Lucas Alba, S.; Meseguer, J.; Gutiérrez Gil, R. (2020). The 2D Dependency Pair Framework for Conditional Rewrite Systems¿Part II: Advanced Processors and Implementation Techniques. Journal of Automated Reasoning. 64(8):1611-1662. https://doi.org/10.1007/s10817-020-09542-3

jar manuscript No.
(will be inserted by the editor)

The 2D Dependency Pair Framework for conditional rewrite systems. Part II: Advanced processors and implementation techniques

Salvador Lucas · José Meseguer · Raúl Gutiérrez

Abstract Proving termination of programs in 'real-life' rewriting-based languages like CafeOBJ, Haskell, Maude, etc., is an important subject of research. To advance this goal, faithfully capturing the impact in the termination behavior of the main language features (e.g., conditions in program rules) is essential. In Part I of this work, we have introduced a *2D Dependency Pair Framework* for automatically proving termination properties of *Conditional Term Rewriting Systems*. Our framework relies on the notion of *processor* as the main practical device to deal with proofs of termination properties of conditional rewrite systems. Processors are used to decompose and simplify the proofs in a *divide and conquer* approach. With the basic proof framework defined in Part I, here we introduce new processors to further improve the ability of the 2D Dependency Pair Framework to deal with proofs of termination properties of conditional rewrite systems. We also discuss relevant implementation techniques to use such processors in practice.

Keywords Conditional term rewriting · dependency pairs · program analysis · operational termination

1 Introduction

The operational semantics of *Conditional Term Rewriting Systems* (CTRSs) admits a simple description as *deduction* in a logic with binary predicates $\rightarrow$ and $\rightarrow^*$ rep-

Partially supported by the EU (FEDER) and projects RTI2018-094403-B-C32, PROMETEO/2019/098, SP20180225. José Meseguer was supported by grants NSF CNS 13-19109 and NRL N00173-17-1-G002. Salvador Lucas' research was partly developed during a sabbatical year at the UIUC.

Salvador Lucas
Valencian Research Institute for Artificial Intelligence (VRAIN), Universitat Politècnica de València, Spain

José Meseguer
CS Dept. at the University of Illinois at Urbana-Champaign

Raúl Gutiérrez
Valencian Research Institute for Artificial Intelligence (VRAIN), Universitat Politècnica de València, Spain

$$x - \mathsf{0} \to x \tag{1}$$
$$\mathsf{0} - y \to \mathsf{0} \tag{2}$$
$$\mathsf{s}(x) - \mathsf{s}(y) \to x - y \tag{3}$$
$$\mathsf{greater}(s(x), s(y)) \to \mathsf{greater}(x, y) \tag{4}$$
$$\mathsf{greater}(s(x), \mathsf{0}) \to \mathsf{true} \tag{5}$$
$$\mathsf{leq}(s(x), s(y)) \to \mathsf{leq}(x, y) \tag{6}$$
$$\mathsf{leq}(\mathsf{0}, x) \to \mathsf{true} \tag{7}$$
$$\mathsf{div}(x, y) \to \mathsf{pair}(\mathsf{0}, x) \Leftarrow \mathsf{greater}(y, x) \to \mathsf{true} \tag{8}$$
$$\mathsf{div}(x, y) \to \mathsf{pair}(\mathsf{s}(q), r) \Leftarrow \mathsf{leq}(y, x) \to \mathsf{true}, \mathsf{div}(x - y, y) \to \mathsf{pair}(q, r) \tag{9}$$

Fig. 1 A Conditional Term Rewriting System for integer division

resenting, respectively, *a single* rewriting step (with $\to$) and *zero-or-more* rewriting steps (with $\to^*$). Deductions proceed according to a well-known *inference system* (see [21] and Figure 2). Consider the CTRS $\mathcal{R}$ in Figure 1 from [30, Example 9]. The intended use of $\mathcal{R}$ concerns the *integer* division of *natural* numbers using the standard division algorithm. Natural numbers are represented in Peano's notation (with 0 represented as $\mathsf{0}$, 1 as $\mathsf{s}(\mathsf{0})$, 2 as $\mathsf{s}(\mathsf{s}(\mathsf{0}))$, and so on). A call to $\mathsf{div}(m, n)$ with m and n representing natural numbers would return, *in a single step of conditional rewriting*, the quotient Q and remainder R as an expression $\mathsf{pair}(Q, R)$ with Q and R again represented in Peano's notation. Note the use of the *conditional* part in rules (8) and (9) to *check* conditions on the arguments x and y before providing a final outcome. For instance, rule (8) returns a quotient $Q = \mathsf{0}$ and remainder $R = x$ (as $\mathsf{pair}(\mathsf{0}, x)$) if the divisor y *exceeds* the divided number x (i.e., $\mathsf{greater}(y, x)$ rewrites into true). Rule (9) returns $Q = \mathsf{s}(q)$ and $R = r$ by not only checking that y is lesser than or equal to x (i.e., $\mathsf{leq}(y, x)$ rewrites into true) but also performing the central part of the division algorithm as an *auxiliary computation*: a recursive call to div on an appropriately decreased first argument if $n \neq \mathsf{0}$. The implementation of the integer division provided by $\mathcal{R}$ is sound, in the sense that for all $m, n \in \mathbb{N}$ with $m \neq 0$, if $\mathsf{div}(m, n)$ rewrites into $\mathsf{pair}(Q, R)$, then Q and R are actually the quotient and remainder of the integer division of m by n. For a practical use of $\mathcal{R}$, though, some questions arise: Is every rewrite sequence issued with $\mathcal{R}$ finite? Are there any other problems regarding the aforementioned auxiliary computations performed 'in the conditional part' of the rules?

In this setting, some termination properties are defined in [24, Sect. 3]: given a CTRS $\mathcal{R}$, a term t is said to be *terminating* iff[1] there is no infinite rewrite sequence $t = t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots$ (a *horizontal dimension* of nonterminating behaviors in CTRSs); t is called *V-terminating* iff (roughly speaking) no attempt to *prove* a *single* rewriting step $t \to_{\mathcal{R}} u$ for some term u leads to infinitely many attempts to prove other one-step rewritings as part of the original proof (a *vertical dimension*). This vertical dimension is due to the use in CTRSs of *conditional rules* $\ell \to r \Leftarrow c$ where an attempt to perform a rewriting step $s \to_{\mathcal{R}} t$ using such a rule may launch other computations involving the evaluation of the conditions $s_i \to t_i$ occurring in the conditional part c of the rule. A CTRS $\mathcal{R}$ is (V-)terminating iff all terms are (V-

[1] In the following, *iff* abbreviates *if and only if*.

)erminating. Finally, $\mathcal{R}$ is called *operationally terminating* if it is both terminating and V-terminating.

Example 1 The evaluation of $\mathsf{div}(0,0)$ using $\mathcal{R}$ in Figure 1 starts with an attempt to apply the conditional rules (8) and (9). Rule (8) cannot be applied because the reachability test $\mathsf{greater}(0,0) \to^* \mathsf{true}$ introduced by the conditional part of the rule *fails*. Regarding rule (9), the first test $\mathsf{leq}(0,0) \to^* \mathsf{true}$ succeeds, but the attempt to check the second one, i.e., $\mathsf{div}(0-0,0) \to^* \mathsf{pair}(s,t)$ for some terms s and t, *never ends*: after a first reduction step on $0-0$ in $\mathsf{div}(0-0,0)$, we obtain $\mathsf{div}(0,0)$ which is exactly our initial expression. Thus, we have infinitely many attempts to prove that some *one-step* rewriting is possible on such an initial expression without actually issuing any one! This is what (non) V-termination captures. Indeed, $\mathcal{R}$ is *not* V-terminating and therefore it is not operationally terminating. We will show how to prove this in practice (see Example 20).[2]

The *twofold* origin of infinite computations sketched above is captured by *two* CTRSs: $\mathsf{DP}_H(\mathcal{R})$ (the *horizontal* dependency pairs of $\mathcal{R}$) and $\mathsf{DP}_V(\mathcal{R})$ (the *vertical* dependency pairs of $\mathcal{R}$) [24]. $\mathsf{DP}_H(\mathcal{R})$ consists of the rules $\ell^\sharp \to v^\sharp \Leftarrow c$ that are obtained from each rule $\ell \to r \Leftarrow c$ in $\mathcal{R}$ by choosing a *defined subterm* v of r, i.e., v is a subterm of r such that the root symbol of v is a *defined*[3] symbol. The notation $t^\sharp$ represents the *marking* of the *root symbol* f of a term t to distinguish it from f. We often *capitalize* f into F rather than writing $f^\sharp$. Instead, $\mathsf{DP}_V(\mathcal{R})$ consists of rules $\ell^\sharp \to v^\sharp \Leftarrow s_1 \to t_1, \ldots, s_{i-1} \to t_{i-1}$ for each rule $\ell \to r \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n$ in $\mathcal{R}$ and $1 \le i \le n$ such that v is a defined subterm of s_i.

Example 2 For $\mathcal{R}$ in Figure 1, we have:

$$\mathsf{DP}_H(\mathcal{R}):\quad \mathsf{s}(x) -^\sharp \mathsf{s}(y) \to x -^\sharp y \tag{10}$$
$$\mathsf{GREATER}(s(x),s(y)) \to \mathsf{GREATER}(x,y) \tag{11}$$
$$\mathsf{LEQ}(s(x),s(y)) \to \mathsf{LEQ}(x,y) \tag{12}$$

$$\mathsf{DP}_V(\mathcal{R}):\quad \mathsf{DIV}(x,y) \to \mathsf{GREATER}(y,x) \tag{13}$$
$$\mathsf{DIV}(x,y) \to \mathsf{LEQ}(y,x) \tag{14}$$
$$\mathsf{DIV}(x,y) \to \mathsf{DIV}(x-y,y) \Leftarrow \mathsf{leq}(y,x) \to \mathsf{true} \tag{15}$$
$$\mathsf{DIV}(x,y) \to x -^\sharp y \Leftarrow \mathsf{leq}(y,x) \to \mathsf{true} \tag{16}$$

The 2D Dependency Pairs $\mathsf{DP}_H(\mathcal{R})$ and $\mathsf{DP}_V(\mathcal{R})$ of a CTRS $\mathcal{R}$ allow the analysis of termination, V-termination, and operational termination of CTRSs as the absence of *infinite chains* of rules coming from $\mathsf{DP}_H(\mathcal{R})$ or $\mathsf{DP}_V(\mathcal{R})$ [24, Section 7]. The *2D DP Framework* developed in Part I [27] provides a basis to *mechanize* proofs of *termination*, *V-termination*, and *operational termination* of CTRSs. The main idea is that, by using the same kind of structure $(\mathcal{P},\mathcal{Q},\mathcal{R},f)$, called a *CTRS problem* (where $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ are CTRSs and f is a *flag variable*), we are able to prove or disprove the aforementioned properties for $\mathcal{R}$ by appropriately specifying

[2] Schernhammer and Gramlich considered this implementation of integer division as an example of a 'careless' definition of a program where detecting operational nontermination "points to a flaw in the specification of R allowing division by zero" [30, page 675]. However, they could not provide an automatic proof of operational nontermination.

[3] A k-ary symbol f is *defined* in a CTRS $\mathcal{R}$ if there is a rule $f(\ell_1,\ldots,\ell_k) \to r \Leftarrow c$ in $\mathcal{R}$.

$\mathcal{P}$, $\mathcal{Q}$, and f. In this setting, a central notion is that of *processor*, which *transforms* a given *CTRS problem* into a *set* of *simpler* problems which can then be handled independently. This *divide and conquer* approach is paramount in the (2D) DP Framework. In Part I [27], *eight* processors were introduced: (i) the *Strongly Connected Components (SCC) processor* P_{SCC} permits the use of graph techniques to *decompose* termination problems; (ii) the *subterm* processor $\mathsf{P}_{\triangleright}$ adapts Hirokawa and Middeldorp's *subterm criterion* for TRSs [15]; (iii) the *Reduction Triple Processor* P_{RT} relies on well-founded relations to *simplify* termination problems; (iv,v) processors P_{IR} and P_{RIR} remove rules that cannot be used due to unsatisfiable conditions; (vi,vii) processors P_{NR} and P_{NQ} extend and generalize the narrowing processor for TRSs in [12]; and finally (viii) processor P_{Inf} is used to specifically *disprove* operational termination of CTRSs.

After a preliminary Section 2 and a brief introduction to the 2D Dependency Framework in Section 3, in Section 4 we extend the 2D DP Framework with several new powerful processors beyond the eight presented in [27] that greatly increase the 2D DP Framework's effectiveness:

1. Processor P_{NC} transforms conditional rules $\ell \to r \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n$ by unfolding the left-hand sides s_i of the conditions by means of narrowing.
2. Processor P_{SUC} *simplifies the conditional part* of the rules by removing conditions $s_i \to t_i$ such that s_i and t_i *unify*.
3. Processor P_{SUNC} combines P_{NC} and P_{SUC}.
4. We introduce a *semantic version* of P_{RT} which is amenable for implementation: the application of P_{RT} to a CTRS problem is translated into a many-sorted first-order *satisfiability* problem which can be automatically treated by means of tools like AGES [14] or Mace4 [28].
5. We define a new version P_{RTN} of the Reduction Triple processor P_{RT} in [27] where comparisons are restricted to *usable rules* only. We also provide a semantic version of this new processor.

Our benchmarks show that the 2D DP Framework augmented with these five new processors outperforms all existing tools for proving operational termination of CTRSs (Section 5). Section 6 concludes.

This paper contains extended and completely revised versions of processors in [26, Section 6] according to the 2D DP Framework [27]. Processor P_{RTN} (Section 4.4) and the semantic version of P_{RT} (Section 4.3) are entirely new. P_{NC} has been redefined with respect to its first formulation in [26] to fix a bug.

2 Preliminaries

The material in this section follows [29]. A binary relation R on a set A is *terminating* (or *well-founded*) if there is no infinite sequence $a_1 \,\mathsf{R}\, a_2 \,\mathsf{R}\, a_3 \cdots$. For relations $\mathsf{R}, \mathsf{S} \subseteq A \times A$, we let $\mathsf{R} \circ \mathsf{S} = \{(a, c) \in A \times A \mid \exists b \in A, a \,\mathsf{R}\, b \wedge b \,\mathsf{S}\, c\}$.

Throughout the paper $\mathcal{X}$ denotes a countable set of variables and $\mathcal{F}$ denotes a signature, i.e., a set of function symbols $\{\mathtt{f}, \mathtt{g}, \ldots\}$, each having a fixed arity given by a mapping $ar : \mathcal{F} \to \mathbb{N}$. The set of terms built from $\mathcal{F}$ and $\mathcal{X}$ is $\mathcal{T}(\mathcal{F}, \mathcal{X})$. $\mathcal{V}ar(t)$ is the set of variables occurring in term t. A term t is *ground* if it contains no variable (i.e., $\mathcal{V}ar(t) = \emptyset$). A term is said to be *linear* if it does not contain multiple occurrences of the same variable.

Terms are viewed as labeled trees in the usual way. Positions $p, q, \ldots$ are sequences of positive natural numbers used to address subterms of t. We denote the empty sequence by Λ. The set of positions of a term t is $\mathcal{P}os(t)$. Positions of nonvariable symbols in t are denoted as $\mathcal{P}os_{\mathcal{F}}(t)$. The subterm of t at position p is denoted as $t|_p$, and $t[s]_p$ is the term t with the subterm at position p replaced by s. We write $s \trianglerighteq t$, read *t is a subterm of s*, if $t = s|_p$ for some $p \in \mathcal{P}os(s)$ and $s \triangleright t$ if $s \trianglerighteq t$ and $s \neq t$. We write $s \not\triangleright t$ if $s \triangleright t$ does not hold. The symbol labeling the root of t is denoted $root(t)$.

A substitution is a mapping $\sigma : \mathcal{X} \to \mathcal{T}(\mathcal{F}, \mathcal{X})$. The 'identity' substitution $x \mapsto x$ for all $x \in \mathcal{X}$ is denoted ε. The set $\mathcal{D}om(\sigma) = \{x \in \mathcal{X} \mid \sigma(x) \neq x\}$ is called the *domain* of σ. We do *not* impose finiteness of the domain of substitutions σ. This is usual practice in the dependency pair approach, where a single substitution is used to instantiate an infinite number of variables coming from renamed versions of the dependency pairs (see below). When substitutions with *finite* domain are assumed, we explicitly call them *finite* substitutions. A *renaming* is a bijective substitution ρ such that $\rho(x) \in \mathcal{X}$ for all $x \in \mathcal{X}$. A finite substitution σ such that $\sigma(s) = \sigma(t)$ for two terms $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ is called a *unifier* of s and t; we also say that s and t *unify* (with substitution σ). If two terms s and t unify, then there is a unique (up to renaming of variables) *most general unifier* (*mgu*) σ such that for every other unifier σ' we have $\sigma \leq \sigma'$, i.e., there is a substitution θ such that $\theta \circ \sigma = \sigma'$. In the following we often write $s =^?_\sigma t$ if s and t unify with *mgu* σ. A substitution σ *unifies* (or is a *unifier* of) a set of equations E (or, better, unification problems $s =^? t$, following the notation in [3, page 71]) iff for all $s =^? t \in E$, $\sigma(s) = \sigma(t)$.

