookahead. We take the arguments of $\between$ and **new** and the first argument of $\cdot$ to be Λ-liquid, and the arguments of $+$ and the second argument of $\cdot$ to be Λ-frozen. Each Λ-floating variable has at most one occurrence in total in the left-hand sides of the premises and in the target; this occurrence is in all cases at a Λ-liquid position. Hence the TSS is in the partial trace format. The transition rules for the omitted operators are also in this format. So the partial trace preorder is a precongruence with regard to APC.

2.6 Decomposition of HML formulas

In [13] it was shown how to decompose HML formulas with respect to process terms, given a TSS in tyft/tyxt format. The decomposition method uses a collection of pure xytt rules extracted from this TSS, called *ruloids*. We require that there is a proof of a transition $p \xrightarrow{a} q$, with p a closed substitution instance of a term t, iff there exists a proof that uses at the root a ruloid with source t.

▶ **Definition 5.** A collection $\mathcal{R}$ of pure xytt rules is called *a suitable set of ruloids* for a TSS $P = (\Sigma, R)$ if for each $t \in \mathbb{T}(\Sigma)$, $p \in \mathrm{T}(\Sigma)$ and closed substitution σ, the transition $\sigma(t) \xrightarrow{a} p$ is provable from P iff there are a ruloid $\frac{H}{t \xrightarrow{a} u} \in \mathcal{R}$ and a closed substitution σ' where $\sigma'(\alpha)$ is provable from P for all $\alpha \in H$, $\sigma'(t) = \sigma(t)$ and $\sigma'(u) = p$.

Let $\mathcal{R}$ be a collection of ruloids with bounded lookahead that is suitable for a TSS P. The following definition from [13] assigns to each term $t \in \mathbb{T}(\Sigma)$ and each observation $\varphi \in \mathbb{O}$ a collection $t^{-1}_{\mathcal{R}}(\varphi)$ of decomposition mappings $\psi : V \to \mathbb{O}$. A closed substitution instance $\sigma(t)$ satisfies φ iff for some $\psi \in t^{-1}_{\mathcal{R}}(\varphi)$, $\sigma(x)$ satisfies the formula $\psi(x)$ for all $x \in var(t)$.

▶ **Definition 6.** Let $\mathcal{R}$ be a collection of ruloids with bounded lookahead, suitable for a TSS $P = (\Sigma, R)$. Then $\cdot^{-1}_{\mathcal{R}} : \mathbb{T}(\Sigma) \to (\mathbb{O} \to \mathcal{P}(V \to \mathbb{O}))$ is defined by:

- $\psi \in t^{-1}_{\mathcal{R}}(\langle a \rangle \varphi)$ iff there is a ruloid $\frac{H}{t \xrightarrow{a} u} \in \mathcal{R}$ and a $\chi \in u^{-1}_{\mathcal{R}}(\varphi)$ such that $\psi : V \to \mathbb{O}$ is given by

$$\psi(x) = \begin{cases} \bigwedge_{(x \xrightarrow{b} y) \in H} \langle b \rangle \psi(y) \ \wedge \ \chi(x) & \text{if } x \in var(u) \\ \bigwedge_{(x \xrightarrow{b} y) \in H} \langle b \rangle \psi(y) & \text{if } x \notin var(u). \end{cases}$$

- $\psi \in t^{-1}_{\mathcal{R}}(\bigwedge_{i \in I} \varphi_i)$ iff $\psi(x) = \bigwedge_{i \in I} \psi_i(x)$ where $\psi_i \in t^{-1}_{\mathcal{R}}(\varphi_i)$ for all $i \in I$.

- $\psi \in t_{\mathcal{R}}^{-1}(\neg\varphi)$ iff there is a function $h : t_{\mathcal{R}}^{-1}(\varphi) \to var(t)$ such that $\psi : V \to \mathbb{O}$ is given by $\psi(x) = \bigwedge_{\chi \in h^{-1}(x)} \neg\chi(x)$.

This recursive definition is well-founded because the ruloid employed in the case $t_{\mathcal{R}}^{-1}(\langle a\rangle\varphi)$ has bounded lookahead.

We note that, in contrast to the setting of [5] without lookahead, $x \notin var(t)$ here does not imply $\psi(x) \equiv \top$.

▶ **Example 7.** Consider the TSS of xyft rules $\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{b} g(y)}$ and $\frac{x \xrightarrow{c} y \quad y \xrightarrow{d} z}{g(x) \xrightarrow{e} \mathbf{0}}$. Let the suitable collection of ruloids $\mathcal{R}$ contain $\frac{y \xrightarrow{a} x}{f(y) \xrightarrow{b} g(x)}$ and $\frac{x \xrightarrow{c} y \quad y \xrightarrow{d} z}{g(x) \xrightarrow{e} \mathbf{0}}$.

We calculate a $\psi \in f(y)_{\mathcal{R}}^{-1}(\langle b\rangle\langle e\rangle\top)$. For a start, the ruloid $\frac{y \xrightarrow{a} x}{f(y) \xrightarrow{b} g(x)}$ yields $\psi(y) = \langle a\rangle\psi(x)$ and $\psi(x) = \chi(x)$ for some $\chi \in g(x)_{\mathcal{R}}^{-1}(\langle e\rangle\top)$. For the calculation of χ we use the ruloid $\frac{x \xrightarrow{c} y \quad y \xrightarrow{d} z}{g(x) \xrightarrow{e} \mathbf{0}}$. This yields $\chi(x) = \langle c\rangle\chi(y)$ and $\chi(y) = \langle d\rangle\chi(z)$ and $\chi(z) = \top$. Concluding, $\psi(y) = \langle a\rangle\langle c\rangle\langle d\rangle\top$.

The syntactic overloading of y, i.e. its occurrence in both the first and second ruloid, underlines the importance of the separate case in the definition of $\psi \in t_{\mathcal{R}}^{-1}(\langle a\rangle\varphi)$ in Definition 6, for $x \notin var(u)$. Else we would get $\psi(y) = \langle a\rangle\psi(x) \wedge \chi(y)$, yielding a spurious conjunct $\langle d\rangle\top$ for $\psi(y)$.

We reformulate the decomposition result from [13].

▶ **Theorem 8.** *Let $\mathcal{R}$ be a collection of ruloids with bounded lookahead, suitable for a TSS $P = (\Sigma, R)$. Then for each $t \in \mathbb{T}(\Sigma)$, $\sigma : V \to \mathrm{T}(\Sigma)$ and $\varphi \in \mathbb{O}$:*

$$\sigma(t) \models \varphi \quad \Leftrightarrow \quad \exists \psi \in t_{\mathcal{R}}^{-1}(\varphi)\, \forall x \in var(t) : \sigma(x) \models \psi(x)$$

In [13] P was required to be in tyft/tyxt format, and a specific collection $\mathcal{R}$ was constructed. However, the proof of Theorem 8 only uses that $\mathcal{R}$ has the property of Definition 5. The requirement that P be in tyft/tyxt format was needed merely to ensure that such an $\mathcal{R}$ can be found.