2.1 Conditional rewriting and operational termination

An (*oriented*) CTRS is a pair $\mathcal{R} = (\mathcal{F}, R)$ where $\mathcal{F}$ is a signature and R a set of rules $\ell \to r \Leftarrow c$, with c a sequence $s_1 \to t_1, \cdots, s_n \to t_n$ for some $n \geq 0$ and terms $\ell, r, s_1, \ldots, t_n$ such that $\ell \notin \mathcal{X}$. As usual, ℓ and r are called the *left-* and *right-hand sides* of the rule (*lhs* and *rhs*, respectively), and c is the *conditional part* of the rule. We often write $s_i \to t_i \in c$ to say that $s_i \to t_i$ is the i-th condition in c; we also write $s \to t \in c$ to refer an arbitrary condition in c. *Labeled* rules are written $\alpha : \ell \to r \Leftarrow c$, where α is a *label*.

Conditional rules $\ell \to r \Leftarrow c$ are classified according to the distribution of variables among ℓ, r, and c, as follows: *type 1*, if $\mathcal{V}ar(r) \cup \mathcal{V}ar(c) \subseteq \mathcal{V}ar(\ell)$; *type 2*, if $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(\ell)$; *type 3*, if $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(\ell) \cup \mathcal{V}ar(c)$; and *type 4*, if no restriction is given. A rule of type n is often called an n-rule. An n-CTRS contains only n-rules. A TRS is a 1-CTRS whose rules have no conditional part; we display them $\ell \to r$. A 3-CTRS $\mathcal{R}$ is called *deterministic* if for each rule $\ell \to r \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n$ in $\mathcal{R}$ and each $1 \leq i \leq n$, we have $\mathcal{V}ar(s_i) \subseteq \mathcal{V}ar(l) \cup \bigcup_{j=1}^{i-1} \mathcal{V}ar(t_j)$.

Given an atomic formula A of the form $s \to t$ or $s \to^* t$, $pred(A)$ refers to its *predicate* symbol $\to$ or $\to^*$, respectively, and $left(A)$ refers to s. Given a CTRS $\mathcal{R}$, a finite proof tree T (for the inference system in Figure 2 is either:[4] (i) an *open goal* G of the form $s \to t$ or $s \to^* t$ for terms s, t; then, we denote $root(T) = G$;

[4] Note that the inference rules are *schematic* in the sense that each inference rule $\frac{B_1 \cdots B_n}{A}$ can be used under any *instance* $\frac{\sigma(B_1) \cdots \sigma(B_n)}{\sigma(A)}$ of the rule by a substitution σ.

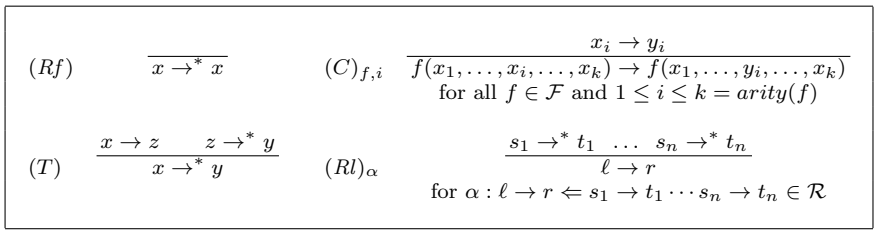


Fig. 2 Inference rules for conditional rewriting with a CTRS $\mathcal{R}$ with signature $\mathcal{F}$

otherwise, (ii) is a *derivation tree* with root G, denoted as

$$\frac{T_1 \quad \cdots \quad T_n}{G}(\rho) \tag{17}$$

where G is as above, T_1,...,T_n are finite proof trees (for $n \ge 0$), and $\rho : \frac{B_1 \cdots B_n}{A}$ is an inference rule such that $G = \sigma(A)$ and $root(T_i) = \sigma(B_i)$ for some substitution σ and $1 \le i \le n$. We write $root(T) = G$. A finite proof tree T is *closed* if it contains no open goals. A finite proof tree T is a *proper prefix* of a finite proof tree T' if there are one or more open goals $G_1, \ldots, G_n$ in T such that T' is obtained from T by replacing each G_i by a derivation tree T_i with root G_i. We denote this as $T \subset T'$. An *infinite proof tree* T is a sequence $T = \{T_i\}_{i \in \mathbb{N}}$ such that for all i, $T_i \subset T_{i+1}$. We let $root(T) = root(T_0)$.

A finite proof tree T is *well-formed* if it is either an open goal, or a closed proof tree, or a derivation tree like (17) where there is i, $1 \le i \le n$, such that $T_1, \ldots, T_{i-1}$ are closed, T_i is a well-formed but not closed finite proof tree, and $T_{i+1}, \ldots, T_n$ are open goals. An infinite proof tree is *well-formed* if it is an increasing chain of well-formed finite proof trees. A term t is *operationally terminating* if there is no infinite well-formed proof tree for $t \to^* u$, where u is an arbitrary term. A CTRS is operationally terminating if every term is operationally terminating.[5]

We write $s \to_{\mathcal{R}} t$ (resp. $s \to^*_{\mathcal{R}} t$) iff there is a well-formed closed proof tree for $s \to t$ (resp. $s \to^* t$). We often drop $\mathcal{R}$ from $\to_{\mathcal{R}}$ or $\to^*_{\mathcal{R}}$ if no confusion arises. Note that $s \to_{\mathcal{R}} t$ iff there is $p \in \mathcal{P}os(s)$, $\ell \to r \Leftarrow c \in \mathcal{R}$ and a substitution σ such that $\sigma(u) \to^*_{\mathcal{R}} \sigma(v)$ for all $u \to v \in c$, $s|_p = \sigma(\ell)$ and $t = s[\sigma(r)]_p$. We can make this explicit by writing $s \xrightarrow{p}_{\mathcal{R}} t$. We also write $s \xrightarrow{>p}_{\mathcal{R}} t$ if there is $q > p$ such that $s \xrightarrow{q}_{\mathcal{R}} t$. It is easy to prove that $s \to^*_{\mathcal{R}} t$ holds if and only if there is a sequence $s_1, \ldots, s_n$ of terms for some $n \ge 1$ such that $s = s_1$, $t = s_n$ and for all i, $1 \le i < n$, $s_i \to_{\mathcal{R}} s_{i+1}$; in particular, we write $s \xrightarrow{>\Lambda}{}^* t$ iff $s \to^* t$ and for all $i \ge 0$, $s_i \xrightarrow{>\Lambda} s_{i+1}$ holds. Given CTRSs $\mathcal{R}$ and $\mathcal{S}$, and terms s, t, we write $s \xrightarrow{\Lambda}_{\mathcal{S},\mathcal{R}} t$ if there is $\ell \to r \Leftarrow c \in \mathcal{S}$ and a substitution σ such that $s = \sigma(\ell)$, $t = \sigma(r)$ and $\sigma(u) \to^*_{\mathcal{R}} \sigma(v)$ for all $u \to v \in c$. That is, the one-step rewrite $s \xrightarrow{\Lambda} t$ uses a rule in $\mathcal{S}$ but the rule's condition c is evaluated with rules from $\mathcal{R}$. Also, we write $s \to_{\mathcal{S},\mathcal{R}} t$ iff there is a subterm s' of s at position $p \in \mathcal{P}os(s)$, i.e., $s' = s|_p$, such that $s' \xrightarrow{\Lambda}_{\mathcal{S},\mathcal{R}} t'$ and $t = s[t']_p$. Note that $\to_{\mathcal{R}} = \to_{\mathcal{R},\mathcal{R}}$.

[5] Our definition of operational termination depends on the (reasonable, but discretional) use of well-formed proof trees; see [18, Section 3.1] for a discussion about the impact of this decision in the analysis of the termination behavior of computational systems.

Let $\mathcal{R}$ be a CTRS, $\alpha : \ell \to r \Leftarrow c \in \mathcal{R}$ and σ be a substitution terminating over $\mathcal{V}ar(\ell)$ (i.e., for all $x \in \mathcal{V}ar(\ell)$, $\sigma(x)$ is terminating w.r.t. $\mathcal{R}$). We say that α *preserves termination* of σ (w.r.t. $\mathcal{R}$) iff σ is terminating over $\mathcal{V}ar(r)$ whenever $\sigma(s) \to_{\mathcal{R}}^* \sigma(t)$ for all $s \to t \in c$. We say that $\mathcal{R}$ *preserves terminating substitutions* if for all substitutions σ and rules $\alpha : \ell \to r \Leftarrow c \in \mathcal{R}$, if σ is terminating over $\mathcal{V}ar(\ell)$, then α preserves termination of σ. 2-CTRSs preserve terminating substitutions.

Let $\mathcal{R}$ be a CTRS and $\alpha : \ell \to r \Leftarrow c$ be a rule. We say that (the conditional part of) α is $\mathcal{R}$-*feasible* if there is a substitution σ such that for all $s \to t \in c$, $\sigma(s) \to_{\mathcal{R}}^* \sigma(t)$. Otherwise, it is called $\mathcal{R}$-*infeasible*. In the following, we often assume a CTRS $\mathcal{R}$ partitioned as $\mathcal{R} = \mathcal{R}_F \uplus \mathcal{R}_I$, with $\mathcal{R}_I$ a (possibly empty) set of $\mathcal{R}$-*infeasible* rules of $\mathcal{R}$. Since $\mathcal{R}$-infeasibility is, in general, undecidable, we just assume that rules in $\mathcal{R}_I$ are really (i.e., proved to be) infeasible ([20,31,32] develop some specific criteria for infeasibility). Some rules in $\mathcal{R}_F$ may also be infeasible, though.

3 The (Open) 2D DP Framework for CTRSs

To make the paper both reasonably self-contained and easier to read, this section summarizes some key ideas from Part I; missing details can be found in [27]. Besides $\mathsf{DP}_H(\mathcal{R})$ and $\mathsf{DP}_V(\mathcal{R})$, we also need the *connecting* CTRS $\mathsf{DP}_{VH}(\mathcal{R})$ which is the subset of rules in $\mathsf{DP}_H(\mathcal{R})$ defining symbols also occurring in the topmost position of the rhs's of rules in $\mathsf{DP}_V(\mathcal{R})$ [24, Definition 59].

Example 3 For $\mathsf{DP}_H(\mathcal{R})$ and $\mathsf{DP}_V(\mathcal{R})$ in Example 2, we have $\mathsf{DP}_{VH}(\mathcal{R}) = \mathsf{DP}_H(\mathcal{R})$. For instance, the root symbol $\mathsf{GREATER}$ in the *rhs* of (13) in $\mathsf{DP}_V(\mathcal{R})$ is defined by rule (11) in $\mathsf{DP}_H(\mathcal{R})$. Hence, (11) belongs to $\mathsf{DP}_{VH}(\mathcal{R})$.

Termination, V-termination, and operational termination of CTRSs are investigated as the absence of (combinations of) the so-called *O-chains*.[6]

Definition 1 [24, Definition 71] Let $\mathcal{P}, \mathcal{Q}, \mathcal{R}$ be CTRSs. A $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain is a finite or infinite sequence of (renamed) rules $u_i \to v_i \Leftarrow c_i \in \mathcal{P}$ together with a substitution σ satisfying that, for all $i \geq 1$,

1. for all $s \to t \in c_i$, $\sigma(s) \to_{\mathcal{R}}^* \sigma(t)$ and
2. $\sigma(v_i)(\to_{\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}})^* \sigma(u_{i+1})$.

A $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain is called *minimal* if for all $i \geq 1$, whenever

$$\sigma(v_i) = w_{i1}(\to_{\mathcal{R}}^* \circ \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}})\, w_{i2}(\to_{\mathcal{R}}^* \circ \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}}) \cdots (\to_{\mathcal{R}}^* \circ \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}})\, w_{im_i} \to_{\mathcal{R}}^* \sigma(u_{i+1}),$$

in the chain, then for all j, $1 \leq j \leq m_i$, w_{ij} is $\mathcal{R}$-operationally terminating.

Provided that $\mathcal{P}_I$, $\mathcal{Q}_I$, and $\mathcal{R}_I$ consist of $\mathcal{R}$-infeasible rules only, we can replace $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ by $\mathcal{P}_F$, $\mathcal{Q}_F$, and $\mathcal{R}_F$ in Definition 1 without losing $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chains. Removing infeasible rules from $\mathcal{R}$ may change *nonminimality* of $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chains due to the lack of preservation of operational termination of $\mathcal{R}$ under addition of rules, even if such rules are infeasible (see [27, Example 12]).

[6] In [24], three notions of *chain* of dependency pairs, namely, H-, V-, and O-chains, were introduced and applied to prove *termination*, *V-termination*, and *operational termination*, respectively. H-chains, though, use dependency pairs that we do not consider here.

In the following, $\mathsf{F} = \{\mathsf{a}, \mathsf{m}\}$ is a signature of *flag constants* referring to arbitrary (resp. minimal) $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chains. Clearly, arbitrary O-chains *include* minimal ones. A *CTRS problem* τ is a tuple $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathsf{a})$ or $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathsf{m})$, where $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ are CTRSs [27, Definition 14]. We speak of τ-chains as $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chains (minimal if indicated in τ with m). We say that τ is *finite* if there is no infinite τ-chain; and *infinite* otherwise.

Remark 1 In the DP Framework for TRSs, DP problems are explicitly required to be labeled as either minimal or non-minimal (i.e., arbitrary) depending on the kind of DP chains they refer to. Minimality is required to use some processors (for instance, the subterm processor [15] or the usable rules processor [11]) and it is the default option when a proof of finiteness starts; also, processors must preserve minimality in the returned DP problems (see [12], for instance). However, minimality is *not* necessary to prove DP problems infinite (the existence of an infinite DP chain, whether is minimal or not, suffices to show it). Hence, assuming minimality may prevent some processors from being applied even if they can be used to prove nontermination (see [27, Section 8] for a more detailed discussion).

In the *open* 2D DP Framework we do *not* assume any default option to label CTRS problems. Instead, we often leave the flag symbol in CTRS problems *open* by using a 'variable' $f \in \mathsf{V}$ from a set of *flag variables* V: an open CTRS problem (or OCTRS problem) is a tuple $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$, where $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ are CTRSs, and the *label* $\varphi \in \mathsf{F} \cup \mathsf{V}$ can be a flag variable or a flag constant [27, Definition 32]. A substitution $\varsigma : \mathsf{V} \to \mathsf{F} \cup \mathsf{V}$ is called a *flag substitution*. The *instance* by ς (or ς-instance) of an OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ is $\varsigma(\check{\tau}) = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varsigma(\varphi))$. Let ς_{a} (resp. ς_{m}) be the *constant* flag substitution that replaces every flag variable by a (resp. m). Note that the instantiation $\varsigma_k(\check{\tau})$ of an open CTRS problem $\check{\tau}$ by a constant flag substitution ς_k for $k \in \mathsf{F}$ is a CTRS problem.

Remark 2 Open CTRS problems $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ cannot be immediately qualified as finite or infinite. Since φ can be a flag variable f, only *after its instantiation* with some constant flag substitution ς_k for some $k \in \mathsf{F}$ to obtain a CTRS problem $\varsigma_k(\check{\tau})$ we can talk of finiteness or infiniteness of the corresponding $\varsigma_k(\check{\tau})$-chains.

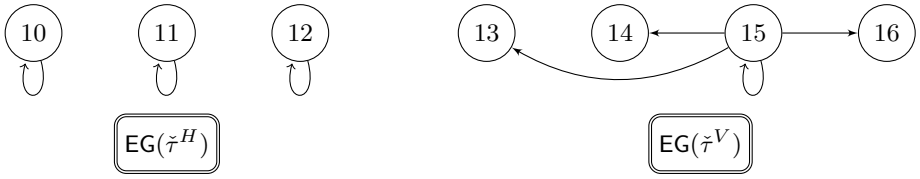
There are two important OCTRS problems: given a *flag variable* $f \in \mathsf{V}$,

$$\check{\tau}^H = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R}, f) \text{ and } \check{\tau}^V = (\mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}), \mathcal{R}, f) \tag{18}$$

Example 4 For $\mathcal{R}$ in Figure 1, we have $\check{\tau}^H = (\{(10), (11), (12)\}, \emptyset, \mathcal{R}, f)$ and $\check{\tau}^V = (\{(13), (14), (15), (16)\}, \{(10), (11), (12)\}, \mathcal{R}, f)$, for $\mathsf{DP}_H(\mathcal{R})$, $\mathsf{DP}_V(\mathcal{R})$ and $\mathsf{DP}_{VH}(\mathcal{R})$ as in Examples 2 and 3.