2.7 Construction of ruloids

We briefly sketch the extraction of ruloids from a TSS P in tyft/tyxt format, as employed in [13]. First, employing a conversion from [18], if the source of a rule is of the form x then this variable is replaced by a term $f(x_1, \ldots, x_{ar(f)})$ for each $f \in \Sigma$. This yields an intermediate TSS $P^\dagger$ in tyft format, of which all rules are provable from P. Next, using a construction from [11], the left-hand sides of premises are reduced to variables. Roughly the idea is, given a premise $f(t_1, \ldots, t_{ar(f)}) \xrightarrow{a} y$ in a rule r, and a rule $\frac{H}{f(x_1,\ldots,x_{ar(f)}) \xrightarrow{a} t}$, to transform r by replacing the aforementioned premise by H, y by t, and the x_i by the t_i; this is repeated (transfinitely) until all premises with a non-variable left-hand side have disappeared. Each infinite sequence of such substitutions converges to an infinite sequence of variable replacements; these variables are unified. The result is a TSS $P^\ddagger$ in xyft format, of which all rules are provable from $P^\dagger$ [11]. Next, the premises for which there is no backward chain in the dependency graph to a variable in the source are eliminated, by substituting closed terms for the variables in such premises. The resulting TSS in pure xyft format is denoted by P^+; its rules are provable from $P^\ddagger$ [11]. By [13, Proposition 3 – aliased 3.4] the pure xytt rules irredundantly provable from P^+ constitute a suitable collection of ruloids for P.

▶ **Example 9.** Consider the process algebra APC from Section 2.5, with $A = \{a, b, c, d, e\}$ and a communication function that includes $a|b = c$ and $c|d = e$. The next ruloid can be derived using the third rule for parallel composition and the third rule for sequential composition.

$$\frac{x_1 \xrightarrow{\surd} y_1 \quad y_1 \xrightarrow{a} z_1 \quad x_2 \xrightarrow{b} y_2 \quad x_3 \xrightarrow{d} y_3}{(x_1 \cdot x_2) \between x_3 \xrightarrow{e} (z_1 \between y_2) \between y_3}$$

Using [11, Lemma 2.10], it follows that these ruloids are provable from P. Hence another suitable collection of ruloids is given by *all* pure xytt rules provable from P, or all pure xytt rules *irredundantly* provable from P. In Section 3 we will define *P-general ruloids* such that each pure xytt rule irredundantly provable from P is a substitution instance of a P-general ruloid with the same source. This implies that the collection of P-general ruloids is suitable.

In [13] an additional step in the construction of ruloids was made to ensure that they all have bounded lookahead. Each ruloid with unbounded lookahead was replaced by an equivalent ruloid with bounded lookahead, by endowing each infinite forward chain with an ordinal count-down. However, the ruloids produced by this step violate most congruence formats. (A notable exception is the full tyft/tyxt format, as a congruence format for bisimulation semantics; see [13, Corollary 1 – aliased 4.1].) In particular, starting with a TSS in partial trace format, this step produces ruloids in which Λ-floating variables may have multiple occurrences in left-hand sides of premises. This is no surprise, as in Example 3 it was shown that the partial trace format must exclude unbounded lookahead. Here we avoid this additional step by considering only TSSs in tyft/tyxt format with bounded lookahead. We will prove in Section 4 that for such TSSs P, each P-general ruloid has bounded lookahead.

3 Structured proofs and general rules

The following example shows that the second condition of the partial trace format is not always preserved by irredundant proofs of pure xytt rules.

▶ **Example 10.** Consider the TSS with bounded lookahead consisting of the xyft rules

$$\frac{\{y_{i+1} \xrightarrow{a} y_i \mid i \in \mathbb{Z}_{\geq 0}\}}{f(x) \xrightarrow{b} g(x, y_0)} \qquad \frac{}{x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{c} y}{f(x) \xrightarrow{d} f(y)}$$

In view of the third rule, the argument of f is Λ-liquid. So by the first rule, the arguments of g are Λ-liquid as well. Clearly the TSS is in partial trace format.

Substituting z for x and for all y_i in the first rule as well as for x in the second rule, we can derive the rule $\frac{}{f(z) \xrightarrow{b} g(z,z)}$. The two occurrences of the Λ-floating variable z in the target violate the partial trace format.

This counter-example is spurious: the derived rule is a substitution instance of the rule $\frac{}{f(z) \xrightarrow{b} g(z,z')}$ with $z \neq z'$, which does adhere to the partial trace format. The latter rule can be derived in a similar fashion, by substituting z' (instead of z) for all y_i in the first rule as well as for x in the second rule. The irredundant proof of $\frac{}{f(z) \xrightarrow{b} g(z,z)}$ is not "general": there is no need to replace the two arguments of g in the target by the same variable. On the other hand, the same variable must be substituted for all the y_i, so that the premises of the first rule can be derived by the second rule in the TSS.

We will show that a suitable collection of ruloids is formed by the so-called "general" pure xytt rules, which are derived by an irredundant proof in which terms $\sigma(x)$ and $\sigma(y)$ with

$x \neq y$ only have variables in common if this is imposed by the proof. For example, let the TSSs P_1 and P_2 both contain the rule $\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{b} y}$, while P_1 contains $\frac{}{g(x) \xrightarrow{a} x}$ and P_2 contains $\frac{}{g(x) \xrightarrow{a} y}$. The rule $\frac{}{f(g(z)) \xrightarrow{b} z}$ is irredundantly provable from both P_1 and P_2; however, this rule is P_1-general but not P_2-general. In contrast, the rule $\frac{}{f(g(z)) \xrightarrow{b} z'}$ is P_2-general, but not provable from P_1.

In Section 4 and 5 it will be shown that general rules do preserve the partial trace format. To formally define the notion of a general rule, we first provide an alternative characterisation of (irredundant) provability, roughly following [11]. We will consider irredundant proofs with minimal variable unifications. However, irredundant proofs abstract away from the variables that are being unified; in contrast, proof structures contain variable unifications explicitly.

▶ **Definition 11.** Let $\pi = (B, \alpha, \varphi)$ where:

- B is a set of transition rules which do not have any variables in common,
- α is a literal of the form $s_\pi \xrightarrow{a} w$ with $s_\pi \in \mathbb{T}(\Sigma)$ and $w \in V$ where $var(\alpha) \cap var(B) = \emptyset$ and $w \notin var(s_\pi)$, and
- φ is an injective mapping from B to $\{\alpha\} \cup \{\beta \mid \beta$ a premise of a rule in $B\}$, such that
  - the conclusion of b and $\varphi(b)$ carry the same action for all $b \in B$, and
  - all chains $b_0, b_1, b_2, \ldots$ in B with each $\varphi(b_{i+1})$ a premise of b_i are finite.

In the sequel, the *premises* of π are α and the premises of rules in B, and $top(\pi)$ denotes the collection of premises of π that are outside the image of φ.

A rule $b_0 \in B$, or a premise of b_0, is said to be *above* a premise β if there exists a chain $b_1, \ldots, b_n$ in B with $\varphi(b_i)$ a premise of b_{i+1} for all $0 \leq i < n$ and $\varphi(b_n) = \beta$.

π is a *proof structure* if each rule in B is above α. It is a proof structure *over* a TSS $P = (\Sigma, R)$ if each rule in B is in R modulo alpha-conversion (i.e., renaming of variables).

A substitution σ *matches* π if $\sigma(s_\pi) = s_\pi$ and, for all $b \in B$, $\sigma(conclusion(b)) = \sigma(\varphi(b))$.

▶ **Proposition 12.** *A rule $\frac{H}{\gamma}$ is provable from a TSS P iff there exists a proof structure $\pi = (B, \alpha, \varphi)$ over P and a substitution σ that matches π, such that $\sigma(top(\pi)) \subseteq H$ and $\sigma(\alpha) = \gamma$. It is irredundantly provable if $\sigma(top(\pi)) = H$.*

Proof. Given a proof structure $\pi = (B, \alpha, \varphi)$ and a matching substitution σ, an irredundant proof of $\sigma(\frac{top(\pi)}{\alpha})$ is obtained as the (multi)set of premises of π, each premise β labelled by $\sigma(\beta)$, ordered into a tree by the "above" relation; $top(\pi)$ will be the set of "hypotheses".