Given $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$, we define a *graph* $\mathsf{G}(\check{\tau})$ whose nodes are the ($\mathcal{R}$-feasible) rules in $\mathcal{P}$; there is an arc from a node $\alpha : u \to v \Leftarrow c$ to a node $\alpha' : u' \to v' \Leftarrow c'$ iff α, α' is a $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain [27, Section 7.1]. $\mathsf{G}(\check{\tau})$ is not computable, but it can be *(over)estimated* as $\mathsf{EG}(\check{\tau})$ [27, Section 7.1].

Example 5 For $\check{\tau}^H$ and $\check{\tau}^V$ in Example 4, the estimated graphs are:



An *open* CTRS *processor* P is a partial function from OCTRS problems into sets of OCTRS problems; it can also return "no". The OCTRS problems returned by an open processor cannot introduce new flag variables: for any OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$, if $(\mathcal{P}', \mathcal{Q}', \mathcal{R}', \varphi') \in \mathsf{P}(\check{\tau})$, then $\varphi' \in \mathsf{F} \cup \{\varphi\}$ [27, Definition 35]. The domain of P (as a partial function) is $\mathsf{Dom}^{\check{}}(\mathsf{P})$.

Example 6 Processor P_{SCC} [27, Definition 51] decomposes an OCTRS problem $\check{\tau}$ into a (possibly empty) set of OCTRS problems with the rules required to represent the *strongly connected components* (minimal cycles in a graph) in $\mathsf{EG}(\check{\tau})$. For instance, with $\mathsf{EG}(\check{\tau}^H)$ and $\mathsf{EG}(\check{\tau}^V)$ as in Example 5, $\mathsf{P}_{SCC}(\check{\tau}^H) = \{\check{\tau}^H_{11}, \check{\tau}^H_{12}, \check{\tau}^H_{13}\}$ with $\check{\tau}^H_{11} = (\{(10)\}, \emptyset, \mathcal{R}, f)$, $\check{\tau}^H_{12} = (\{(11)\}, \emptyset, \mathcal{R}, f)$, and $\check{\tau}^H_{13} = (\{(12)\}, \emptyset, \mathcal{R}, f)$. Also, $\mathsf{P}_{SCC}(\check{\tau}^V) = \{\check{\tau}^V_1\}$ where $\check{\tau}^V_1 = (\{(15)\}, \emptyset, \mathcal{R}, f)$. Note that the $\mathcal{Q}$ component of $\check{\tau}^V_1$ becomes *empty*. This is because no rule in $\mathsf{DP}_{VH}(\mathcal{R})$ (the $\mathcal{Q}$ component of $\check{\tau}^V$) can be used in any $\check{\tau}^V_1$-chain (see [27, Section 7.1]).

Example 7 Roughly speaking, processor $\mathsf{P}_{\triangleright}$ [27, Section 7.2] removes rules $u \to v \Leftarrow c$ from $\mathcal{P}$ (or $\mathcal{Q}$) in an OCTRS problem $\check{\tau}$ if there is an immediate subterm v_j of $v = g(v_1, \ldots, v_n)$, $1 \le j \le n$ which is a *strict subterm* of an immediate subterm u_i of $u = f(u_1, \ldots, u_m)$, $1 \le i \le m$ (i.e., $u_i \triangleright v_j$). E.g., with rule (10), i.e.,

$$\mathsf{s}(x) -^\sharp \mathsf{s}(y) \to x -^\sharp y$$

we have $\mathsf{s}(x) \triangleright x$. Besides, for any other rules $u' \to v' \Leftarrow c'$ in $\mathcal{P}$ (or $\mathcal{Q}$), the previous extraction of immediate subterms from u' and v' must be compatible with the subterm relation $\trianglerighteq$. Hence, continuing Example 5, we have $\mathsf{P}_{\triangleright}(\check{\tau}^H_{11}) = \{(\emptyset, \emptyset, \mathcal{R}, f)\}$ and, similarly, $\mathsf{P}_{\triangleright}(\check{\tau}^H_{12}) = \mathsf{P}_{\triangleright}(\check{\tau}^H_{13}) = \{(\emptyset, \emptyset, \mathcal{R}, f)\}$.

Remark 3 Roughly speking, in the DP Framework for TRSs, a DP processor P is called *sound* if it proves a DP problem τ finite whenever *all* returned DP problems $\mathsf{P}(\tau)$ are finite. Similarly, P is said to be complete if it proves τ infinite whenever *some* of the returned problems $\mathsf{P}(\tau)$ is infinite. As mentioned in Remark 2, there is no notion of (in)finiteness for open CTRS problems $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ due to the possibility of different instatiations of φ. The definition of soundness and completeness for open CTRS processors must consider all possibilities. Thus, in the 2D DP Framework, we cannot just borrow the usual definition from the DP Framework.

Our most basic notions of soundness and completeness depend on the considered OCTRS problem $\check{\tau}$ and constant flag substitution ς_k (see Remark 3). We make them explicit as a pair $(\check{\tau}, k)$ in the following definition. Let $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) \in \mathsf{Dom}^{\check{}}(\mathsf{P})$ and $k \in \mathsf{F}$ be such that $\varsigma_k(\varphi) = k$ (i.e., the k-chains are represented by φ, either because $\varphi = k$ or because $\varphi \in \mathsf{V}$). An open CTRS processor P is $(\check{\tau}, k)$-*sound* iff $\varsigma_k(\check{\tau})$ is finite whenever $\mathsf{P}(\check{\tau}) \neq$ "no" and for all $\check{\tau}' \in \mathsf{P}(\check{\tau})$, $\varsigma_k(\check{\tau}')$ is finite; P is called $(\check{\tau}, k)$-*complete* iff $\varsigma_k(\check{\tau})$ is infinite whenever $\mathsf{P}(\check{\tau}) =$ "no" or there is $\check{\tau}' \in \mathsf{P}(\check{\tau})$ such that $\varsigma_k(\check{\tau}')$ is infinite. Given $k \in \mathsf{F}$, P is called k-*sound* (k-*complete*) if it is $(\check{\tau}, k)$-sound ($(\check{\tau}, k)$-complete) for all $\check{\tau} \in \mathsf{Dom}^{\check{}}(\mathsf{P})$. P is *sound* (*complete*) if it is k-sound (k-complete) for all $k \in \mathsf{F}$. Soundness of processors is required to prove a CTRS problem *finite*; completeness is required to prove it *infinite*.

Example 8 Processor P_{SCC} (see Example 6) is sound and complete. Processor $\mathsf{P}_{\triangleright}$ (see Example 7) is complete. $\mathsf{P}_{\triangleright}$ is only m-sound if $\mathcal{P} \cup \mathcal{Q}$ contains no rule $\ell \to r \Leftarrow c$

where r is a variable and the root symbols in lhs's and rhs's of rules in $\mathcal{P} \cup \mathcal{Q}$ and the root symbols in the lhs's in $\mathcal{R}$ are disjoint, see [27, Theorem 65]. In general, $\mathsf{P}_{\triangleright}$ is *not* a-sound (see [27, Example 67]).

In the open 2D DP Framework, open processors are applied to OCTRS problems $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ without specifically targeting a proof of finiteness or infiniteness. The application of open processors usually depends on $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ only; φ is not used (see [27, Sections 5.2 & 6.1]). Actually, this just formalizes a usual practice in proofs of termination with dependency pairs (also in the DP Framework for TRSs). As a counterpart, though, we need provide the *conditions* of use of P, guaranteeing soundness and completeness, to finally being able to conclude about finiteness or infiniteness of the considered CTRS problem.

Remark 4 When a processor P is applied to $\check{\tau}$ as part of a proof, the soundness or completeness properties that the application of such a processor exhibits for the considered OCTRS problem $\check{\tau}$ are represented by means of a *plugging scheme*. At the end of the proof, the information in those plugging schemes is considered at once by means of a *set of equations* to obtain an appropriate conclusion of the analysis (if any, see Theorem 1 below).

A *plugging scheme* is a pair $\psi = \langle \varphi_s, \varphi_c \rangle$, where $\varphi_s, \varphi_c \in \mathsf{F}^{\bullet} \cup \mathsf{V}$ for $\mathsf{F}^{\bullet} = \mathsf{F} \cup \{\bullet\}$ with '$\bullet$' a new constant symbol and letters 's' and 'c' referring to *soundness* and *completeness* respectively. The new symbol $\bullet$ is used when soundness or completeness cannot be guaranteed if P is used with the considered OCTRS problem (see Example 9). A processor P *follows* a plugging scheme $\langle \varphi_s, \varphi_c \rangle$ with $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) \in \mathsf{Dom}^{\check{}}(\mathsf{P})$ iff for all $k \in \mathsf{F}$, if $\varsigma_k(\varphi) = \varsigma_k(\varphi_s)$, then P is $(\check{\tau}, k)$-sound; and if $\varsigma_k(\varphi) = \varsigma_k(\varphi_c)$, then P is $(\check{\tau}, k)$-complete.

Example 9 Processor P_{SCC} can be applied to any OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ (i.e., $\mathsf{Dom}^{\check{}}(\mathsf{P})$ is the set of OCTRS problems). Since P_{SCC} is sound (complete), with $f \in \mathsf{V}$, we have that for all $k \in \mathsf{F}$, if $\varsigma_k(\varphi) = \varsigma_k(f) = k$, then P_{SCC} is $(\check{\tau}, \varphi)$-sound ($(\check{\tau}, \varphi)$-complete) for all OCTRS problems $\check{\tau}$. Thus, P_{SCC} follows the plugging scheme $\langle f, f \rangle$ with all OCTRS problems $\check{\tau}$.

The subterm processor $\mathsf{P}_{\triangleright}$ also applies to any OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$. The procesor is complete, but, as remarked in Example 8, it is m-sound only. Furthermore, it is m-sound provided that $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ satisfy some syntactic conditions. Thus, for CTRS problems $\check{\tau}$ satisfying such conditions, we will be able to use the plugging scheme $\langle \mathsf{m}, f \rangle$, which then $\mathsf{P}_{\triangleright}$ follows with $\check{\tau}$. However, if the CTRSs in $\check{\tau}$ do not satisfy the required conditions, we still can use $\mathsf{P}_{\triangleright}$ with the plugging scheme $\langle \bullet, f \rangle$ which $\mathsf{P}_{\triangleright}$ trivially follows by completeness and due to the fact that, for all $k \in \mathsf{F}$, $\varsigma_k(\varphi) \neq \bullet = \varsigma_k(\bullet)$ because $\bullet \notin \mathsf{F}$ (and then no soundness requirement is made on $\mathsf{P}_{\triangleright}$).

The *set* $\Pi(\mathsf{P})$ of plugging schemes associated to a given (open) processor P is obtained from the soundness and completeness results for P. For the processors P in [27], $\Pi(\mathsf{P})$ is in Table 1 (see Example 9 for P_{SCC} and $\mathsf{P}_{\triangleright}$). Note that, although a specific flag variable f has been used in Table 1, this choice is irrelevant due to the use of renamings when plugging schemes from $\Pi(\mathsf{P})$ are used in proofs (see below). Section 5.2 below provides a further discussion about how to establish the plugging schemes to be used in a proof tree as described in the following.

Table 1 Plugging schemes for the processors in [27]

P	$\Pi(\mathsf{P})$	P	$\Pi(\mathsf{P})$
P_{Inf}	$\{\langle\bullet,\mathsf{a}\rangle,\langle\bullet,f\rangle\}$	P_{SCC}	$\{\langle f,f\rangle\}$
$\mathsf{P}_{\triangleright}$	$\{\langle\mathsf{m},f\rangle,\langle\bullet,f\rangle\}$	P_{RT}	$\{\langle f,f\rangle\}$
P_{IR}	$\{\langle f,f\rangle\}$	P_{RIR}	$\{\langle f,\mathsf{a}\rangle\}$
P_{NR}	$\{\langle f,\mathsf{a}\rangle,\langle\bullet,\mathsf{a}\rangle\}$	P_{NQ}	$\{\langle f,\mathsf{a}\rangle,\langle\bullet,\mathsf{a}\rangle\}$

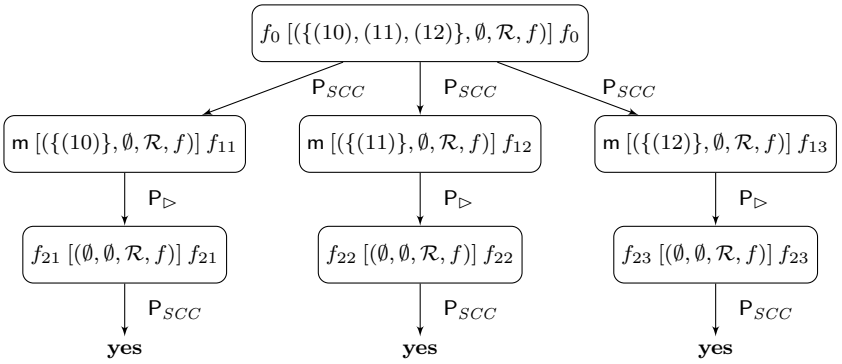


Fig. 3 OCTRSP-tree $\check{\mathcal{T}}^H$ for $\check{\tau}^H = (\{(10),(11),(12)\},\emptyset,\mathcal{R},f)$ of $\mathcal{R}$ in Figure 1

An *open CTRS Proof tree* $\check{\mathcal{T}}$ (OCTRSP-tree) for an OCTRS problem $\check{\tau}_0$ is a tree each of whose nodes is labeled with an OCTRS problem $\check{\tau}$ and a plugging scheme ψ (we often call them "OCTRS label" and "plugging scheme label", respectively). For $\psi = \langle\varphi_s,\varphi_c\rangle$, such a labeling is displayed as follows:

$$\boxed{\varphi_s\ [\ \check{\tau}\ ]\ \varphi_c}$$

The leaves may also be labeled with either "yes" or "no". The root node of $\check{\mathcal{T}}$ is given OCTRS label $\check{\tau}_0$. For each inner node n (including the root node) with OCTRS label $\check{\tau}$, there is an open processor P such that (i) $\check{\tau} \in \check{\mathsf{Dom}}(\mathsf{P})$, and (ii) there is $\psi \in \Pi(\mathsf{P})$ such that P follows ψ with $\check{\tau}$, and then:

1. A renamed version ψ' of ψ (such that flag variables in ψ' occur in no other node in the tree) is used as the plugging scheme label for n.
2. If $\mathsf{P}(\check{\tau}) = \mathsf{no}$, then n has just one child n' with label "no".
3. If $\mathsf{P}(\check{\tau}) = \emptyset$, then n has just one child n' with label "yes".
4. If $\mathsf{P}(\check{\tau}) = \{\check{\tau}_1,\ldots,\check{\tau}_k\}$ with $k > 0$, then n has exactly k children $\mathsf{n}_1,\ldots,\mathsf{n}_k$ with OCTRS labels $\check{\tau}_1,\ldots,\check{\tau}_k$, respectively.

Each outcoming *arc* of n is labeled with P.

Example 10 For $\mathcal{R}$ in Figure 1 and $\check{\tau}^H = (\{(10),(11),(12)\},\emptyset,\mathcal{R},f)$ in Example 4, the OCTRSP-tree $\check{\mathcal{T}}^H$ is in Figure 3. For the application of P_{SCC} to $\check{\tau}^H$ (Example 6), we use the renamed version $\langle f_0,f_0\rangle$ of the (only) plugging scheme $\langle f,f\rangle$ for P_{SCC} (see Table 1). For the applications of $\mathsf{P}_{\triangleright}$ to $\check{\tau}_{11}$, $\check{\tau}_{12}$, and $\check{\tau}_{13}$ (Example 7), we use the renamed versions $\langle\mathsf{m},f_{11}\rangle$, $\langle\mathsf{m},f_{12}\rangle$, and $\langle\mathsf{m},f_{13}\rangle$ of the plugging

scheme $\langle \mathsf{m}, f\rangle$ (see also Table 1). The use of $\langle \mathsf{m}, f\rangle$ instead of $\langle \bullet, f\rangle$ is *inferred* from the conditions of the application of $\mathsf{P}_{\triangleright}$ to OCTRS problems $\check{\tau}_{11}$, $\check{\tau}_{12}$, and $\check{\tau}_{13}$ (see Example 9 and also Section 5.2 for a general discussion on the selection of plugging schemes in proofs).

Note that flag variables used in OCTRS and plugging scheme labels of nodes in $\check{\mathcal{T}}$ are disjoint due to the use of renamings when plugging schemes are introduced. Connections among them are established through unification as follows (see also Theorem 1 below). The set $\mathcal{E}_s(\check{\mathcal{T}})$ of *soundness equations* for $\check{\mathcal{T}}$ consists of an equation $\varphi =^? \varphi_s$ for each inner node n labeled with $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ and $\langle \varphi_s, \varphi_c\rangle$ (including the root). The set $\mathcal{E}_c^{\mathsf{L}}(\check{\mathcal{T}})$ of *completeness equations* for the *path* Γ in $\check{\mathcal{T}}$ leading from the root of $\check{\mathcal{T}}$ to a leaf L consists of an equation $\varphi =^? \varphi_c$ for each node n in Γ labeled with $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ and $\langle \varphi_s, \varphi_c\rangle$.

Example 11 For $\mathcal{R}$ in Figure 1, and the OCTRSP-tree $\check{\mathcal{T}}^H$ in Figure 3, the set of *soundness* equations is $\mathcal{E}_s(\check{\mathcal{T}}^H) = \{f =^? f_0, f =^? \mathsf{m}, f =^? f_{21}, f =^? f_{22}, f =^? f_{23}\}$.

Theorem 1 *Let $\check{\mathcal{T}}$ be an OCTRSP-tree for an OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$.*

1. *If all leaves in $\check{\mathcal{T}}$ are labeled with "*yes*" and there is a ground flag substitution ς that* unifies *$\mathcal{E}_s(\check{\mathcal{T}})$ and $\varsigma(\varphi) \neq \bullet$, then $\varsigma(\check{\tau})$ is* finite.
2. *If there is a leaf n in $\check{\mathcal{T}}$ with label "*no*" and there is a ground flag substitution ς that* unifies *$\mathcal{E}_c^{\mathsf{n}}(\check{\mathcal{T}})$ and $\varsigma(\varphi) \neq \bullet$, then $\varsigma(\check{\tau})$ is* infinite.

Example 12 $\mathcal{E}_s(\check{\mathcal{T}}^H)$ in Example 11 unifies with ς_{m}, i.e., $\varsigma_{\mathsf{m}}(\check{\tau}^H)$ is *finite*.

According to the possible instantiations of f in $\check{\tau}^H$ and $\check{\tau}^V$ and depending on the respective proofs of **F**initeness or **I**nfiniteness, Table 2 shows the possible conclusions of the analyses [27, Table 2].

Table 2 Open CTRS problems for proving termination properties

Termination prop.	$\check{\tau}^H$ / $\check{\tau}^V$	f **instantiation**	**Requirements on** $\mathcal{R}$	**F/I**
Termination	$\check{\tau}^H$	$f \mapsto \mathsf{a}$	preserves terminating substitutions	F
Nontermination	$\check{\tau}^H$	$f \mapsto \mathsf{m}$ or $f \mapsto \mathsf{a}$	None	I
V-Termination	$\check{\tau}^V$	$f \mapsto \mathsf{a}$	deterministic 3-CTRS	F
Non-V-Termination	$\check{\tau}^V$	$f \mapsto \mathsf{m}$ or $f \mapsto \mathsf{a}$	None	I
Op. termination	$\check{\tau}^H$	$f \mapsto \mathsf{m}$ or $f \mapsto \mathsf{a}$	deterministic 3-CTRS	F
	$\check{\tau}^V$	$f \mapsto \mathsf{m}$ or $f \mapsto \mathsf{a}$		F
Op. nontermination	$\check{\tau}^H$	$f \mapsto \mathsf{m}$ or $f \mapsto \mathsf{a}$	None	I
Op. nontermination	$\check{\tau}^V$	$f \mapsto \mathsf{m}$ or $f \mapsto \mathsf{a}$	None	I

4 New processors for the (Open) 2D DP Framework

In this section we introduce five new processors for their use in the Open 2D DP Framework and illustrate their application with several examples. As pointed out in the Introduction, the addition of these new processors is the reason for the 2D

DP Framework to currently outperform in practice all other tools in CTRS proofs of operational termination. The following notation is used in the definition of our processors.

Notation 1 (Replacement of conditions) *Given a conditional rule $\alpha : \ell \to r \Leftarrow c$, where c is $s_1 \to t_1, \ldots, s_n \to t_n$ for some $n > 0$, an index $i \in \{1, \ldots, n\}$, and a (possibly empty) sequence d of conditions $s'_{i1} \to t'_{i1}, \ldots, s'_{im} \to t'_{im}$ for some $m \geq 0$, we write $\ell \to r \Leftarrow c[d]_i$ to denote the new rule which is obtained by replacing condition $s_i \to t_i$ by d in the conditional part c of α, i.e., $c[d]_i$ is as follows:*

$$s_1 \to t_1, \ldots, s_{i-1} \to t_{i-1}, s'_{i1} \to t'_{i1}, \ldots, s'_{im} \to t'_{im}, s_{i+1} \to t_{i+1}, \ldots, s_n \to t_n$$

In the following, we denote the empty *sequence of conditions as $\diamond$. Thus, $\ell \to r \Leftarrow c[\diamond]_i$ denotes the* removal *of condition $s_i \to t_i$ from c in α.*

Notation 2 (Replacement of rules) *Given a CTRS $\mathcal{R}$ and a rule α, the (possible)* replacement *$\mathcal{R}[\mathcal{S}]_\alpha$ of α in $\mathcal{R}$ by the (possibly empty) set of rules $\mathcal{S}$ is $\mathcal{R}[\mathcal{S}]_\alpha = (\mathcal{R} - \{\alpha\}) \cup \mathcal{S}$ if $\alpha \in \mathcal{R}$; and $\mathcal{R}[\mathcal{S}]_\alpha = \mathcal{R}$ otherwise. Note that, $\mathcal{R}[\emptyset]_\alpha$ denotes the* removal *of α from $\mathcal{R}$ if α belongs to $\mathcal{R}$.*

The first three processors we consider here have a common focus: they investigate the *conditional part* c of a rule $\alpha : u \to v \Leftarrow c$ in a OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ to try to either: (i) *transform a condition* in the rule, or (ii) *remove a condition* from the rule. Processor P_{NC} unfolds the left-hand side s of a condition $s \to t \in c$ by using narrowing, so that the 'reachability distance' between the instances $\sigma(s)$ and $\sigma(t)$ becomes 'shorter' (Section 4.1). On the other hand, if s and t actually *unify*, without any possibility of rewriting (instances of) s, one may think of *removing* the condition, provided that appropriate instantiations of variables are propagated to the remainder of the rule; processors P_{SUC} and P_{SUNC} implement this approach (Section 4.2).

In [27, Section 7.3], the *Removal Triple Processor* P_{RT} which uses well-founded relations to *simplify* OCTRS problems was introduced: a pair $\alpha : u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$ can be removed from $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, f)$ provided that u and v can be compared by using a *well-founded relation* $\sqsupset$ *and* some additional conditions are fulfilled. In Section 4.3 we show that the application of P_{RT} to $\check{\tau}$ to remove α can be transformed into a satisfiability problem. We introduce a many-sorted first-order theory $\mathcal{S}^{RT}_{\check{\tau},\alpha}$ and a first-order formula $\phi^{\sqsupset}_{\alpha}$ associated to α. If there is an *interpretation* $\mathcal{A}$ satisfying $\mathcal{S}^{RT}_{\check{\tau},\alpha} \cup \{\phi^{\sqsupset}_{\alpha}\}$, then α can be removed as desired.

In Section 4.4 we improve P_{RT} so that the amount of rules from $\mathcal{R}$ that *need* to be dealt with is substantially reduced in most cases. As for P_{RT}, we provide a *semantic* approach to deal with such a new processor P_{RTN}.

The following auxiliary results are used later.

Corollary 1 [27, Corollary 27] *Let P be a processor such that, for all OCTRS problems $\tau = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, f) \in \mathsf{Dom}(\mathsf{P})$, $\mathsf{P}(\tau) \neq \mathsf{no}$ and if $\tau' = (\mathcal{P}', \mathcal{Q}', \mathcal{R}', f') \in \mathsf{P}(\tau)$, then $\mathcal{P}' \subseteq \mathcal{P}$, $\mathcal{Q}' \subseteq \mathcal{Q}$, $\mathcal{R}' = \mathcal{R}$ and $f' = \mathsf{m}$. Then, P is complete.*

Lemma 1 *Let $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ be such that $\mathcal{V}ar(s) \cap \mathcal{V}ar(t) = \emptyset$, and $\sigma(s) = \varsigma(t)$ for substitutions σ and ς. Then, s and t unify.*

Proof Trivial. □

Proposition 1 *Let $\mathcal{R}$ be a CTRS and s and t be terms with $\mathcal{V}ar(s) = \{x_1, \ldots, x_n\}$ and such that $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ for some substitution σ. If (1) s and t do not unify, (2) s is linear, and (3) $\mathcal{V}ar(s) \cap \mathcal{V}ar(t) = \emptyset$, then there is a substitution σ' satisfying $\sigma(x_i) \to^*_{\mathcal{R}} \sigma'(x_i)$ for all $1 \leq i \leq n$, and $p \in \mathcal{P}os_{\mathcal{F}}(s)$ such that $\sigma(s) = s[\sigma(x_1), \ldots, \sigma(x_n)] \to^* s[\sigma'(x_1), \ldots, \sigma'(x_n)] \xrightarrow{p}_{\mathcal{R}} w \to^* \sigma(t)$.*

Proof Assume that p does not exist. By (2), we can write

$$\sigma(s) = s[\sigma(x_1), \ldots, \sigma(x_n)] \to^*_{\mathcal{R}} s[\sigma'(x_1), \ldots, \sigma'(x_n)] = \sigma(t)$$

Thus, $\sigma'(s) = \sigma(t)$. By (3) and Lemma 1, s and t unify, contradicting (1).

4.1 Narrowing the conditions of the rules

Reachability problems $\sigma(s) \to^* \sigma(t)$ are often investigated using *narrowing* and *unification* directly over terms s and t. The following definition provides a suitable extension of *narrowing* for CTRSs from the usual definition for TRSs.

Definition 2 [27, Definition 79] Let $\mathcal{R}$ be a CTRS. A term s *narrows* to a term t (written $s \rightsquigarrow_{\mathcal{R},\theta,p} t$ or just $s \rightsquigarrow_{\mathcal{R},\theta} t$ or even $s \rightsquigarrow t$), iff there is a nonvariable position $p \in \mathcal{P}os_{\mathcal{F}}(s)$, a *renamed* rule $\ell \to r \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n$ in $\mathcal{R}$, substitutions $\theta_0, \ldots, \theta_n, \tau_1, \ldots, \tau_n$, and terms $t'_1, \ldots, t'_n$ such that:

1. $s|_p =^?_{\theta_0} \ell$,
2. for all i, $1 \leq i \leq n$, $\eta_{i-1}(s_i) \rightsquigarrow^*_{\mathcal{R},\theta_i} t'_i$ and $t'_i =^?_{\tau_i} \theta_i(\eta_{i-1}(t_i))$, where $\eta_0 = \theta_0$ and for all $i > 0$, $\eta_i = \tau_i \circ \theta_i \circ \eta_{i-1}$, and
3. $t = \theta(s[r]_p)$, where $\theta = \eta_n$.

We write $u \rightsquigarrow^*_{\mathcal{R},\beta} v$ for terms u, v and substitution β iff there are terms $u_1, \ldots, u_{m+1}$ and substitutions $\beta_1, \ldots, \beta_m$ for some $m \geq 0$ such that

$$u = u_1 \rightsquigarrow_{\mathcal{R},\beta_1} u_2 \rightsquigarrow_{\mathcal{R},\beta_2} \cdots \rightsquigarrow_{\mathcal{R},\beta_m} u_{m+1} = v$$

and $\beta = \beta_m \circ \cdots \circ \beta_1$ (or $\beta = \varepsilon$ if $m = 0$).

Example 13 Consider the CTRS $\mathcal{R}$ in Figure 1. We can apply a narrowing step to term $\mathsf{leq}(y, x)$ by using (renamed versions of) rules (6) and (7):

$$\underline{\mathsf{leq}(y, x)} \rightsquigarrow_{(6)',\{x \mapsto \mathsf{s}(y'), y \mapsto \mathsf{s}(x')\}} \mathsf{leq}(x', y') \tag{19}$$

$$\underline{\mathsf{leq}(y, x)} \rightsquigarrow_{(7)',\{x \mapsto x', y \mapsto \mathsf{0}\}} \mathsf{true} \tag{20}$$

No further narrowing step is possible on $\mathsf{leq}(y, x)$.

Given a CTRS $\mathcal{S}$, $NRules(\mathcal{S}, s)$ is the set of rules $\alpha : \ell \to r \Leftarrow c \in \mathcal{S}$ such that a nonvariable subterm t of s is a *narrex* of α, i.e., t and ℓ unify with *mgu* θ_0 (we assume $\mathcal{V}ar(t) \cap \mathcal{V}ar(\ell) = \emptyset$), and $\theta_0(c)$ is $\mathcal{S}$-feasible, i.e., there is a substitution σ such that for all $s \to t \in c$, $\sigma(\theta_0(s)) \to^*_{\mathcal{S}} \sigma(\theta_0(t))$ holds, cf. [20, Definition 2].

Example 14 For $\mathsf{leq}(y, x)$ and $\mathcal{R}$ in Ex. 13, $NRules(\mathcal{R}, \mathsf{leq}(y, x)) = \{(6), (7)\}$.

Then, $N_1(\mathcal{S}, s)$ represents the set of one-step $\mathcal{S}$-narrowings issued from s:

$$N_1(\mathcal{S}, s) = \{(t, \theta{\downarrow_{\mathcal{V}ar(s)}}) \mid s \rightsquigarrow_{\ell \to r \Leftarrow c, \theta} t, \ell \to r \Leftarrow c \in \mathit{NRules}(\mathcal{S}, s)\} \tag{21}$$

where $\theta{\downarrow_{\mathcal{V}ar(s)}}$ is a substitution defined by $\theta{\downarrow_{\mathcal{V}ar(s)}}(x) = \theta(x)$ if $x \in \mathcal{V}ar(s)$ and $\theta{\downarrow_{\mathcal{V}ar(s)}}(x) = x$ otherwise. As discussed in [27, Section 7.5], $N_1(\mathcal{S}, s)$ can be *infinite* if $\mathit{NRules}(\mathcal{S}, s)$ is *not* a TRS, i.e., it contains 'proper' conditional rules. In [27, Proposition 87] some sufficient conditions for finiteness of $N_1(\mathcal{S}, s_i)$ are given.

The following example shows that the natural idea of transforming $u \to v \Leftarrow c$ by instantiation with the narrowing substitutions θ used to narrow a given condition $s \to t \in c$ (as done by P_{NR} and P_{NQ} in [27, Section 7.5]) may lead to an unsound processor.

Example 15 Consider the following CTRS:

$$\mathsf{b} \to \mathsf{c} \tag{22}$$

$$\mathsf{h}(\mathsf{c}) \to \mathsf{d} \tag{23}$$

$$\mathsf{f}(x) \to \mathsf{f}(\mathsf{b}) \Leftarrow x \to \mathsf{b}, \mathsf{h}(x) \to \mathsf{d} \tag{24}$$

Note that $\mathsf{f}(\mathsf{b}) \to \mathsf{f}(\mathsf{b})$, i.e., the CTRS is not terminating. $\mathsf{DP}_H(\mathcal{R})$ consists of a single pair:

$$\mathsf{F}(x) \to \mathsf{F}(\mathsf{b}) \Leftarrow x \to \mathsf{b}, \mathsf{h}(x) \to \mathsf{d} \tag{25}$$

If we narrow the *lhs* of condition $\mathsf{h}(x) \to \mathsf{d}$ using $\mathsf{h}(x) \rightsquigarrow_{\{x \mapsto \mathsf{c}\}} \mathsf{d}$ and instantiate the rule with substitution $\{x \mapsto \mathsf{c}\}$, we obtain

$$\mathsf{F}(\mathsf{c}) \to \mathsf{F}(\mathsf{b}) \Leftarrow \mathsf{c} \to \mathsf{b}, \mathsf{d} \to \mathsf{d} \tag{26}$$

which is an *infeasible* rule. Thus, the transformed pair has no associated (finite or infinite) chain and we would wrongly conclude operational termination of $\mathcal{R}$.