Conversely, each irredundant proof π of a rule $\frac{H}{t \xrightarrow{a} u}$ can be converted into a proof structure (B, α, φ) by replacing each non-hypothesis node in π by an incarnation of the transition rule applied in that node, where, using alpha-conversion, all incarnations are given disjoint sets of variables. Take $\alpha := (t \xrightarrow{a} w)$ for a fresh variable w. When the rules for each two nodes have disjoint variable sets, the substitutions used in all nodes can be united into one substitution matching the entire proof structure.

One point of concern in the above construction is whether there are enough variables to allocate a disjoint set of variables to the rules for each node in π. As V is infinite, this constraint is satisfied if the number of nodes in π is not larger than $|V|$, which is the case if the branching degree of π, i.e. the number of premises in each rule, is no larger than $|V|$. In [11] this was achieved by means of a requirement on TSSs, namely of being "small". Here we just make sure that the set of *all* literals is not larger than $|V|$. This is a simple consequence of our requirements that $|\Sigma| \leq |V|$ and $|A| \leq |V|$ (cf. Lemma 6 – aliased 6.4 – in [5]). ◀

A different proof of a small variation of this characterisation can be found in [11].

▶ **Example 13.** Consider the TSS in Example 10. As running example in this section we introduce a proof structure of the following shape, where downward arrows depict φ.
One matching substitution σ_1 maps w to $g(z,z)$ and all other variables to z. Another matching substitution σ_2 maps w to $g(z,z')$, x to z and all other variables to z'.

A variable x in a proof structure π is *prime* if there exists a matching substitution σ for π with $\sigma(x)$ a variable. The relation $\sim_\pi$ relates those prime variables that are mapped to the same term by each matching substitution for π.

$$\dfrac{\begin{array}{cc} \cdots & x_1 \xrightarrow{a} x_1 \quad x_0 \xrightarrow{a} x_0 \\ & \downarrow \qquad\quad \downarrow \\ \cdots & y_2 \xrightarrow{a} y_1 \quad y_1 \xrightarrow{a} y_0 \end{array}}{\begin{array}{c} f(x) \xrightarrow{b} g(x,y_0) \\ \downarrow \\ f(z) \xrightarrow{b} w \end{array}}$$

▶ **Definition 14.** Let $\pi = (B, \alpha, \varphi)$ be a proof structure. Let $\sim_\pi$ be the least equivalence relation on $\mathbb{T}(\Sigma)$ satisfying:
- if $b = \frac{H}{t \xrightarrow{a} u} \in B$ and $\varphi(b) = (t' \xrightarrow{a} u')$ then $t \sim_\pi t'$ and $u' \sim_\pi u$, and
- if $f(t_1, \ldots, t_k) \sim_\pi f(u_1, \ldots, u_k)$ then $t_i \sim_\pi u_i$ for all $i = 1, \ldots, k$.

A variable $x \in var(\pi)$ is *composite* if $x \sim_\pi t$ with $t \notin V$, and *prime* otherwise.

▶ **Observation 15.** *A substitution σ matches π iff $\sigma(s_\pi) = s_\pi$ and $\sigma(t) = \sigma(u)$ for all terms $t, u \in \mathbb{T}(\Sigma)$ with $t \sim_\pi u$.*

▶ **Example 16.** For the proof structure in Example 13, $x_i \sim_\pi y_i$ and $x_i \sim_\pi y_{i+1}$ for all $i \in \mathbb{Z}_{\geq 0}$. Moreover, $x \sim_\pi z$ and $w \sim_\pi g(x, y_0)$. So the two equivalence classes of prime variables modulo $\sim_\pi$ are $\{x_i, y_i \mid i \in \mathbb{Z}_{\geq 0}\}$ and $\{x, z\}$.

A substitution is *minimal* for a proof structure if it is matching and provides as little syntactic structure to (substitution instances of) variables as possible, and induces as few identifications of variables as possible.

▶ **Definition 17.** A substitution ρ for a proof structure π is *minimal* if:
- $\rho(x) = x$ for each $x \in var(s_\pi)$ and $\rho(x) \in V$ for each prime variable $x \in var(\pi)$,
- $\rho(x) = \rho(y)$ iff $x \sim_\pi y$, for each pair of prime variables $x, y \in var(\pi)$, and
- $\rho(t) = \rho(u)$ for each two terms $t, u \in \mathbb{T}(\Sigma)$ with $t \sim_\pi u$.

A rule r is *P-general* if there exists a proof structure $\pi = (B, \alpha, \varphi)$ over P and a substitution ρ that is minimal for π such that $r = \rho(\frac{top(\pi)}{\alpha})$. The pair (π, ρ) is called a *structured proof* of r from P.

▶ **Example 18.** The first matching substitution σ_1 for the proof structure in Example 13 is not minimal, because it maps the variables in the two equivalence classes modulo $\sim_\pi$ to the same variable z.

The second matching substitution σ_2 for this proof structure is minimal, meaning that the rule $\frac{}{f(z) \xrightarrow{b} g(z,z')}$ is general with regard to the TSS in Example 10.

The following proposition is a pivotal result for this paper.

▶ **Proposition 19.** *A rule is irredundantly provable from a TSS P iff it is a substitution instance of a P-general rule with the same source.*

Proof. $\boxed{\Leftarrow}$ Let the rule r be a substitution instance of a rule $\rho(\frac{top(\pi)}{s_\pi \xrightarrow{a} w})$ with the same source, where $\pi = (B, s_\pi \xrightarrow{a} w, \varphi)$ is a proof structure over P and ρ a minimal substitution for π. Then $r = \sigma(\rho(\frac{top(\pi)}{s_\pi \xrightarrow{a} w}))$ for some substitution σ. By assumption, $\sigma(\rho(s_\pi)) = \rho(s_\pi) = s_\pi$. By Observation 15 and the third requirement on minimal substitutions, ρ matches π. Therefore, also $\sigma \circ \rho$ matches π, so by Proposition 12 r is irredundantly provable from P.

$\boxed{\Rightarrow}$ Let the rule r be irredundantly provable from P. By Proposition 12 $r = \sigma(\frac{top(\pi)}{s_\pi \xrightarrow{a} w})$ for some proof structure $\pi = (B, s_\pi \xrightarrow{a} w, \varphi)$ and a matching substitution σ. We will now construct a substitution ρ that is minimal for π, and a substitution ν with $\sigma = \nu \circ \rho$. This immediately yields the required result.

For each $\sim_\pi$-equivalence class C of prime variables we pick a $y_C \in C$ and take $\rho(x) := y_C$ for all $x \in C$ – if possible we choose $y_C \in var(s_\pi)$. This way the first two requirements of a minimal substitution are met. In particular, if $x, y \in var(s_\pi)$ with $x \sim_\pi y$ then $\sigma(x) = x$ and $\sigma(y) = y$, which implies that x and y are prime, and by Observation 15 $x = \sigma(x) = \sigma(y) = y$; thus $\rho(x) = x$. Moreover, take $\nu(y_C) := \sigma(y_C)$, so that $\sigma(x) = \sigma(y_C) = \nu(y_C) = \nu(\rho(x))$ for all $x \in C$, using Observation 15. The substitution ν satisfies $\nu(z) = z$ for all other variables z.

With structural induction on $\sigma(x)$ we proceed to define $\rho(x)$ for composite variables $x \in var(\pi)$, such that $\sigma(x) = \nu(\rho(x))$. Simultaneously, with structural induction on $\sigma(t)(= \sigma(u))$, we establish $\rho(t) = \rho(u)$ for each pair of terms t, u with $t \sim_\pi u$.

Let $t, u \notin V$ be terms with $t \sim_\pi u$. Let $t = f(t_1, \ldots, t_k)$ and $u = g(u_1, \ldots, u_m)$. By Observation 15 $\sigma(t) = \sigma(u)$, so $f = g$ and $k = m$. By Definition 14 $t_i \sim_\pi u_i$, so by induction $\rho(t_i) = \rho(u_i)$, for all $i = 1, \ldots, k$, and hence $\rho(t) = \rho(u)$.