In order to avoid the problem discussed in Example 15, we *delay* any possible instantiation of the rules to be done by means of processor P_{SUC} in the next section. In this way, each variable can be considered individually. For this purpose, the bindings in the substitution part of the narrowing steps are included in the conditional part of the new rule as *additional* conditions. Given a rule $\alpha : \ell \to r \Leftarrow c$ with n conditions and i, $1 \leq i \leq n$, we let

$$\overline{\mathcal{N}}(\mathcal{S}, \alpha, i) = \{\ell \to r \Leftarrow c[\boldsymbol{\theta}, w \to t_i]_i \mid s_i \to t_i \in c, (w, \theta) \in N_1(\mathcal{S}, s_i)\}$$

where $\boldsymbol{\theta}$ consists of new conditions $x_1 \to \theta(x_1), \ldots, x_m \to \theta(x_m)$ obtained from the bindings in θ for variables in $\mathcal{V}ar(s_i) = \{x_1, \ldots, x_m\}$.

Example 16 For instance, rule (25) in Example 15 is transformed as follows:

$$\mathsf{F}(x) \to \mathsf{F}(\mathsf{b}) \Leftarrow x \to \mathsf{b}, x \to \mathsf{c}, \mathsf{d} \to \mathsf{d} \tag{27}$$

Clearly, $\overline{\mathcal{N}}(\mathcal{S}, \alpha, i)$ is finite iff $N_1(\mathcal{S}, s_i)$ is finite (up to variable renaming) Note that rules in $\overline{\mathcal{N}}(\mathcal{S}, \alpha, i)$ are *deterministic* if α is. Thus, we define the following processor.

Definition 3 (Narrowing the conditions of rules) Let $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an OC-TRS problem, $\alpha : u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$, $s_i \to t_i \in c$, and $\mathcal{N} \subseteq \overline{\mathcal{N}}(\mathcal{R}, \alpha, i)$ finite. P_{NC} is given by

$$\mathsf{P}_{NC}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) = \{(\mathcal{P}[\mathcal{N}]_\alpha, \mathcal{Q}[\mathcal{N}]_\alpha, \mathcal{R}, \varphi)\}$$

Theorem 2 P_{NC} *is complete. If* $\mathcal{N} = \overline{\mathcal{N}}(\mathcal{R}, \alpha, i)$ *and* $s_i \to t_i \in c$ *is such that* s_i *and* t_i *do* not *unify and either* s_i *is* ground *and* $\mathcal{R}$ *is a 2-CTRS or (1)* $NRules(\mathcal{R}, s_i)$ *is a TRS and (2)* s_i *is linear, and (3)* $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t_i) = \emptyset$*, then* P_{NC} *is* $\check{\tau}$*-sound. Therefore,* $\Pi(\mathsf{P}_{NC}) = \{\langle f, f\rangle, \langle \bullet, f\rangle\}$.

Proof Let $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an OCTRS problem, $\alpha : u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$, $s_i \to t_i \in c$, and $\mathcal{N} \subseteq \overline{\mathcal{N}}(\mathcal{R}, \alpha, i)$. Let $\mathcal{P}' = \mathcal{P}[\mathcal{N}]_\alpha$ and $\mathcal{Q}' = \mathcal{Q}[\mathcal{N}]_\alpha$. With regard to *completeness*, assume the existence of an infinite $(\mathcal{P}', \mathcal{Q}', \mathcal{R})$-O-chain A

$$\sigma(u^1) \to_{\mathcal{P}'} \sigma(v^1)(\to_{\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q}'})^*\sigma(u^2) \to_{\mathcal{P}'} \sigma(v^2)(\to_{\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q}'})^*\sigma(u^3) \to_{\mathcal{P}'} \cdots$$

for some substitution σ. Assume that $\alpha' : u \to v \Leftarrow c[\boldsymbol{\theta}_{s_i}, w \to t_i]_i \in \mathcal{N}$ occurs in A for some $(w, \theta) \in N_1(\mathcal{R}, s_i)$. Thus, for all $s \to t \in c[\diamond]_i$, $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$. Also, $\sigma(x) \to^*_{\mathcal{R}} \sigma(\theta(x))$ for all $x \in \mathcal{V}ar(s_i)$, for the conditions $\boldsymbol{\theta}_{s_i}$, and $\sigma(w) \to^*_{\mathcal{R}} \sigma(t_i)$. Since $s_i \rightsquigarrow_{\mathcal{R},\theta} w$, we also have $\sigma(\theta(s_i)) \to_{\mathcal{R}} \sigma(w)$, hence $\sigma(s_i) \to^*_{\mathcal{R}} \sigma(\theta(s_i)) \to^*_{\mathcal{R}} \sigma(t_i)$. Therefore, when replacing α' by α in A, we obtain an infinite $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain with substitution σ. For all $i \geq 1$, $\sigma(v^i)$ remains operationally terminating because $\mathcal{R}$ has not changed. Thus, minimality is preserved as well.

For *soundness*, assume that there is an infinite $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain A

$$\sigma(u^1) \to_{\mathcal{P}} \sigma(v^1)(\to_{\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q}})^*\sigma(u^2) \to_{\mathcal{P}} \sigma(v^2)(\to_{\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q}})^*\sigma(u^3) \to_{\mathcal{P}} \cdots$$

for some substitution σ. Assume that $\alpha : u \to v \Leftarrow c$ is used in A (the proof for $\alpha \in \mathcal{Q}$ would be analogous). Then, $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ holds for all $s \to t \in c$. Since s_i and t_i do not unify, there is at least one rewriting step in the sequence $\sigma(s_i) \to^*_{\mathcal{R}} \sigma(t_i)$, i.e., we actually have $\sigma(s_i) \to^+_{\mathcal{R}} \sigma(t_i)$. Now we consider two cases. First, assume that s_i is *ground*. Then, $\sigma(s_i) = s_i$ and narrowing on s_i is just rewriting and $N_1(\mathcal{R}, s_i)$ can be assumed to be finite (by [27, Proposition 87]) and containing representations (w, ε) of *all* possible reducts w of s_i by $\mathcal{R}$. For each sequence $\sigma(s_i) = s_i \to^+_{\mathcal{R}} \sigma(t_i)$, we have $s_i \to_{\mathcal{R}} w \to^* \sigma(t_i)$ for some of these reducts. Thus, since $\mathcal{N} = \overline{\mathcal{N}}(\mathcal{R}, \alpha, i)$, we can replace each occurrence of α in A by an appropriate rule $\alpha' : u \to v \Leftarrow c[w \to t_i]_i \in \mathcal{N}$ to obtain a new infinite chain A' with the same substitution σ.

Now, if s_i is *not* ground, then by the hypothesis s_i is linear and properties (1) and (3) hold. Then, by Proposition 1 we can write $\sigma(s_i) \to^*_{\mathcal{R}} \sigma(t_i)$ as:

$$\sigma(s_i) \overset{>\Lambda}{\to}{}^*_{\mathcal{R}} \sigma'(s_i) \overset{p}{\to}_{\mathcal{R}} w' \to^*_{\mathcal{R}} \sigma(t_i) \tag{28}$$

for some substitution σ' such that $\sigma(x) \to^*_{\mathcal{R}} \sigma'(x)$ for all $x \in \mathcal{V}ar(s_i)$, and where $p \in \mathcal{P}os_{\mathcal{F}}(s_i)$, $\sigma'(s_i)|_p = \sigma'(s_i|_p) = \gamma(\ell)$ for some rule $\beta : \ell \to r \Leftarrow d \in \mathcal{R}$, and substitution γ, and $w' = \sigma'(s_i)[\gamma(r)]_p$. Note that β belongs to $NRules(\mathcal{R}, s_i)$ and, by (1), it is actually an *unconditional rule* $\ell \to r$. Therefore, there is a narrowing w of s_i with β and mgu θ, i.e.,

$$s_i \rightsquigarrow_{\beta,\theta,p} \theta(s_i[r]_p) = w \tag{29}$$

where (since β is an unconditional rule) θ is the *mgu* of $s_i|_p$ and ℓ and $(w, \theta) \in N_1(\mathcal{R}, s_i)$. Thus, there is a substitution ϕ such that, for all $x \in \mathcal{V}ar(s_i|_p)$, $\sigma'(x) = \phi(\theta(x))$ and for all $y \in \mathcal{V}ar(\ell)$, $\gamma(x) = \phi(\theta(y))$ (hence $\sigma'(s_i|_p) = \phi(\theta(s_i|_p))$ and $\gamma(\ell) = \phi(\theta(\ell))$). Note that,

$$\text{for all } x \in \mathcal{V}ar(s_i), \sigma(x) \to^*_{\mathcal{R}} \sigma'(x) = \phi(\theta(x)). \tag{30}$$

Hence, $\sigma'(s_i|_p) = \phi(\theta(s_i|_p)) = \phi(\theta(\ell)) = \gamma(\ell)$ and $w' = \sigma'(s_i)[\gamma(r)]_p = \phi(\theta(s_i[r]_p))$.

We define $\hat{\sigma}$ as follows: for all variables x, $\hat{\sigma}(x) = \sigma(x)$ if $x \notin \mathcal{R}ng(\theta) = \bigcup_{x \in \mathcal{V}ar(s_i)} \mathcal{V}ar(\theta(x))$, and $\hat{\sigma}(x) = \phi(x)$ otherwise. Note that $w' = \hat{\sigma}(\theta(s_i[r]_p)) = \hat{\sigma}(w)$. Therefore, from (30) and (28), respectively, we have

$$\text{for all } x \in \mathcal{V}ar(s_i), \hat{\sigma}(x) \to_{\mathcal{R}}^* \hat{\sigma}(\theta(x)) \tag{31}$$

$$\hat{\sigma}(w) = w' \to_{\mathcal{R}}^* \sigma(t_i) = \hat{\sigma}(t_i) \tag{32}$$

where $\sigma(t_i) = \hat{\sigma}(t_i)$ in (32) holds because we can assume that $\mathcal{V}ar(t_i) \cap \mathcal{R}ng(\theta) = \emptyset$. Thus, if we replace α by $\alpha' : u \to v \Leftarrow c[\boldsymbol{\theta}_{s_i}, w' \to t_i]_i \in \mathcal{N}(\mathcal{R}, \alpha, i)$ in A we obtain a new chain A' with substitution $\hat{\sigma}$. Thus, A' is an infinite (minimal) $(\mathcal{P}[\mathcal{N}(\mathcal{R}, \alpha, i)]_\alpha, \mathcal{Q}[\mathcal{N}(\mathcal{R}, \alpha, i)]_\alpha, \mathcal{R})$-O-chain. □

Example 17 For $\mathcal{R}$ in Figure 1 and $\check{\tau}_1^V = (\{(15)\}, \emptyset, \mathcal{R}, f)$ in Example 6, since (15) is

$$\mathsf{DIV}(x, y) \to \mathsf{DIV}(x - y, y) \Leftarrow \mathsf{leq}(y, x) \to \mathsf{true}$$

we have $\mathsf{P}_{NC}(\check{\tau}_1^V) = \{\check{\tau}_2^V\}$, where $\check{\tau}_2^V = (\{(33), (34)\}, \emptyset, \mathcal{R}, f)$ with (see the narrowings computed in Example 13):

$$\mathsf{DIV}(x, y) \to \mathsf{DIV}(x - y, y) \Leftarrow x \to \mathsf{s}(y'), y \to \mathsf{s}(x'), \mathsf{leq}(x', y') \to \mathsf{true} \tag{33}$$

$$\mathsf{DIV}(x, y) \to \mathsf{DIV}(x - y, y) \Leftarrow x \to x', y \to \mathsf{0}, \mathsf{true} \to \mathsf{true} \tag{34}$$

Since (1) $\mathit{NRules}(\mathcal{R}, \mathsf{leq}(y, x))$ is a TRS (see Example 14) and (2) $\mathsf{leq}(y, x)$ is linear, by Theorem 2 we can use the plugging scheme $\langle f, f \rangle$.

Removing the no unification requirement may prevent P_{NC} from being sound.

Example 18 Consider the CTRS $\{\mathsf{a} \to \mathsf{b}, \mathsf{c} \to \mathsf{d} \Leftarrow \mathsf{a} \to \mathsf{a}\}$. The left-hand side a of the condition in the second rule narrows into b. But $\mathsf{c} \to \mathsf{d} \Leftarrow \mathsf{b} \to \mathsf{a}$ (as obtained by P_{NC}), now *forbids* the rewriting step $\mathsf{c} \to \mathsf{d}$.

Also removing the disjointness requirement could be dangerous.

Example 19 Consider $\check{\tau} = (\{\alpha\}, \emptyset, \mathcal{R}, \varphi)$ with $\alpha : \mathsf{B}(x) \to \mathsf{B}(x) \Leftarrow \mathsf{g}(x) \to \mathsf{g}(\mathsf{f}(x))$ and $\mathcal{R} = \{\mathsf{a} \to \mathsf{f}(\mathsf{a})\}$. Since $\mathsf{g}(\underline{\mathsf{a}}) \to_{\mathcal{R}} \mathsf{g}(\mathsf{f}(\mathsf{a}))$, there is an infinite chain $(\alpha_i)_{i \geq 1}$ with α_i obtained by renaming variable x in α as x_i and substitution σ defined as $\sigma(x_i) = \mathsf{a}$ for all $i \geq 1$. Thus, $\check{\tau}$ is infinite.

Note that $\mathsf{g}(x)$ and $\mathsf{g}(\mathsf{f}(x))$ do *not* unify. Since $\overline{\mathcal{N}}(\mathcal{R}, \alpha, 1)$ is empty, $\mathsf{P}_{NC}(\check{\tau}) = \{(\emptyset, \emptyset, \mathcal{R}, \varphi)\}$ would wrongly prove finiteness of $\check{\tau}$.

Note that the applicability of P_{NC} in Theorem 2 when we want to preserve soundness depends on the computability of $\mathcal{N} = \overline{\mathcal{N}}(\mathcal{R}, \alpha, i)$. In practice, we use the sufficient conditions (1)–(2) given in [27, Proposition 87]: either $\mathit{NRules}(\mathcal{R}, t)$ is a TRS, or t is ground and $\mathcal{R}$ is a 2-CTRS. Sufficient Condition (3) (t is ground and $\mathcal{U}(\mathcal{R}, t)$ is a terminating and deterministic 3-CTRS) could be used in practice if the obtained 3-CTRS is easy to prove operationally terminating.

4.2 Simplification by unification

Conditions $s \to t$ in the conditional part c of a rule $\ell \to r \Leftarrow c$ may fulfill the intended reachability condition $\sigma(s) \to^* \sigma(t)$ without issuing any rewriting step. Satisfiability of a condition $s \to t$ in 0 steps can then be viewed as *unification* problems $s =^? t$. Therefore, $s \to t$ can be *removed* from c if we instantiate the rule with the *most general unifier* θ of s and t. In this section we exploit this idea to define a new processorP_{SUC}.

Notation 3 *Given a rule $\alpha : \ell \to r \Leftarrow c$ with n conditions, $i \in \{1, \ldots, n\}$, and a substitution θ, we let $\alpha_{\theta,i}$ be the rule $\alpha_{\theta,i} : \theta(\ell) \to \theta(r) \Leftarrow \theta(c[\diamond]_i)$.*

Definition 4 (Simplifying unifiable conditions) Let $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an OCTRS problem, $\alpha : \ell \to r \Leftarrow c \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{R}$, and $s_i \to t_i \in c$. P_{SUC} is given by

$$\mathsf{P}_{SUC}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) = \{(\mathcal{P}[\{\alpha_{\theta,i}\}]_\alpha, \mathcal{Q}[\{\alpha_{\theta,i}\}]_\alpha, \mathcal{R}[\{\alpha_{\theta,i}\}]_\alpha, \varphi)\}$$

iff $s_i =^?_\theta t_i$.

Before being able to establish the correctness and completeness results for P_{SUC}, we need some auxiliary results. Our first result establishes that for non-narrowable linear terms s, any one-step reduction occurring after the instantiation with a substitution σ to s is 'captured' by one of the variables of s and the obtained term t can be seen as an instance $\sigma'(s)$ of s by a new substitution σ' which is obtained for σ by a one-step reduction of its bindings.

Proposition 2 *Let $\mathcal{R}$ be a CTRS, σ be a substitution, and $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ be such that s is linear and $NRules(\mathcal{R}, s) = \emptyset$. If $\sigma(s) \to_\mathcal{R} t$, then $t = \sigma'(s)$ for some substitution σ' and there is $x \in \mathcal{V}ar(s)$ such that $\sigma(x) \to_\mathcal{R} \sigma'(x)$ and $\sigma(y) = \sigma'(y)$ for all $y \in \mathcal{V}ar(s) - \{x\}$.*

Proof By induction on the depth N of the proof tree for $\sigma(s) \to_\mathcal{R} t$. If (Rl) applies, i.e., $\sigma(s) = \gamma(\ell)$ for some $\ell \to r \Leftarrow c \in \mathcal{R}$ and substitution γ, then, since we can assume $\mathcal{V}ar(s) \cap \mathcal{V}ar(\ell) = \emptyset$, by Lemma 1, s and ℓ unify. Therefore, $NRules(\mathcal{R}, s) \neq \emptyset$. This contradicts our assumption, unless we have $s = x \in \mathcal{X}$. Then, we can define $\sigma'(x) = t$ and the conclusion follows. If (C) applies, we consider two cases: (i) if $s = x \in \mathcal{X}$, then $\sigma(x) = f(s_1, \ldots, s_i, \ldots, s_k)$ and $t = f(s_1, \ldots, t_i, \ldots, s_k)$ and $s_i \to_\mathcal{R} t_i$ for some i, $1 \leq i \leq k$ and terms $s_1, \ldots, s_k, t_i$. Again, we just let $\sigma'(x) = t$. (ii) If $s = f(s_1, \ldots, s_i, \ldots, s_k)$, then $t = f(t_1, \ldots, t_i, \ldots, t_k)$ and $\sigma(s_i) \to_\mathcal{R} t_i$ for some i, $1 \leq i \leq k$. By the I.H., $t_i = \sigma'(s_i)$ and there is $x \in \mathcal{V}ar(s_i)$ such that $\sigma(x) \to_\mathcal{R} \sigma'(x)$ and $\sigma(y) = \sigma'(y)$ for all $y \in \mathcal{V}ar(s_i) - \{x\}$. Since s is linear, we can extend σ' to the other variables $z \in \mathcal{V}ar(s) - \mathcal{V}ar(s_i)$ by $\sigma'(z) = \sigma(z)$. Then, $t = \sigma'(s)$ as desired. □

Corollary 2 *Let $\mathcal{R}$ be a CTRS, σ be a substitution, and $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ be such that s is linear and $NRules(\mathcal{R}, s) = \emptyset$. If $\sigma(s) \to^*_\mathcal{R} t$, then $t = \sigma'(s)$ for some substitution σ' and for all $x \in \mathcal{V}ar(s)$, $\sigma(x) \to^*_\mathcal{R} \sigma'(x)$.*

Proposition 3 *Let $\mathcal{R}$ be a CTRS, $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ be such that s is linear, $\mathcal{V}ar(s) \cap \mathcal{V}ar(t) = \emptyset$, and $NRules(\mathcal{R}, s) = \emptyset$. Let σ be a substitution such that $\sigma(s) \to^*_\mathcal{R} \sigma(t)$. Then, $s =^?_\theta t$ and there is a substitution ϕ such that, (1) for all $x \in \mathcal{V}ar(s)$, $\sigma(x) \to^*_\mathcal{R} \phi(\theta(x))$ and (2) for all $x \in \mathcal{V}ar(t)$, $\phi(\theta(x)) = \sigma(x)$.*

Proof By Corollary 2, $\sigma(t) = \sigma'(s)$ for some substitution σ' such that, for all $x \in \mathcal{V}ar(s)$, $\sigma(x) \to^*_{\mathcal{R}} \sigma'(x)$. Since $\mathcal{V}ar(s) \cap \mathcal{V}ar(t) = \emptyset$, by Lemma 1, s and t unify with *mgu* θ with $\sigma = \phi \circ \theta$ and $\sigma' = \phi \circ \theta$. Thus, for all $x \in \mathcal{V}ar(s)$, $\sigma(x) \to^*_{\mathcal{R}} \phi(\theta(x))$ and for all $x \in \mathcal{V}ar(t)$, $\sigma(x) = \phi(\theta(x))$, as desired. □

Proposition 4 *Let $\mathcal{R}$ be a CTRS, $\alpha : \ell \to r \Leftarrow c \in \mathcal{R}$ and $s_i \to t_i \in c$ be such that $s_i =^?_\theta t_i$. Assume that: (1) s_i is linear, (2) $NRules(\mathcal{R}, s_i) = \emptyset$, (3) $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t_i) = \emptyset$, (4) for all $s \to t \in c[\diamond]_i$, $\mathcal{V}ar(s \to t) \cap \mathcal{V}ar(s_i) = \emptyset$, and (5) $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(r) = \emptyset$. Let $\alpha' : \theta(\ell) \to \theta(r) \Leftarrow \theta(c[\diamond]_i)$ and u, v be terms. If $u \to_{\mathcal{R}} v$ ($u \to^*_{\mathcal{R}} v$), then $u \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$.*

Proof By simultaneous induction on the depth N of the proof trees for $u \to_{\mathcal{R}} v$ and $u \to^*_{\mathcal{R}} v$. If $N = 0$, then (i) if $u \to_{\mathcal{R}} v$, then there is a single application of (Rl) where $u = \sigma(\ell)$ and $v = \sigma(r)$ for some *unconditional* rule $\ell \to r \in \mathcal{R}$. Since $\ell \to r \in \mathcal{R}[\{\alpha'\}]_\alpha$, we have $u \to_{\mathcal{R}[\{\alpha'\}]_\alpha} v$. (ii) If $u \to^*_{\mathcal{R}} v$, then there is a single application of (Rf), i.e., $u = v$ and therefore $u \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$.

If $N > 0$, then (i) if $u \to^*_{\mathcal{R}} v$, there is an application of $(Tran)$, i.e., there is a term w such that $u \to_{\mathcal{R}} w$ and $w \to^*_{\mathcal{R}} v$ holds. By the I.H., $u \to_{\mathcal{R}[\{\alpha'\}]_\alpha} w$ and $w \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$ hold as well. Thus, $u \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$ follows. (ii) If $u \to_{\mathcal{R}} v$, then we distinguish two cases:

- (C) applies, i.e., $u = f(u_1, \ldots, u_i, \ldots, u_k)$ for some $f \in \mathcal{F}$ and terms $u_1, \ldots, u_k$; $u_i \to_{\mathcal{R}} v_i$ for some term v_i, and $v = f(u_1, \ldots, v_i, \ldots, v_k)$. By the I.H., $u_i \to_{\mathcal{R}[\{\alpha'\}]_\alpha} v_i$. Therefore, $u \to_{\mathcal{R}[\{\alpha'\}]_\alpha} v$, hence $u \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$.
- There is an application of (Rl) with a conditional rule which we can assume is α. Therefore, we have $u = \sigma(\ell)$ and $v = \sigma(r)$ for some substitution σ; and there are proof trees for $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ for all $s \to t \in c$. By the I.H., we also have $\sigma(s) \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} \sigma(t)$. Due to (1), (2), and (3), by Proposition 3, there is a substitution ϕ such that for all $x \in \mathcal{V}ar(s_i)$, $\sigma(x) \to^*_{\mathcal{R}} \phi(\theta(x))$ and for all $y \notin \mathcal{V}ar(s_i)$, $\sigma(y) = \phi(\theta(y))$. By the I.H., for all $x \in \mathcal{V}ar(s_i)$, $\sigma(x) \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} \phi(\theta(x))$. By (4), for all $s \to t \in c[\diamond]_i$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t) = \mathcal{V}ar(s_i) \cap \mathcal{V}ar(t_i) = \emptyset$; hence, $\sigma(s) = \phi(\theta(s))$ and $\sigma(t) = \phi(\theta(t))$. Therefore, for all $s \to t \in c[\diamond]_i$, we have $\sigma(s) = \phi(\theta(s)) \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} \phi(\theta(t)) = \sigma(t)$ and therefore $u = \sigma(\ell) \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} \phi(\theta(\ell)) \to_{\mathcal{R}[\{\alpha'\}]_\alpha} \phi(\theta(r))$. By (5), $\sigma(r) = \phi(\theta(r)) = v$. Thus, $u \to^+_{\mathcal{R}[\{\alpha'\}]_\alpha} v$.

□

The proof of the following result is analogous to that of Proposition 4.

Proposition 5 *Let $\mathcal{R}$ be a CTRS, $\alpha : \ell \to r \Leftarrow c \in \mathcal{R}$, and $s_i \to t_i \in c$ be such that $s_i =^?_\theta t_i$ and conditions (1) to (5) in Proposition 4 hold. If a term t is $\mathcal{R}[\{\alpha_{\theta,i}\}]_\alpha$-operationally terminating, then it is $\mathcal{R}$-operationally terminating.*

Proposition 6 *Let $\mathcal{R}$ be a CTRS, $\alpha : \ell \to r \Leftarrow c \in \mathcal{R}$, and $s_i \to t_i \in c$ be such that $s_i =^?_\theta t_i$. Let $\alpha' : \theta(\ell) \to \theta(r) \Leftarrow \theta(c[\diamond]_i)$. If $u \to_{\mathcal{R}[\{\alpha'\}]_\alpha} v$ (resp. $u \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$), then $u \to_{\mathcal{R}} v$ ($u \to^*_{\mathcal{R}} v$).*

Proof By induction on the depth N of the corresponding proof tree.

If $N = 0$, then (i) If $u \to_{\mathcal{R}[\{\alpha'\}]_\alpha} v$, then there is a single application of (Rl) where $u = \sigma(\ell')$ and $v = \sigma(r')$ for some *unconditional* rule $\ell' \to r' \in \mathcal{R}[\{\alpha'\}]_\alpha$. We consider two cases:

1. If $\ell' \to r'$ is *not* α', then $\ell' \to r' \in \mathcal{R}$ and we have $u \to_{\mathcal{R}} v$ as well.
2. If $\ell' \to r'$ is α' and α is $\ell \to r \Leftarrow s_1 \to t_1$. Then $\ell' = \theta(\ell)$ and $r' = \theta(r)$, and $\theta(s_1) = \theta(t_1)$. Therefore, $\sigma(\theta(s_1)) = \sigma(\theta(t_1))$ holds and $\sigma(\theta(s_1)) \to^* \sigma(\theta(t_1))$ can be proved in $\mathcal{R}$ using (Rf). Thus, $u = \sigma(\theta(\ell)) \to_{\mathcal{R}} \sigma(\theta(r)) = v$ holds as well.

(ii) If $u \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$, then there is a single application of (Rf), i.e., $u = v$ and therefore $u \to^*_{\mathcal{R}} v$ holds.

If $N > 0$, then we distinguish two cases. (i) If $u \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$, there is an application of (T), i.e., there is a term w such that $u \to_{\mathcal{R}[\{\alpha'\}]_\alpha} w$ and $w \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} v$ holds. Then, by the I.H., $u \to_{\mathcal{R}} w$ and $w \to^*_{\mathcal{R}} v$ hold as well. Thus, $u \to^*_{\mathcal{R}} v$ follows. (ii) If $u \to_{\mathcal{R}[\{\alpha'\}]_\alpha} v$, we also distinguish two cases:

1. (C) applies, i.e., $u = f(u_1, \ldots, u_i, \ldots, u_k)$ for some $f \in \mathcal{F}$ and terms $u_1, \ldots, u_k$ and we have $u_i \to_{\mathcal{R}[\{\alpha'\}]_\alpha} v_i$ for some term v_i, and $v = f(u_1, \ldots, v_i, \ldots, v_k)$. By the I.H., $u_i \to_{\mathcal{R}} v_i$. Therefore, we also have $u \to_{\mathcal{R}} v$.
2. There is an application of (Rl) with a conditional rule which, without loss of generality, we can assume is α'. Therefore, we have $u = \sigma(\theta(\ell))$ and $v = \sigma(\theta(r))$ for some substitution σ; and there are proof trees for $\sigma(\theta(s)) \to^*_{\mathcal{R}[\{\alpha'\}]_\alpha} \sigma(\theta(t))$ for all $s \to t \in c[\diamond]_i$. By the I.H., we also have $\sigma(\theta(s)) \to^*_{\mathcal{R}} \sigma(\theta(t))$. Since $\theta(s_i) = \theta(t_i)$, we also have $\sigma(\theta(s_i)) = \sigma(\theta(t_i))$, with a proof of $\sigma(\theta(s_i)) \to^*_{\mathcal{R}} \sigma(\theta(t_i))$ using (Rf). Hence, $u = \sigma(\theta(\ell)) \to_{\mathcal{R}} \sigma(\theta(r)) = v$ holds.

□

The proof of the following result is analogous to that of Proposition 6.

Proposition 7 *Let $\mathcal{R}$ be a CTRS, $\alpha : \ell \to r \Leftarrow c \in \mathcal{R}$, and $s_i \to t_i \in c$ be such that $s_i =^?_\theta t_i$. If a term t is $\mathcal{R}$-operationally terminating, then it is $\mathcal{R}[\{\alpha_{\theta,i}\}]_\alpha$-operationally terminating.*

Theorem 3 *Given i, if (1) s_i is linear, (2) $NRules(\mathcal{R}, s_i) = \emptyset$, (3) $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(r) = \emptyset$, (4) for all $s \to t \in c[\diamond]_i$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(s) = \emptyset$, and (5) for all $s \to t \in c$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t) = \emptyset$, then P_{SUC} is $\check{\tau}$-sound. P_{SUC} is a-complete; if $\alpha \notin \mathcal{R}$, or $\alpha \in \mathcal{R}_F$ and (1)–(5) hold, then it is $\check{\tau}$-complete. Therefore, $\Pi(\mathsf{P}_{SUC}) = \{\langle f, f\rangle, \langle f, \mathsf{a}\rangle, \langle \bullet, f\rangle, \langle \bullet, \mathsf{a}\rangle\}$.*

Proof In the following, α' denotes $\alpha_{\theta,i}$, $\mathcal{P}'$ denotes $\mathcal{P}[\{\alpha'\}]_\alpha$ and similarly for $\mathcal{Q}'$ and $\mathcal{R}'$. Regarding soundness, we consider two cases: first, assume $\alpha \in \mathcal{R}$. By Proposition 4, for terms u and v, if $u \to^*_{\mathcal{R}} v$, then $u \to^*_{\mathcal{R}'} v$. Thus, every $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain $A : (u^i \to v^i \Leftarrow c^i)_{i \geq 1}$ with substitution σ is also a $(\mathcal{P}, \mathcal{Q}, \mathcal{R}')$-O-chain. Then, the absence of infinite $(\mathcal{P}, \mathcal{Q}, \mathcal{R}')$-O-chains implies the absence of infinite $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chains. Thus, P_{SUC} is a-sound. Actually, P_{SUC} is sound because $\mathcal{R}$-operationally terminating terms are $\mathcal{R}'$-operationally terminating (Proposition 7).

Now, assume $\alpha \notin \mathcal{R}$; hence $\mathcal{R}[\{\alpha'\}]_\alpha = \mathcal{R}$. Assume that there is an infinite $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain $A : (u^i \to v^i \Leftarrow c^i)_{i \geq 1}$ for some substitution σ. Without loss of generality, we can assume that $\alpha \in \mathcal{P} \cup \mathcal{Q}$ is used in this chain (if not, then A is a $(\mathcal{P}', \mathcal{Q}', \mathcal{R})$-O-chain as well). We transform A into an infinite $(\mathcal{P}', \mathcal{Q}', \mathcal{R})$-O-chain A' that uses α' instead.

Note that $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ holds for all $s \to t \in c$. Due to (2) and (5), by Proposition 3 there is a substitution ϕ such that, for all $x \in \mathcal{V}ar(s_i)$, $\sigma(x) \to^*_{\mathcal{R}}$

$\phi(\theta(x))$ and for all $x \in \mathcal{V}ar(t_i)$, $\phi(\theta(x)) = \sigma(x)$. Define $\hat{\sigma}$ as follows: $\hat{\sigma}(x) = \phi(x)$ if $x \in \mathcal{V}ar(\theta(y))$ for some $y \in \mathcal{V}ar(s_i) \cup \mathcal{V}ar(t_i)$ and $\hat{\sigma}(x) = \sigma(x)$ otherwise. Then, for all $s \to t \in c[\diamond]_i$, we have:

- if $x \in \mathcal{V}ar(s)$, then (by (4)), $x \notin \mathcal{V}ar(s_i)$. Therefore, either $x \in \mathcal{V}ar(t_i)$, and hence $\hat{\sigma}(\theta(x)) = \phi(\theta(x)) = \sigma(x)$; or else $x \notin \mathcal{V}ar(s_i) \cup \mathcal{V}ar(t_i)$, and hence $\hat{\sigma}(x) = \sigma(x)$; furthermore, since we can assume $\theta(x) = x$ (being θ a most general unifier), we can even write $\hat{\sigma}(\theta(x)) = \sigma(x)$. In both cases, $\hat{\sigma}(\theta(s)) = \phi(\theta(s))) = \sigma(s)$.
- If $x \in \mathcal{V}ar(t)$, then (by (5)), $x \notin \mathcal{V}ar(s_i)$. We similarly conclude $\hat{\sigma}(\theta(t)) = \phi(\theta(t))) = \sigma(t)$.