Now let $x \in var(\pi)$ be composite; say $x \sim_\pi t$ for some term $t \notin V$. By Observation 15 $\sigma(x) = \sigma(t)$, so for each $y \in var(t)$ the term $\sigma(y)$ is a proper subterm of $\sigma(x)$. By induction, $\rho(y)$ has already been defined before we get to defining $\rho(x)$, and $\sigma(y) = \nu(\rho(y))$. Hence $\rho(t)$ is well-defined, and $\sigma(t) = \nu(\rho(t))$, so we can take $\rho(x) := \rho(t)$, thereby obtaining $\sigma(x) = \nu(\rho(x))$. By the argument above, this definition is independent of the choice of t.

Finally, if $x \sim_\pi y$ and one of these variables is composite, then both are composite and $x \sim_\pi t \sim_\pi y$ for some term $t \notin V$. Now $\rho(x) = \rho(y)$ follows by transitivity. ◀

▶ **Example 20.** With regard to the TSS in Example 10, the irredundantly provable rule $\frac{}{f(z) \xrightarrow{b} g(z,z)}$ is a substitution instance of the general rule $\frac{}{f(z) \xrightarrow{b} g(z,z')}$.

Now we define a *P-general ruloid* as a P-general pure xytt rule. It follows from Proposition 19 that each irredundantly provable pure xytt rule is a substitution instance of a P-general ruloid with the same source, so the P-general ruloids form a suitable collection of ruloids for P.

We now consider TSSs in univariate tytt format; these syntactic restrictions are part of all congruence formats in the literature. The following definition makes relations between different occurrences of a variable z in a structured proof explicit. The underlying idea of its first case is that syntactic structure is inherited in an upward fashion at the left-hand side of each branch of a proof, and in a downward fashion at the right-hand side. The second case expresses that occurrences of a variable x in a rule in the proof inherit syntactic structure from the (unique) occurrence of x in the source or in a right-hand side of a premise of this rule. The third case extends this relationship to variables that are free in a rule in the proof: all occurrences of a free variable x are syntactically linked to each other.

▶ **Definition 21.** Let (π, ρ) with $\pi = (B, \alpha, \varphi)$ be a structured proof from a TSS in univariate tytt format. An *occurrence* of a variable z in this proof is represented by a triple $\theta = (b, \iota, \eta)$ with either $b \in B$ or $b = \alpha$, ι an occurrence (i.e. position) of a variable x in b, and η an

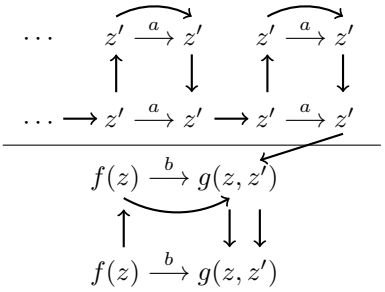
occurrence of z in $\rho(x)$. Sometimes we address such an occurrence as $\langle b, \iota_t, \eta\rangle$ where ι_t is an occurrence of a *term* t in b, and η an occurrence of z in $\rho(t)$.

The relations $\to_z$ and $\leftrightsquigarrow_z$ between the occurrences of z in (π, ρ) are given by:

- if $b = \frac{H}{t \xrightarrow{a} u} \in B$ and $\varphi(b) = (t' \xrightarrow{a} u')$, writing b' for the rule (or α) containing the premise $\varphi(b)$ and ι_t, $\iota_{t'}$, ι_u and $\iota_{u'}$ for the indicated occurrences of t in b, t' in b', u in b and u' in b', respectively, then $\langle b', \iota_{t'}, \eta\rangle \to_z \langle b, \iota_t, \eta\rangle$ for any occurrence η of z in $\rho(t') = \rho(t)$, and $\langle b, \iota_u, \eta'\rangle \to_z \langle b', \iota_{u'}, \eta'\rangle$ for any occurrence η' of z in $\rho(u) = \rho(u')$;
- if $b \in B$ and η is an occurrence of z in $\rho(x)$ for some $x \in var(b)$, then $(b, \iota, \eta) \to_z (b, \iota', \eta)$ where ι is an occurrence of x either in the source of b or in the right-hand side of a premise of b, and ι' is an occurrence of x in the left-hand side of a premise or in the target of b;
- if η is an occurrence of z in $\rho(x)$ with $x \in var(b)$, and either $b = \alpha$, or $b \in B$ and x occurs neither in the source of b nor in the right-hand sides of its premises, then $(b, \iota, \eta) \leftrightsquigarrow_z (b, \iota', \eta)$ for ι and ι' any two different occurrences of x.

Let $\sim_z$ denote the smallest equivalence relation containing $\to_z \cup \leftrightsquigarrow_z$.

▶ **Example 22.** Consider the structured proof from Example 13, after applying the matching substitution σ_2 to it. The relations $\to_z$ and $\to_{z'}$ are depicted by arrows. (The arrows depicting φ have been omitted here.) Relations between different rules (i.e., the vertical ones) are due to the first case of Definition 21, while relations within one rule are due to the second case of Definition 21. Since the TSS in Example 10 does not contain free variables, the third case of Definition 21 does not apply.



We partition the variable occurrences in a rule r into three types: we speak of an *incoming* occurrence if it occurs in the source of r, or in the right-hand side of a premise; an *upwards outgoing* occurrence if it occurs in the left-hand side of a premise; and a *downwards outgoing* occurrence if it occurs in the target of r. This applies to rules $\rho(b)$ associated to a structured proof (π, ρ) with $\pi = (B, \alpha, \varphi)$ and $b \in B$; it also applies to $\rho(\alpha)$ by considering this literal to be a premise. This terminology is motivated by the following observation on the above constructed graph of occurrences of a variable z in a structured proof (π, ρ).

▶ **Observation 23.** *If $(b, \iota, \eta) \to_z (b', \iota', \eta')$ then either*

- *$b' = b$, ι is an incoming occurrence and ι' an (upwards or downwards) outgoing one, or*
- *$\varphi(b')$ is a premise of b, ι is an upwards outgoing occurrence in b, and ι' is an incoming occurrence in b', or*
- *$\varphi(b)$ is a premise of b', ι is a downwards outgoing occurrence in b, and ι' is an incoming occurrence in b'.*

▶ **Example 24.** Consider the arrows in the picture in Example 22 that depict the relations $\to_z$ and $\to_{z'}$. The upward arrows are from an upwards outgoing to an incoming occurrence of z or z', the downward arrows from a downwards outgoing to an incoming occurrence of z or z', the straight horizontal arrows from an incoming to an upwards outgoing occurrence of

z', and the diagonal and curved horizontal arrows from an incoming to a downwards outgoing occurrence of z or z'.

Note that an occurrence $\theta = (b, \iota, \eta)$ of a variable z in a structured proof (π, ρ) refers to an occurrence of z in the rule $\rho(b)$; θ is called *incoming* or *outgoing*, or Γ-*liquid*, for a given predicate Λ on arguments of function symbols, iff the referred occurrence of z in $\rho(b)$ has these properties. Note that θ is *incoming* or *outgoing* iff ι is such an occurrence.

▶ **Observation 25.** *For each outgoing occurrence θ of z in a structured proof (π, ρ) there is at most one incoming occurrence θ' of z with $\theta \to_z \theta'$. There is none iff θ occurs in $top(\pi)$. The occurrence θ' is Λ-liquid (for a given predicate Λ) iff θ is Λ-liquid.*

This last statement is trivial, because θ and θ' refer to the same occurrence in a term $\rho(t)$ occurring in b as well as in b'.

The following key proposition will be needed in the proofs in Section 4 and 5.