Therefore, for all $s \to t \in c[\diamond]_i$, we have $\hat{\sigma}(\theta(s)) = \sigma(s) \to_{\mathcal{R}}^* \sigma(t) = \hat{\sigma}(\theta(t))$, i.e., the conditional part $\theta(c[\diamond]_i)$ of α' is satisfied when instantiated by $\hat{\sigma}$. Thus, the rewriting step $\hat{\sigma}(\theta(\ell)) \to \hat{\sigma}(\theta(r))$ (with either $\mathcal{P}$ or $\mathcal{Q}$) is possible. Furthermore, note that, since (by (3)) $\mathcal{V}ar(r) \cap \mathcal{V}ar(s_i) = \emptyset$, we have $\hat{\sigma}(\theta(r)) = \sigma(r)$. Note that, for every other rule used in A, we can assume that σ and $\hat{\sigma}$ realize exactly the *same* instantiations of variables. Therefore, the only *possible gaps* in the definition of A' obtained from A are the *connections* between a step with α' and the *previous* steps in the sequence. But, since we have $\sigma(x) \to_{\mathcal{R}}^* \hat{\sigma}(x)$ for all variables $x \in \mathcal{V}ar(t_i)$ and $\sigma(x) = \hat{\sigma}(x)$ otherwise, we conclude that such rewriting connections will be possible by just adding some additional rewritings with $\mathcal{R}$. Assume that the M-th step involves α in A for some $M > 1$. We have

$$\hat{\sigma}(v^{M-1}) = \sigma(v^{M-1})(\to_{\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}})^* \sigma(u^M) \to_{\mathcal{R}}^* \phi(\theta(u^M)) = \hat{\sigma}(\theta(u^M))$$

Now, notice that $\sigma(s) \to_{\mathcal{R}}^* \sigma(t)$ for all $s \to t \in c$. By (4) and (5), and by definition of $\hat{\sigma}$, we have $\hat{\sigma}(s) = \sigma(s) \to_{\mathcal{R}}^* \sigma(t) = \hat{\sigma}(t)$ as needed to allow the use of α'. By (3) we can write $\hat{\sigma}(\theta(v^M)) = \phi(\theta(v^M)) = \sigma(v^M)$ and hence

$$\hat{\sigma}(\theta(v^M)) = \sigma(v^M)(\to_{\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}})^* \sigma(u^{M+1}) = \hat{\sigma}(u^{M+1}),$$

i.e., we can continue the chain without any problem. Hence, P_{SUC} is $(\check{\tau}, \mathsf{a})$-sound. Actually, since $\alpha \notin \mathcal{R}$, it is $\check{\tau}$-sound.

Regarding *completeness*, assume that there is an infinite $(\mathcal{P}', \mathcal{Q}', \mathcal{R}')$-O-chain $A : (u^i \to v^i \Leftarrow c^i)_{i \geq 1}$ where for all $i \geq 1$, $\sigma(v^i)(\to_{\mathcal{R}'} \cup \overset{\Lambda}{\to}_{\mathcal{Q}'})^* \sigma(u^{i+1})$ for some substitution σ, and for all $s \to t \in c^i$, $\sigma(s) \to_{\mathcal{R}'}^* \sigma(t)$. Assume that α' is used in this chain. Then, we let $\hat{\sigma}(x) = \sigma(x)$ for each x that does not occur in (any occurrence of) this rule, and $\hat{\sigma}(x) = \sigma(\theta(x))$ otherwise. Since $\sigma(\theta(\ell)) \to \sigma(\theta(r))$ holds, we must have $\hat{\sigma}(s) = \sigma(\theta(s)) \to_{\mathcal{R}'}^* \sigma(\theta(t)) = \hat{\sigma}(t)$ for all $s \to t \in c[\diamond]_\alpha$. Therefore, we have $\hat{\sigma}(s) \to_{\mathcal{R}}^* \hat{\sigma}(t)$ (if $\alpha \notin \mathcal{R}$, then $\mathcal{R}' = \mathcal{R}$ and it is obvious; if $\alpha \in \mathcal{R}$, use Proposition 6). And, since $\theta(s_i) = \theta(t_i)$, we also have $\sigma(\theta(s_i)) = \sigma(\theta(t_i))$, i.e., $\hat{\sigma}(s_i) = \sigma(\theta(s_i)) \to_{\mathcal{R}}^* \sigma(\theta(t_i)) = \hat{\sigma}(t_i)$. Therefore, there is an infinite $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain as well (with substitution $\hat{\sigma}$). Thus, P_{SUC} is a-complete. If $\alpha \notin \mathcal{R}$, then $\mathcal{R}' = \mathcal{R}$ and operational termination of $\mathcal{R}'$ and $\mathcal{R}$ coincide, i.e., minimality is preserved as well, i.e., P_{SUC} is complete. If $\alpha \in \mathcal{R}$, then minimality of A as above is also preserved when A is viewed as a $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain because $\mathcal{R}'$-operationally terminating terms are also $\mathcal{R}$-operationally terminating (by Proposition 5, as (1)–(5) hold). Thus, P_{SUC} is complete. □

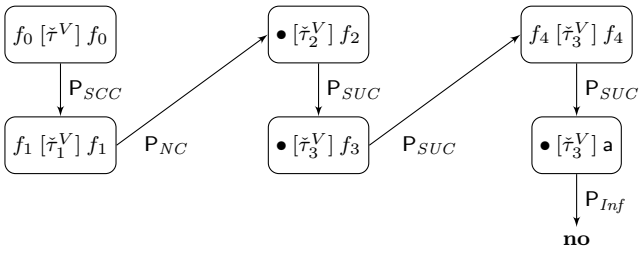


Fig. 4 Non-V-termination of $\mathcal{R}$ in Figure 1

Example 20 (Non-V-termination of $\mathcal{R}$ in Figure 1) We repeatedly apply P_{SUC} to pair (34) of $\check{\tau}_2^V = (\{(33),(34)\},\emptyset,\mathcal{R},f)$ in Example 17, i.e., to

$$\mathsf{DIV}(x,y) \to \mathsf{DIV}(x-y,y) \Leftarrow x \to x', y \to \mathsf{0}, \mathsf{true} \to \mathsf{true}$$

to remove the conditional part:

- Condition $x \to x'$ in (34) is removed with *mgu* $\theta_1 = \{x \mapsto x'\}$ to obtain

$$\mathsf{DIV}(x',y) \to \mathsf{DIV}(x'-y,y) \Leftarrow y \to \mathsf{0}, \mathsf{true} \to \mathsf{true} \tag{35}$$

  Therefore, $\mathsf{P}_{SUC}(\check{\tau}_2^H) = \{\check{\tau}_3^H\}$ where $\check{\tau}_3^H = (\{(33),(35)\},\emptyset,\mathcal{R},f)$. Since condition (3) in Theorem 3 does not hold, but (34) $\notin \mathcal{R}$, we use the plugging scheme $\langle\bullet,f\rangle$.
- Condition $y \to \mathsf{0}$ in (35) is removed with *mgu* $\theta_2 = \{y \mapsto \mathsf{0}\}$ to obtain

$$\mathsf{DIV}(x',\mathsf{0}) \to \mathsf{DIV}(x'-\mathsf{0},\mathsf{0}) \Leftarrow \mathsf{true} \to \mathsf{true} \tag{36}$$

  Now, $\mathsf{P}_{SUC}(\check{\tau}_3^H) = \{\check{\tau}_4^H\}$ where $\check{\tau}_4^H = (\{(33),(36)\},\emptyset,\mathcal{R},f)$. Again, we use the plugging scheme $\langle\bullet,f\rangle$.
- Finally, condition $\mathsf{true} \to \mathsf{true}$ in (36) is removed with $\theta_3 = \epsilon$ to obtain

$$\mathsf{DIV}(x',\mathsf{0}) \to \mathsf{DIV}(x'-\mathsf{0},\mathsf{0}) \tag{37}$$

  $\mathsf{P}_{SUC}(\check{\tau}_4^H) = \{\check{\tau}_5^H\}$ where $\check{\tau}_5^H = (\{(33),(37)\},\emptyset,\mathcal{R},f)$. Here, we use $\langle f,f\rangle$.

Now, we apply P_{Inf} in [27, Section 5.1] to (37) thus finishing the proof: since the right-hand side $\mathsf{DIV}(x'-\mathsf{0},\mathsf{0})$ of (37) is an instance of the left-hand side $\mathsf{DIV}(x',\mathsf{0})$ by substitution $\{x' \mapsto x'-\mathsf{0}\}$, we have $\mathsf{P}_{Inf}(\check{\tau}_5^H) = \mathsf{no}$. The OCTRSP-tree is depicted in Figure 4. According to Table 2, this proves $\mathcal{R}$ in Figure 1 non-V-terminating and hence operationally nonterminating.

Requiring $\mathit{NRules}(\mathcal{R},s_i) = \emptyset$, i.e., condition (2) in Theorem 3, is essential for soundness of P_{SUC}.

Example 21 Consider the CTRS

$$\mathsf{a} \to \mathsf{b}$$
$$\mathsf{c} \to \mathsf{d} \Leftarrow \mathsf{a} \to x, \mathsf{b} \to x$$

Note that a can be rewritten into b, thus satisfying the two conditions in the rule and enabling the rewriting step $\mathsf{c} \to \mathsf{d}$. However, with the rule

$$\mathsf{c} \to \mathsf{d} \Leftarrow \mathsf{b} \to \mathsf{a}$$

which is obtained by removing the condition $\mathsf{a} \to x$, and instantiating the remainder of the rule by $\theta(x) = \mathsf{a}$ the rewriting step $\mathsf{c} \to \mathsf{d}$ is no longer possible.

Requiring $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(r) = \emptyset$, i.e., condition (3) in Theorem 3, is also essential for soundness of P_{SUC}.

Example 22 Let $\mathcal{R}$ be given by

$$\begin{aligned} \mathsf{b} &\to \mathsf{a} \\ \mathsf{b} &\to \mathsf{f}(\mathsf{b}) \end{aligned}$$

and $\mathcal{P}$ consisting of a pair:

$$\mathsf{f}(x) \to x \Leftarrow x \to \mathsf{a} \tag{38}$$

There is an infinite $(\mathcal{P}, \emptyset, \mathcal{R})$-O-chain $\mathsf{f}(\mathsf{b}) \overset{\Lambda}{\to}_{\mathcal{P},\mathcal{R}} \underline{\mathsf{b}} \to_{\mathcal{R}} \mathsf{f}(\mathsf{b}) \overset{\Lambda}{\to}_{\mathcal{P},\mathcal{R}} \cdots$ where the $\overset{\Lambda}{\to}_{\mathcal{P},\mathcal{R}}$-steps are possible using the first rule of $\mathcal{R}$ to evaluate the reachability condition $\mathsf{b} \to^* \mathsf{a}$. Thus, $\check{\tau} = (\mathcal{P}, \emptyset, \mathcal{R}, \varphi)$ is infinite. The application of P_{SUC} transforms (38) into

$$\mathsf{f}(\mathsf{a}) \to \mathsf{a} \tag{39}$$

having no (finite or infinite) chain, i.e., $(\{(39)\}, \emptyset, \mathcal{R}, \varphi)$ is *finite*. Thus, P_{SUC} is not $\check{\tau}$-sound. Note that (1)–(5) in Theorem 3 all hold, except (3).

The transformation performed by P_{SUC} *preserves* deterministic 3-rules.

Proposition 8 *Let $\alpha : \ell \to r \Leftarrow c$ be a deterministic 3-rule and $s_i \to t_i \in c$ be such that $\theta(s_i) = \theta(t_i)$ for some substitution θ with $\mathcal{D}om(\theta) = \mathcal{V}ar(s_i) \cup \mathcal{V}ar(t_i)$. Then, $\alpha_{\theta,i} : \theta(\ell) \to \theta(r) \Leftarrow \theta(c[\diamond]_i)$ is a deterministic 3-rule.*

Proof First we prove that $\alpha_{\theta,i}$ is a 3-rule. We proceed by contradiction. If $\alpha_{\epsilon,i}$ is not a 3-rule, then there is $x \in \mathcal{V}ar(r)$ such that

$$\mathcal{V}ar(\theta(x)) \not\subseteq \mathcal{V}ar(\theta(\ell)) \cup \bigcup_{j \in \{1,\ldots,n\}-\{i\}} \mathcal{V}ar(\theta(s_j)) \cup \mathcal{V}ar(\theta(t_j)) \tag{40}$$

Since α is a 3-rule, $x \in \mathcal{V}ar(r) \subseteq \mathcal{V}ar(\ell) \cup \bigcup_{j=1}^{n} \mathcal{V}ar(s_j) \cup \mathcal{V}ar(t_j)$, i.e., $x \in \mathcal{V}ar(s_i) \cup \mathcal{V}ar(t_i)$ and, since α is deterministic, $\mathcal{V}ar(s_i) \subseteq \mathcal{V}ar(\ell) \cup \bigcup_{j=1}^{i-1} \mathcal{V}ar(t_j)$, so we actually have $x \in \mathcal{V}ar(t_i)$; otherwise, we would contradict (40). Since s_i and t_i unify, there is an *mgu* ϑ such that $\vartheta \leq \theta$. Since $\mathcal{V}ar(\vartheta(x)) \subseteq \mathcal{V}ar(s_i)$, it follows that $\mathcal{V}ar(\theta(x)) \subseteq \mathcal{V}ar(\theta(s_i)) \subseteq \mathcal{V}ar(\theta(\ell)) \cup \bigcup_{j=1}^{i-1} \mathcal{V}ar(\theta(t_j))$ contradicting (40). Thus, $\alpha_{\theta,i}$ is a 3-rule. Now we prove it deterministic. Otherwise, there is $k \in \{1, \ldots, n\} - \{i\}$ and $x \in \mathcal{V}ar(s_k)$ such that

$$\mathcal{V}ar(\theta(x)) \not\subseteq \mathcal{V}ar(\theta(\ell)) \cup \bigcup_{j \in \{1,\ldots,k-1\}-\{i\}} \mathcal{V}ar(\theta(t_j)) \tag{41}$$

If $k < i$, then (41) becomes $\mathcal{V}ar(\theta(x)) \not\subseteq \mathcal{V}ar(\theta(\ell)) \cup \bigcup_{j\in\{1,\ldots,k-1\}} \mathcal{V}ar(\theta(t_j))$, contradicting determinism of α. If $k > i$, we must have $x \in \mathcal{V}ar(t_i)$; otherwise determinism of α would be contradicted as well. As above, there is an *mgu* ϑ such that $\vartheta \leq \theta$. Hence, $\mathcal{V}ar(\vartheta(x)) \subseteq \mathcal{V}ar(s_i)$ and $\mathcal{V}ar(\theta(t_i)) \subseteq \mathcal{V}ar(\theta(s_i))$. By determinism of α,

$$\mathcal{V}ar(\theta(x)) \subseteq \mathcal{V}ar(\theta(s_i)) \subseteq \mathcal{V}ar(\theta(\ell)) \cup \bigcup_{j\in\{1,\ldots,i-1\}} \mathcal{V}ar(\theta(t_j))$$

contradicting (41) (because $k > i$). Therefore, $\alpha_{\theta,i}$ is deterministic. □

Our next processor, P_{SUNC}, combines P_{NC} and P_{SUC}, but avoids the non-unification requirement of P_{NC} and the non-narrowability requirement of P_{SUC} (item (2) in Theorem 3). Thus, it is actually incomparable with both of them.