▶ **Proposition 26.** *Let $\theta = (b, \iota, \eta)$ be an occurrence of a variable z in a structured proof (π, ρ) from a TSS P in univariate tytt format, with ι either an incoming occurrence in $top(\pi)$ or the only occurrence of a variable x in s_π. Then $\theta \to_z^* \theta'$ for any occurrence θ' of z in (π, ρ), with * reflexive and transitive closure.*

▶ **Example 27.** In Example 22, the occurrence of z in s_π is $\to_z^*$-related to the three other occurrences of z in the structured proof.

4 Preservation of bounded lookahead

We show that for any TSS P in tyft/tyxt format with bounded lookahead, all P-general rules have bounded lookahead. Thus congruence formats that allow bounded lookahead can be derived by means of the decomposition method from [5].

▶ **Definition 28.** For a proof structure $\pi = (B, \alpha, \varphi)$, let $\prec_\pi$ be the least relation on $var(\pi)$ such that:

- if x occurs in the left-hand side of a premise of π, and y in its right-hand side, then $x \prec_\pi y$, and
- if $b = \frac{H}{t \xrightarrow{a} u} \in B$ and $\varphi(b) = (t' \xrightarrow{a} u')$ with $x \in var(t') \wedge y \in var(t)$ or $x \in var(u) \wedge y \in var(u')$, then $x \prec_\pi y$.

▶ **Observation 29.** *Let $(b, \iota, \eta) \to_z (b', \iota', \eta')$ for two occurrences of a variable z in a proof structure (π, ρ), with ι an occurrence of an $x \in var(\pi)$ in b, and ι' of a $y \in var(\pi)$ in b'. If $b = b'$ then $x = y$, and if $b \neq b'$ then $x \prec_\pi y$.*

▶ **Proposition 30.** *Let $\pi = (B, \alpha, \varphi)$ be a proof structure. If all rules in B have bounded lookahead, then there is no infinite chain $x_0 \prec_\pi x_1 \prec_\pi x_2 \prec_\pi \cdots$.*

▶ **Theorem 31.** *Let P be a TSS in univariate tytt format with bounded lookahead. Then all P-general rules have bounded lookahead.*

Proof. Let P be a TSS with bounded lookahead, and r a P-general rule, say with structured proof (π, ρ). If r had unbounded lookahead, then $top(\pi)$ would contain premises $t_i \xrightarrow{a_i} u_i$ for $i \in \mathbb{Z}_{\geq 0}$ with $var(\rho(u_i)) \cap var(\rho(t_{i+1})) \neq \emptyset$ for all $i \in \mathbb{Z}_{\geq 0}$. Thus, for each $i \in \mathbb{Z}_{\geq 0}$, there would be a $y_i \in var(u_i)$, an $x_{i+1} \in var(t_{i+1})$ and some $z \in var(\rho(y_i)) \cap var(\rho(x_{i+1}))$. Let $\theta_i = (b_i, \iota_i, \eta_i)$ be the occurrence of z in (π, ρ) where b_i is the topmost rule with premise $t_i \xrightarrow{a_i} u_i$, ι_i is the occurrence of y_i in u_i in b_i, and η_i the occurrence of z in $\rho(y_i)$. Likewise,

$\xi_{i+1} = (b_{i+1}, \iota'_{i+1}, \eta'_{i+1})$ is the occurrence of z in (π, ρ) where ι'_{i+1} is the occurrence of x_{i+1} in t_{i+1} in b_{i+1}, and η'_{i+1} the occurrence of z in $\rho(x_{i+1})$. By Proposition 26 $\theta_i \to_z^* \xi_{i+1}$. Now Observation 29 gives $y_i \prec_\pi^* x_{i+1}$. Since by definition also $x_i \prec_\pi y_i$ for all $i \in \mathbb{Z}_{\geq 0}$, we have found an infinite chain $x_0 \prec_\pi x_1 \prec_\pi x_2 \prec_\pi \dots$, which with Proposition 30 yields the required contradiction. ◀

The following example shows that the restriction in Theorem 31 to P-general rules is essential.

▶ **Example 32.** Consider the rule $\frac{x \xrightarrow{a} y}{f(x) \xrightarrow{b} \mathbf{0}}$. By substituting z for both x and y we derive $\frac{z \xrightarrow{a} z}{f(z) \xrightarrow{b} \mathbf{0}}$, which contains unbounded lookahead.

▶ **Corollary 33.** *Theorem 8 applies to each TSS P in tyft/tyxt format with bounded lookahead by choosing for $\mathcal{R}$ the collection of all P-general ruloids.*

5 Preservation of Λ-partial trace safeness

We say that a rule is Λ-*infinitary trace safe* if each Λ-floating variable has at most one occurrence in total in the left-hand sides of the premises and in the target; this occurrence must be at a Λ-liquid position.

▶ **Observation 34.** *Let P be a TSS in univariate tytt format for which each rule is Λ-infinitary trace safe, and (π, ρ) a structured proof of a rule r from P. For each Λ-liquid incoming occurrence θ of z in (π, ρ) there is at most one outgoing occurrence θ' of z with $\theta \to_z \theta'$; this occurrence must be at a Λ-liquid position.*

This holds because by Observation 23, case 2 of Definition 21 applies, with $\theta = (b, \iota, \eta)$ and $\theta = (b, \iota', \eta)$. As θ is Λ-liquid, ι must be a Λ-liquid occurrence of a variable x, and η a Λ-liquid occurrence of z in $\rho(x)$. Since P is in univariate tytt format, ι is Λ-floating. Hence ι' is Λ-liquid.

▶ **Theorem 35.** *Let P be a TSS in univariate tytt format for which each rule is Λ-infinitary trace safe. Then each P-general rule is Λ-infinitary trace safe.*

Proof. Let (π, ρ) be a structured proof from P of a P-general rule r, where $\pi = (B, \alpha, \varphi)$ with $\alpha = (s_\pi \xrightarrow{a} w)$. Let z be a Λ-floating variable of r. Then z has a Λ-liquid occurrence $\theta = (b, \iota, \eta)$ in (π, ρ), with ι either an incoming occurrence in $top(\pi)$ or the only occurrence of a variable x in s_π.

Consider any occurrence of z in the left-hand sides of the premises or the target of r. It corresponds with an occurrence $\theta' = (b', \iota', \eta')$ of z in either a left-hand side of a premise in $top(\pi)$ or the right-hand side of α (thus making ι' the occurrence of w). There is no θ'' with $\theta' \to_z \theta''$. This follows from Observation 25 if ι' occurs in a left-hand side of $top(\pi)$, or from Definition 21 if it is the right-hand side of α.

By Proposition 26, $\theta = \theta_0 \to_z \theta_1 \to_z \cdots \to_z \theta_k = \theta'$. Observations 23, 25 and 34 together imply that any occurrence θ_i of z in this chain is Λ-liquid, and moreover that for each such θ_i the next occurrence θ_{i+1} (if it exists) is uniquely determined. Since moreover $\theta_k \not\to_z$ it follows that θ' is uniquely determined. Thus there is at most one occurrence of z in the left-hand sides of the premises of r or in the target of r, and this occurrence is at a Λ-liquid position. ◀

A tytt rule is Λ-partial trace safe iff it is Λ-infinitary trace safe and has bounded lookahead. Thus, Theorems 31 and 35 together say that if all rules of a TSS P in univariate tytt format are Λ-partial trace safe, then so is each P-general rule.

6 Precongruence of partial trace preorder

To prove that for each TSS in partial trace format the induced partial trace preorder is a precongruence, it suffices to show that each formula in $\mathbb{O}_T$ decomposes into formulas in $\mathbb{O}_T^{\equiv}$.

▶ **Proposition 36.** *Let P be a TSS in partial trace format and $\mathcal{R}$ the set of P-general ruloids. For each term t, $\varphi \in \mathbb{O}_T$, $\psi \in t_{\mathcal{R}}^{-1}(\varphi)$ and variable x:*

$$\psi(x) \equiv \bigwedge_{i \in I} \psi_i \text{ with } \psi_i \in \mathbb{O}_T^{\equiv} \text{ for all } i \in I.$$

▶ **Corollary 37.** *If a TSS is in partial trace format, then the partial trace preorder it induces is a precongruence.*

Proof. Consider a TSS P in partial trace format. Let t be a term and σ, σ' closed substitutions with $\sigma(x) \sqsubseteq_T \sigma'(x)$ for all $x \in var(t)$; we need to prove that $\sigma(t) \sqsubseteq_T \sigma'(t)$. Suppose that $\sigma(t) \models \varphi \in \mathbb{O}_T$. Let $\mathcal{R}$ denote the set of P-general ruloids. By Theorem 8 in combination with Corollary 33 there is a $\psi \in t_{\mathcal{R}}^{-1}(\varphi)$ with $\sigma(x) \models \psi(x)$ for all $x \in var(t)$. By Proposition 36, $\psi(x) \equiv \bigwedge_{i \in I_x} \psi_{i,x}$ with $\psi_{i,x} \in \mathbb{O}_T^{\equiv}$ for all $x \in var(t)$ and $i \in I_x$. So $\sigma(x) \models \psi_{i,x}$ for all $x \in var(t)$ and $i \in I_x$. By Proposition 1, $\mathcal{O}_T^{\equiv}(\sigma(x)) \subseteq \mathcal{O}_T^{\equiv}(\sigma'(x))$ for all $x \in var(t)$. This implies $\sigma'(x) \models \psi_{i,x}$ for all $x \in var(t)$ and $i \in I_x$. So $\sigma'(x) \models \psi(x)$ for all $x \in var(t)$. Therefore, by Theorem 8, $\sigma'(t) \models \varphi$. So $\mathcal{O}_T(\sigma(t)) \subseteq \mathcal{O}_T(\sigma'(t))$. Hence, by Proposition 1, $\sigma(t) \sqsubseteq_T \sigma'(t)$. ◀

7 Conclusion and future work

We introduced the notion of a general rule, which has a proof with minimal variable unifications. To this end we used *proof structures* as alternatives for irredundant proofs, because irredundant proofs abstract away from the variables that are being unified. We moreover showed that if a TSS has bounded lookahead, then the same holds for its general rules. This means that the decomposition method of modal formulas from [13] applies directly to TSSs with bounded lookahead, without first having to turn unbounded into bounded lookahead by means of ordinal count-downs. Both the notion of a general rule and the preservation of bounded lookahead were crucial in the derivation of a congruence format for the partial trace preorder, using the decomposition method.

When restricting attention to TSSs whose rules have finitely many premises, the restriction to bounded lookahead can be dropped from the partial trace format. The reason is that unbounded lookahead is eliminated when converting such a TSS to pure xyft format. With this extension included, our format extends the earlier congruence format for partial trace semantics presented in Bloom [4]. The latter can be seen as the restriction of our format to TSSs in tyft format, allowing only rules with finitely many premises, and requiring Λ to hold universally. The binary Kleene star (see [5]) is an example of an operator that falls in our format, and in that of [5], but not in that of [4]. The application to APC in Section 2.5 falls outside the formats of both [5] and [4].

Future work is to extend the results to TSSs with negative premises, and develop a congruence format for partial trace *equivalence* that allows negative premises. We conjecture that the techniques and results introduced in this paper make it possible to develop congruence formats with lookahead for ready simulation and the decorated trace semantics.

Some applications of lookahead mentioned in the introduction require a richer format, which may be based on the basic format given here. There is a rich body of work extending

existing congruence formats with features like time, probabilities and binders. Recently [8] employed the modal decomposition technique to obtain congruence formats for probabilistic semantics. Our approach lays the groundwork to extend those formats with lookahead.

In [14, 12], modal decomposition is used to derive congruence for weak semantics. By extending this work with lookahead, congruence formats may be developed that e.g. cover the lookahead in the τ-rules from [7]. In [17, Section 8] a congruence format for weak bisimilarity with τ-rules and lookahead is presented, but using a bisimulation-specific method.

Acknowledgement. Paulien de Wind observed that the transformation from unbounded to bounded lookahead in [13] violates the partial trace format.

References

1 P. America and J. W. de Bakker. Designing equivalent semantic models for process creation. *Theoretical Computer Science*, 60(2):109–176, 1988. doi:10.1016/0304-3975(88)90048-5.

2 J. C. M. Baeten and F. W. Vaandrager. An algebra for process creation. *Acta Informatica*, 29(4):303–334, 1992. doi:10.1007/BF01178776.

3 Marco Bernardo. Enriching empa with value passing: A symbolic approach based on lookahead. In *Proc. PAPM 1997*, pages 35–49, 1997.

4 B. Bloom. When is partial trace equivalence adequate? *Formal Aspects of Computing*, 6(3):317–338, 1994. doi:10.1007/BF01215409.

5 B. Bloom, W. J. Fokkink, and R. J. van Glabbeek. Precongruence formats for decorated trace semantics. *Transactions on Computational Logic*, 5(1):26–78, 2004. doi:10.1145/963927.963929.

6 B. Bloom, S. Istrail, and A. R. Meyer. Bisimulation can't be traced. *Journal of the ACM*, 42(1):232–268, 1995. doi:10.1145/200836.200876.

7 R. Bol and J. F. Groote. The meaning of negative premises in transition system specifications. *Journal of the ACM*, 43(5):863–914, 1996. doi:10.1145/234752.234756.

8 V. Castiglioni, D. Gebler, and S. Tini. Modal decomposition on nondeterministic probabilistic processes. In *Proc. CONCUR 2016*, volume 59 of *LIPIcs*, pages 36:1–36:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CONCUR.2016.36.

9 P. R. D'Argenio and M. D. Lee. Probabilistic transition system specification: Congruence and full abstraction of bisimulation. In *Proc. FOSSACS 2012*, volume 7213 of *LNCS*, pages 452–466. Springer, 2012. doi:10.1007/978-3-642-28729-9_30.

10 A. J. Dos Reis. *Compiler Construction Using Java, JavaCC, and Yacc*. Wiley-IEEE, 2012. doi:10.1002/9781118112762.

11 W. J. Fokkink and R. J. van Glabbeek. Ntyft/ntyxt rules reduce to ntree rules. *Information and Computation*, 126(1):1–10, 1996. doi:10.1006/inco.1996.0030.

12 W. J. Fokkink and R. J. van Glabbeek. Divide and congruence II: Delay and weak bisimilarity. In *Proc. LICS 2016*, pages 778–787. ACM/IEEE, 2016. doi:10.1145/2933575.2933590.

13 W. J. Fokkink, R. J. van Glabbeek, and P. de Wind. Compositionality of Hennessy-Milner logic by structural operational semantics. *Theoretical Computer Science*, 354(3):421–440, 2006. doi:10.1016/j.tcs.2005.11.035.

14 W. J. Fokkink, R. J. van Glabbeek, and P. de Wind. Divide and congruence: From decomposition of modal formulas to preservation of branching and η-bisimilarity. *Information and Computation*, 214:59–85, 2012. doi:10.1016/j.ic.2011.10.011.

15 R. J. van Glabbeek. Full abstraction in structural operational semantics (extended abstract). In *Proc. AMAST 1993*, Workshops in Computing, pages 75–82. Springer, 1993. doi:10.1007/978-1-4471-3227-1_7.

16 R. J. van Glabbeek. The linear time – branching time spectrum I: The semantics of concrete, sequential processes. In J. A. Bergstra, A. Ponse, and S. A. Smolka, editors, *Handbook of Process Algebra*, chapter 1, pages 3–99. Elsevier, 2001.

17 R. J. van Glabbeek. On cool congruence formats for weak bisimulations. *Theoretical Computer Science*, 412(28):3283–3302, 2011. doi:10.1016/j.tcs.2011.02.036.

18 J. F. Groote and F. Vaandrager. Structured operational semantics and bisimulation as a congruence. *Information and Computation*, 100(2):202–260, 1992. doi:10.1016/0890-5401(92)90013-6.

19 M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *Journal of the ACM*, 32(1):137–161, 1985. doi:10.1145/2455.2460.

20 K. V. Hindriks and M. B. van Riemsdijk. Satisfying maintenance goals. In *Proc. DALT 2007*, volume 4897 of *LNCS*, pages 86–103. Springer, 2007. doi:10.1007/978-3-540-77564-5_6.

21 K. G. Larsen. *Context-Dependent Bisimulation between Processes*. PhD thesis, University of Edinburgh, 1986.

22 L. Léonard and G. Leduc. A formal definition of time in LOTOS. *Formal Aspects of Computing*, 10(3):248–266, 1998. doi:10.1007/s001650050015.

23 M. R. Mousavi. Causality in the semantics of Esterel: Revisited. In *Proc. SOS 2009*, volume 18 of *EPTCS*, pages 32–45, 2009. doi:10.4204/EPTCS.18.3.

24 G. D. Plotkin. A structural approach to operational semantics. *Journal of Logic and Algebraic Programming*, 60-61:17–139, 2004. doi:10.1016/j.jlap.2004.05.001.

25 R. de Simone. Higher-level synchronising devices in Meije–SCCS. *Theoretical Computer Science*, 37(3):245–267, 1985. doi:10.1016/0304-3975(85)90093-3.

A Omitted proofs

A.1 Proof of Proposition 26

Proposition 26 states that for a structured proof (π, ρ), if a variable z occurs in the right-hand side of a premise in $top(\pi)$ or exactly once in s_π, then it is related to all other occurrences of z in π through $\rightarrow_z$. We start with some lemmas needed in the proof of Proposition 26.

▶ **Lemma 38.** *Each variable z occurring in a structured proof (π, ρ) has the form $\rho(x)$ for a prime variable $x \in var(\pi)$.*

Proof. With structural induction on $\rho(y)$ for any $y \in var(\pi)$ we show that each $z \in var(\rho(y))$ has the form $z = \rho(x)$ for a prime variable $x \in var(\pi)$.

- In case y is prime, $\rho(y) \in V$ by the first clause of Definition 17, so $z = \rho(y)$ and we are done.
- Suppose that y is composite. So $y \sim_\pi t$ for some $t \notin V$, and by the third clause of Definition 17, $\rho(y) = \rho(t)$. Hence z occurs in $\rho(t)$, and therefore in $\rho(y')$ for a variable y' occurring in t. Since $\rho(y')$ is a proper subterm of $\rho(y)$, by induction $z = \rho(x)$ for a prime variable $x \in var(\pi)$. ◀

▶ **Lemma 39.** *Let $\theta = \langle b, \iota, \eta\rangle$ and $\theta' = \langle b', \iota', \eta\rangle$ be two occurrences of a variable z in a structured proof (π, ρ) from a TSS P in univariate tytt format, with ι an occurrence of a subterm t in b, and ι' an occurrence of a subterm u in b', where $t \sim_\pi u$. Then $\theta \sim_z \theta'$.*

Proof. We apply induction on the derivation of $t \sim_\pi u$.

- The case that $t \sim_\pi u$ is obtained by the first clause of Definition 14 follows immediate from the definitions, in particular using the first clause of Definition 21, but only when ι and ι' are the indicated occurrences ι_t and $\iota_{t'}$ (or ι_u and $\iota_{u'}$) in Definition 21. We also need to show that if a subterm t occurs multiple times in π, the corresponding occurrences of z in the occurrences of t are related by $\sim_z$. Since t must contain variables, and the sets of variables in different rules $b \in B$ (and α) in a proof structure $\pi = (B, \alpha, \varphi)$ are pairwise disjoint, all occurrences of t lay in the same rule b. The second and third clause of Definition 21 together with the fact that P is in univariate tytt format guarantee that all induced occurrences of z are $\sim_z$-related.
- The case that $t \sim_\pi u$ is obtained by the second clause of Definition 14 is trivial.
- The case that $t \sim_\pi u$ is obtained by reflexivity, symmetry or transitivity is trivial too. ◀

▶ **Lemma 40.** *For each two occurrences θ and θ' of a variable z in a structured proof (π, ρ) from a TSS in univariate tytt format, we have $\theta \sim_z \theta'$.*

Proof. Let $\pi = (B, \alpha, \varphi)$. By Lemma 38, for each variable z occurring in (π, ρ) we can choose an occurrence of the form (b, ι, η) with $b \in B \cup \{\alpha\}$, ι an occurrence of a prime variable x in b, and η the occurrence of z in $\rho(x) = z$. Let (b', ι', η') be another occurrence of z in (π, ρ), with ι' an occurrence of a variable y' in b' and η' an occurrence of z in $\rho(y')$. With structural induction on $\rho(y')$ we show that $(b, \iota, \eta) \sim_z (b', \iota', \eta')$.

- Let y' be prime. Then $\rho(y') \in V$, so $\rho(y') = z = \rho(x)$. By the second clause of Definition 17, $x \sim_\pi y'$. The result now follows from Lemma 39.
- Let y' be composite. Then $y' \sim_\pi t$ for some $t \notin V$, so by the third clause of Definition 17, $\rho(y') = \rho(t)$. Hence the occurrence η' of z in $\rho(y')$ appears as an occurrence η'' of z in $\rho(y'')$ for a variable y'' occurring in t. Let b'' be the rule containing t, ι_t an occurrence of t in b'' and ι'' the appropriate occurrence of y'' within ι_t. Then $(b'', \iota'', \eta'') = \langle b'', \iota_t, \eta' \rangle$ is an occurrence of z in (π, ρ). Since $\rho(y'')$ is a proper subterm of $\rho(y')$, by induction $(b, \iota, \eta) \sim_z (b'', \iota'', \eta'')$. Furthermore, Lemma 39 yields $\langle b'', \iota_t, \eta' \rangle \sim_z \langle b', \iota', \eta' \rangle = (b', \iota', \eta')$. ◀

In the following observations, which are also needed in the proof of Proposition 26, (π, ρ) is a structured proof from a TSS P in univariate tytt format with $\pi = (B, \alpha, \varphi)$.

▶ **Observation 41.** *For each incoming occurrence θ' of z in (π, ρ) there is at most one outgoing occurrence θ of z with $\theta \to_z \theta'$. There is none iff θ' occurs in $top(\pi)$.*

▶ **Observation 42.** *For each outgoing occurrence $\theta' = (b', \iota', \eta')$ of z in (π, ρ) there is at most one incoming occurrence $\theta = (b, \iota, \eta)$ of z with $\theta \to_z \theta'$, where it must be the case that $b' = b \neq \alpha$.*

Now we are ready to present the proof of Proposition 26.

Proof. By Lemma 40, $\theta \sim_z \theta'$. By the definition of $\sim_z$, $\theta = \theta_0 \sim_z^1 \theta_1 \sim_z^1 \cdots \sim_z^1 \theta_k = \theta'$, where $\sim_z^1 = \to_z \cup \leftarrow_z \cup \leftrightsquigarrow_z$. Without loss of generality we assume that there are no repeated occurrences of z in this sequence. By induction on i we show that $\theta_i \to_z \theta_{i+1}$ for all $i = 0, \ldots, k-1$.

- Let $i = 0$, and $\theta_0 = (b, \iota, \eta)$ with ι the only occurrence of x in s_π. By Definition 21 there is no θ_1 with $\theta_1 \to_z \theta_0$ or – using that ι is the only occurrence of x in α – $\theta_0 \leftrightsquigarrow_z \theta_1$.
- Let $i = 0$ and $\theta_0 = (b, \iota, \eta)$ with ι occurring in $top(\pi)$. Since θ_0 is an incoming occurrence, by Definition 21 there is no θ_1 with $\theta_0 \leftrightsquigarrow_z \theta_1$. By Observation 41 there is no θ_1 with $\theta_1 \to_z \theta_0$.

- Let $i > 0$ and θ_i be an incoming occurrence. By Definition 21 there is no θ_{i+1} with $\theta_i \leftrightsquigarrow_z \theta_{i+1}$. By Observation 41 and our convention that $\theta_{i+1} \neq \theta_{i-1}$, we cannot have $\theta_i \leftarrow_z \theta_{i+1}$.
- Let $i > 0$ and θ_i be an outgoing occurrence. By Observation 23 θ_{i-1} must be a incoming occurrence, occurring in the same rule. Hence by Definition 21 there is no θ_{i+1} with $\theta_i \leftrightsquigarrow_z \theta_{i+1}$. By Observation 42 and our convention that $\theta_{i+1} \neq \theta_{i-1}$, we cannot have $\theta_i \leftarrow_z \theta_{i+1}$. ◀

A.2 Proof of Proposition 30

We now present the proof of Proposition 30, which states that if all rules used in a proof structure π have bounded lookahead, then there is no infinite chain $x_0 \prec_\pi x_1 \prec_\pi x_2 \prec_\pi \cdots$.

Proof. With structural induction on proof structures $\pi = (B, \alpha, \varphi)$, seen as well-founded trees. The case that $\alpha \in \mathit{top}(\pi)$ is trivial. So assume $\alpha \notin \mathit{top}(\pi)$.

Let $b_0 \in B$ be the unique rule with $\varphi(b_0) = \alpha$. For each premise β of b_0, let $B_\beta \subseteq B$ be the collection of rules that are above β, and let $\pi_\beta = (B_\beta, \beta, \varphi{\restriction}B_\beta)$. The structured proofs π_β are subproofs of π and by induction do not contain infinite chains as above.

Suppose an infinite chain $x_0 \prec_\pi x_1 \prec_\pi \cdots$ occurred in π. With the possible exception of x_0, which could lay in α, this entire chain can be divided up in connected segments, each of which lays entirely in one of the π_βs. Each segment has at least two variables in it, and two adjacent segments – laying in π_β and π_γ, respectively – overlap in exactly one variable, which must occur in the right-hand side of β as well as in the left-hand side of γ. Here we use that the sets $\mathit{var}(b)$ for $b \in B$ are pairwise disjoint. Using the induction hypothesis, all these segments must be finite. Hence, there must be infinitely many. Restricting the sequence $x_0 \prec_\pi x_1 \prec_\pi x_2 \prec_\pi \cdots$ to those variables that lay in two adjacent segments yields an infinite forward chain of variables in the dependency graph of b_0, contradicting the supposed absence of unbounded lookahead in the rules of B. ◀

A.3 Proof of Proposition 36

Proposition 36 states that given a TSS in partial trace format, the modal decomposition method turns each formula from $\mathbb{O}_T$ into conjunctions of formulas from $\mathbb{O}_T^{\equiv}$.

The following lemma is needed in the proof of Proposition 36. There it is only used in case $x \notin \mathit{var}(t)$, but within the proof of the lemma we also need the case that x has one, Λ-liquid occurrence in t.

▶ **Lemma 43.** *Let P be a TSS in partial trace format, where its rules are Λ-partial trace safe. Let $\mathcal{R}$ denote the set of P-general ruloids. For each term t, $\varphi \in \mathbb{O}_T$, $\psi \in t_{\mathcal{R}}^{-1}(\varphi)$, and variable x that occurs at most once in t, at a Λ-liquid position, we have $\psi(x) \in \mathbb{O}_T^{\equiv}$.*

Proof. We apply induction on the structure of $\varphi \in \mathbb{O}_T$. Let $\psi \in t_{\mathcal{R}}^{-1}(\varphi)$. The two possible syntactic forms of φ in the BNF grammar of $\mathbb{O}_T$ are considered. In case $\varphi = \top$, i.e., $\varphi = \bigwedge_{i \in \emptyset} \varphi_i$, by the second clause of Definition 6, $\psi(x) = \top \in \mathbb{O}_T^{\equiv}$, and we are done. In case $\varphi = \langle a \rangle \varphi'$, by the first clause of Definition 6, $\psi(x) = \bigwedge_{(x \xrightarrow{b} y) \in H} \langle b \rangle \psi(y)[\wedge \chi(x)]$, for some P-general ruloid $r = \frac{H}{t \xrightarrow{a} u}$ and $\chi \in u_{\mathcal{R}}^{-1}(\varphi')$, where the square brackets around the conjunct $\chi(x)$ indicate that it is optional: it is present only if $x \in \mathit{var}(u)$. By Theorems 31 and 35, r is Λ-partial trace safe. Since by assumption x is Λ-floating in r, by Definition 2, x has at most one occurrence in total in the left-hand sides of H and in u; this occurrence must be at a Λ-liquid position. We apply a nested induction on the lookahead of x in H (formally

defined in [13, page 19 – aliased 434]). Suppose first that x occurs in the left-hand side of H, say $x \xrightarrow{c} z$. Since r is tytt, z does not occur in $var(t)$. So by induction on the lookahead of z, $\psi(z) \in \mathbb{O}_T^{\equiv}$. Hence $\psi(x) = \langle c \rangle \psi(z) \in \mathbb{O}_T^{\equiv}$. Suppose now that x does not occur in the left-hand sides of H. Since x occurs at most once in u, at a Λ-liquid position, by induction on the structure of φ', $\chi(x) \in \mathbb{O}_T^{\equiv}$. Hence either $\psi(x) = \chi(x) \in \mathbb{O}_T^{\equiv}$ or $\psi(x) = \top \in \mathbb{O}_T^{\equiv}$. ◀

Now we present the proof of Proposition 36.

Proof. All rules in P are Λ-partial trace safe, for some Λ. We apply induction on the structure of $\varphi \in \mathbb{O}_T$. Let $\psi \in t_{\mathcal{R}}^{-1}(\varphi)$. We consider the two possible syntactic forms of φ in the BNF grammar of $\mathbb{O}_T$. In case $\varphi = \top$, by the second clause of Definition 6, $\psi(x) = \top$, and we are done. In case $\varphi = \langle a \rangle \varphi'$, by the first clause of Definition 6, $\psi(x) = \bigwedge_{(x \xrightarrow{b} y) \in H} \langle b \rangle \psi(y)[\wedge \chi(x)]$, for some P-general ruloid $r = \frac{H}{t \xrightarrow{a} u}$ and $\chi \in u_{\mathcal{R}}^{-1}(\varphi')$, where the conjunct $\chi(x)$ is present only if $x \in var(u)$. By Theorems 31 and 35, r is Λ-partial trace safe. Since r is tytt, for each $(x \xrightarrow{b} y) \in H$, y does not occur in $var(t)$. So by Lemma 43, $\psi(y) \in \mathbb{O}_T^{\equiv}$ for each $(x \xrightarrow{b} y) \in H$. Moreover, by induction, $\chi(x) \equiv \bigwedge_{i \in I} \chi_i$ with $\chi_i \in \mathbb{O}_T^{\equiv}$ for all $i \in I$. Thus $\psi(x)$ is also of this required form. ◀