Definition 5 (Simplification and narrowing) Let $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an OCTRS problem, $\alpha : u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$, $s_i \to t_i \in c$, $\mathcal{N} \subseteq \overline{\mathcal{N}}(\mathcal{R}, \alpha, i)$ finite, and $U = \{\alpha_{\theta,i}\}$ if $s_i =^?_\theta t_i$ and $U = \emptyset$ otherwise. P_{SUNC} is given by

$$\mathsf{P}_{SUNC}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) = \{(\mathcal{P}[\mathcal{N} \cup U]_\alpha, \mathcal{Q}[\mathcal{N} \cup U]_\alpha, \mathcal{R}, \varphi)\}$$

Theorem 4 *P_{SUNC} is complete. If $\mathcal{N}(\mathcal{R}, \alpha, i) = \overline{\mathcal{N}}(\mathcal{S}, \alpha, i)$ and either s_i is ground or (1) NRules$(\mathcal{R}, s_i)$ is a TRS, (2) s_i is linear, (3) $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t_i) = \emptyset$, (4) for all $s \to t \in c[\diamond]_i$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(s) = \emptyset$, and (5) for all $s \to t \in c$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t) = \emptyset$, and (6) $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(v) = \emptyset$, then P_{SUNC} is $\check{\tau}$-sound. Therefore, $\Pi(\mathsf{P}_{SUNC}) = \{\langle f, f\rangle, \langle \bullet, f\rangle\}$.*

Example 23 Consider the following CTRS $\mathcal{R}$ [30, Example 17]:

$$\mathsf{a} \to \mathsf{h}(\mathsf{b}) \quad (42) \qquad\qquad \mathsf{f}(x) \to y \Leftarrow \mathsf{a} \to \mathsf{h}(y) \quad (44)$$

$$\mathsf{a} \to \mathsf{h}(\mathsf{c}) \quad (43) \qquad\qquad \mathsf{g}(x, \mathsf{b}) \to \mathsf{g}(\mathsf{f}(\mathsf{c}), x) \Leftarrow \mathsf{f}(\mathsf{b}) \to x, x \to \mathsf{c} \quad (45)$$

We have:

$$\mathsf{DP}_H(\mathcal{R}): \qquad \mathsf{G}(x, \mathsf{b}) \to \mathsf{G}(\mathsf{f}(\mathsf{c}), x) \Leftarrow \mathsf{f}(\mathsf{b}) \to x, x \to \mathsf{c} \quad (46)$$

$$\mathsf{G}(x, \mathsf{b}) \to \mathsf{F}(\mathsf{c}) \Leftarrow \mathsf{f}(\mathsf{b}) \to x, x \to \mathsf{c} \quad (47)$$

$$\mathsf{DP}_V(\mathcal{R}): \qquad \mathsf{F}(x) \to \mathsf{A} \quad (48)$$

$$\mathsf{G}(x, \mathsf{b}) \to \mathsf{F}(\mathsf{b}) \quad (49)$$

and $\mathsf{DP}_{VH}(\mathcal{R}) = \emptyset$. Since $\mathsf{P}_{SCC}(\check{\tau}^V) = \emptyset$, $\check{\tau}^V$ is finite. We have $\mathsf{P}_{SCC}(\check{\tau}^H) = \{\check{\tau}_1^H\}$, where $\check{\tau}_1^H = \{(\{(46)\}, \emptyset, \mathcal{R}, \varphi)\}$. Now, we apply P_{SUNC} to $\check{\tau}_1^H$ using condition $\mathsf{f}(\mathsf{b}) \to x$ in (46). We have two one-step narrowings on $\mathsf{f}(\mathsf{b})$ only: (i) $\mathsf{f}(\mathsf{b}) \rightsquigarrow_{(44),\theta_1} \mathsf{b}$, where $\theta_1 = \{y' \mapsto \mathsf{b}\}$ (with y' the renaming of variable y occurring in (44)), and (ii) $\mathsf{f}(\mathsf{b}) \rightsquigarrow_{(44),\theta_2} \mathsf{c}$ where $\theta_2 = \{y' \mapsto \mathsf{c}\}$. Thus, $N_1(\mathcal{R}, s) = \{(\mathsf{b}, \epsilon), (\mathsf{c}, \epsilon)\}$ and $\overline{\mathcal{N}}(\mathcal{R}, (46), 1) = \{(50), (51)\}$, where

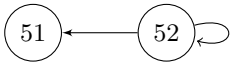
$$\mathsf{G}(x, \mathsf{b}) \to \mathsf{G}(\mathsf{f}(\mathsf{c}), x) \Leftarrow \mathsf{b} \to x, x \to \mathsf{c} \quad (50)$$

$$\mathsf{G}(x, \mathsf{b}) \to \mathsf{G}(\mathsf{f}(\mathsf{c}), x) \Leftarrow \mathsf{c} \to x, x \to \mathsf{c} \quad (51)$$

On the other hand, since $\mathsf{f}(\mathsf{b}) =^?_{\theta_3} x$ with $\theta_3 = \{x \mapsto \mathsf{f}(\mathsf{b})\}$, we have $\mathsf{P}_{SUNC}(\check{\tau}_1^H) = \{\check{\tau}_2^H\}$ where $\check{\tau}_2^H = (\{(50), (51), (52)\}, \emptyset, \mathcal{R}, f)$, with

$$\mathsf{G}(\mathsf{f}(\mathsf{b}), \mathsf{b}) \to \mathsf{G}(\mathsf{f}(\mathsf{c}), \mathsf{f}(\mathsf{b})) \Leftarrow \mathsf{f}(\mathsf{b}) \to \mathsf{c} \quad (52)$$

According to Theorem 4, we can use the plugging scheme $\langle f, f\rangle$. Now, the application of P_{SCC} to $\check{\tau}_2^H$ discards (50), which is clearly $\mathcal{R}$-infeasible (as both b and c are irreducible) and shows that the estimated graph is as follows:



Thus, $\mathsf{P}_{SCC}(\check{\tau}_2^H) = \{\check{\tau}_3^H\}$ where $\check{\tau}_3^H = (\{(52)\}, \emptyset, \mathcal{R}, f)$.[7]

Using P_{NC} or P_{SUC} instead of P_{SUNC} in the OCTRSP-tree does *not* lead to a proof of finiteness of $\check{\tau}^H$: since $\mathsf{f}(\mathsf{b})$ and x unify, according to Theorem 2, the only plugging scheme that P_{NC} follows with $\check{\tau}^H$ is $\langle \bullet, f\rangle$, which disallows any operational termination proof due to $\bullet$ in the soundness component of the plugging scheme. Similarly, due to the narrowability of $\mathsf{f}(\mathsf{b})$, the only plugging schemes that P_{SUC} follows with $\check{\tau}^H$ are $\langle \bullet, f\rangle$ and $\langle \bullet, \mathsf{a}\rangle$, again disallowing any operational termination proof.

4.3 A Semantic Version of the Removal Triple Processor

A *removal triple* $(\gtrsim, \succeq, \sqsupset)$ consists of relations $\gtrsim, \succeq, \sqsupset$ on terms such that (i) $\sqsupset$ is well-founded, (ii) $\gtrsim \circ \sqsupset \subseteq \sqsupset$, and (iii) $\succeq \circ \sqsupset \subseteq \sqsupset$ [27, Definition 68]. The following definition slightly generalizes [27, Definition 69].

Definition 6 Let $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an OCTRS problem, and $\mathcal{S}$ be a CTRS. A removal triple $(\gtrsim, \succeq, \sqsupset)$ is $(\check{\tau}, \mathcal{S})$-*compatible* iff for all terms s, t,

1. if $s \to_{\mathcal{S},\mathcal{R}} t$, then $s \gtrsim t$ and
2. if $s \overset{\Lambda}{\to}_{\mathcal{P}\cup\mathcal{Q},\mathcal{R}} t$, then $s \bowtie t$ holds for some $\bowtie \in \{\gtrsim, \succeq, \sqsupset\}$.

Since $\to_{\mathcal{R},\mathcal{R}}$ and $\to_{\mathcal{R}}$ coincide, compatibility in the sense of [27, Definition 69] is $(\check{\tau}, \mathcal{R})$-compatibility whenever $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$, i.e., $\mathcal{S} = \mathcal{R}$ in Definition 6. In this section we only use this case; in Section 4.4 (see Definition 11) we will be using the general case $\mathcal{S} \neq \mathcal{R}$. Removal triples are used to simplify OCTRS problems $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ by removing rules from $\mathcal{P}$ and $\mathcal{Q}$.

Definition 7 (Removal triple processor) Let $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an OCTRS problem and $(\gtrsim, \succeq, \sqsupset)$ be a removal triple which is $(\check{\tau}, \mathcal{R})$-compatible. Let $\alpha : u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$. Then, P_{RT} is given by $\mathsf{P}_{RT}(\check{\tau}) = \{(\mathcal{P}[\emptyset]_\alpha, \mathcal{Q}[\emptyset]_\alpha, \mathcal{R}, \varphi)\}$ iff

$$\text{for all substitutions } \sigma, \text{ if } \sigma(s) \to_{\mathcal{R}}^* \sigma(t) \text{ for all } s \to t \in c, \text{ then } \sigma(u) \sqsupset \sigma(v). \quad (53)$$

In order to use P_{RT}, we need relations $\gtrsim$, $\succeq$, and $\sqsupset$ on terms satisfying conditions (i)–(iii) above (i.e., qualifying $(\gtrsim, \succeq, \sqsupset)$ as a removal triple), and also fulfilling the requirements in Definition 6. Finally, there is a specific condition (53) for the application of the processor to remove a rule from $\mathcal{P}$ or $\mathcal{Q}$.

In [22] we pointed to the use of *logical models* as an appropriate way to deal with the aforementioned problems when dealing with CTRSs. In [18] some work

[7] Actually, the cycle could also be dismissed by using the satisfiability approach in [20, Section 4.5]. This would immediately lead to an operational termination proof. However, we use the simpler (syntactic) approximation described in [27, Section 7.1.1] and delay the final proof to provide an application of our next processor in Section 4.3.

developing the practical use of the idea has been presented. Following this approach we express the conditions for the application of a given processor P to an OCTRS problem $\check{\tau}$ as a (many-sorted) *theory* $\mathcal{S}^{RT}_{\check{\tau},\alpha}$ and a first-order sentence $\phi^{\sqsupset}_{\alpha}$ representing condition (53) so that finding a *model* of $\mathcal{S}^{RT}_{\check{\tau},\alpha} \cup \{\phi^{\sqsupset}_{\alpha}\}$ implies that the processor can be applied to remove α.

For this purpose, we use a *two-sorted* signature with set of sorts $S_{DP} = \{\mathrm{S}, \mathrm{P}\}$ when dealing with dependency pairs [9, Section 5]. The main point is reinforcing the distinction between *root* reduction steps issued with *pairs* $u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$ and other reduction steps (at any depth) with rules $\ell \to r \Leftarrow d$ by using *sorts*. Root symbols in u and v are of sort P; more precisely, such symbols take terms of sort S as arguments, but 'return' an expression of sort P, i.e., they have a *rank* $\mathrm{S}\cdots\mathrm{S} \to \mathrm{P}$. Any other function symbol is considered to have a 'normal' rank $\mathrm{S}\cdots\mathrm{S} \to \mathrm{S}$. In the following section, we briefly introduce the basics of first-order logic with sorts needed to formalize our treatment.

4.3.1 Many-sorted signatures with predicates

In the following, given a set of *sorts* S, a *many-sorted signature* (S, Σ) is an $S^* \times S$-indexed family of sets $\Sigma = \{\Sigma_{w,s}\}_{(w,s)\in S^*\times S}$ containing *function symbols* with a given string of argument sorts and a result sort. If $f \in \Sigma_{s_1\cdots s_n,s}$, we often write $f : s_1 \cdots s_n \to s$ (a *rank* declaration for symbol f). Symbols f can be *overloaded*, i.e., they can have several rank declarations. Constant symbols, however, have only one rank declaration $c : \lambda \to s$ (where λ denotes the *empty* sequence). Given an S-sorted set $\mathcal{X} = \{\mathcal{X}_s \mid s \in S\}$ of *mutually disjoint* sets of variables (which are also disjoint from the signature Σ), the set $\mathcal{T}_\Sigma(\mathcal{X})_s$ of terms of sort s is the least set such that (i) $\mathcal{X}_s \subseteq \mathcal{T}_\Sigma(\mathcal{X})_s$; and (ii) for each $f : s_1 \dots s_n \to s$ and $t_i \in \mathcal{T}_\Sigma(\mathcal{X})_{s_i}$, $1 \le i \le n$, $f(t_1, \dots, t_n) \in \mathcal{T}_\Sigma(\mathcal{X})_s$. If $\mathcal{X} = \emptyset$, we write $\mathcal{T}_\Sigma$ rather than $\mathcal{T}_\Sigma(\emptyset)$ for the set of *ground* terms.

A *many-sorted signature with predicates* or just *signature* in the following [13] is a triple $\Omega = (S, \Sigma, \Pi)$ where (S, Σ) is a many-sorted signature of *function symbols* and $\Pi = \{\Pi_w \mid w \in S^*\}$ is a family of ranked *predicate symbols*. We write $P : w$ for $P \in \Pi_w$. Overloading is also allowed on predicates. The formulas φ of a many-sorted signature with predicates Ω are built up from atoms $P(t_1, \dots, t_n)$ with $P \in \Pi_w$ for some $w = s_1 \cdots s_n \in S^*$ and $t_i \in \mathcal{T}_\Sigma(\mathcal{X})_{s_i}$, $1 \le i \le n$, logic connectives ($\wedge$, $\neg$, ...) and quantifiers ($\forall, \exists$) as usual.

A structure $\mathcal{A}$ for $\Omega = (S, \Sigma, \Pi)$ is an S-sorted family of sets $\mathcal{A} = \{\mathcal{A}_s \mid s \in S\}$ together with a rank preserving interpretation of the function and predicate symbols of the language so that each function symbol $f : w \to s$ is interpreted as a mapping $f^{\mathcal{A}} : \mathcal{A}_w \to \mathcal{A}_s$, where $\mathcal{A}_w = \mathcal{A}_{s_1} \times \cdots \times \mathcal{A}_{s_n}$ whenever $w = s_1 \cdots s_n \in S^*$; and $P : w$ is interpreted as $P^{\mathcal{A}} \subseteq \mathcal{A}_w$. Then, interpretation of first-order formulas with respect to the structure is defined as usual. A model for a set $\mathcal{S}$ of first-order sentences (i.e., formulas whose variables are all *quantified*) is a structure $\mathcal{A}$ that makes them all true, written $\mathcal{A} \models \mathcal{S}$ (see [4,16]).

4.3.2 Implementing the use of P_{RT} as a satisfiability problem

Following the ideas in [18], given an OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ and a rule $\alpha \in \mathcal{P} \cup \mathcal{Q}$, the signature $\Omega^{\check{\tau}} = (S_{DP}, \Sigma^{\check{\tau}}, \Pi^{\check{\tau}})$ consists of the function symbols

$\Sigma^{\check{\tau}}$ in the signatures of $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ with the following *given* ranks: symbols f occurring at the *root* of u or v in $u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$ have rank $\mathrm{S} \cdots \mathrm{S} \to \mathrm{P}$; any other symbols f have rank $\mathrm{S} \cdots \mathrm{S} \to \mathrm{S}$.

We have two sets $\mathit{\Pi}^{\check{\tau}}_{\mathrm{PP}}$ and $\mathit{\Pi}^{\check{\tau}}_{\mathrm{SS}}$ of predicate symbols:

$$\mathit{\Pi}^{\check{\tau}}_{\mathrm{PP}} = \{\to_{\mathcal{R}}, \to_{\mathcal{P}}, \to_{\mathcal{Q}}, \pi_{\gtrsim}, \pi_{\succeq}, \pi_{\sqsupset}\} \text{ and } \mathit{\Pi}^{\check{\tau}}_{\mathrm{SS}} = \{\to_{\mathcal{R}}, \to^*_{\mathcal{R}}\}.$$

Roughly speaking, the predicate symbols play the following roles:

- $\pi_{\gtrsim}$, $\pi_{\succeq}$ and $\pi_{\sqsupset}$ correspond to the components of the removal triple.
- $\to_{\mathcal{R}}$ and $\to^*_{\mathcal{R}}$ represent one-step and many-step rewriting relations on terms by using the CTRS $\mathcal{R}$ in the considered OCTRS problem. Note that $\to_{\mathcal{R}}$ is *overloaded* to rewrite terms of sorts P and S.
- $\to_{\mathcal{P}}$ and $\to_{\mathcal{Q}}$ represent $\overset{\Lambda}{\to}_{\mathcal{P},\mathcal{R}}$- and $\overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}}$-rewriting, respectively. Thus, the rewriting step is performed (at the *root*) by using ℓ and r from a rule $\ell \to r \Leftarrow c$ in $\mathcal{P}$ (resp. $\mathcal{Q}$), but the *conditional part* c of the rule is evaluated using $\mathcal{R}$ (in both cases).

Now, we define the aforementioned theory $\mathcal{S}^{RT}_{\check{\tau},\alpha}$ as a union of (sub)theories:

$$\mathcal{S}^{RT}_{\check{\tau},\alpha} = \mathcal{S}^{RT}_{Rel,\alpha} \cup \mathcal{S}^{RT}_{CRel,\alpha} \cup \mathcal{S}^{RT}_{\mathcal{R}} \cup \mathcal{S}^{RT}_{\mathcal{P},\alpha} \cup \mathcal{S}^{RT}_{\mathcal{Q},\alpha}$$

where

1. $\mathcal{S}^{RT}_{Rel,\alpha}$ contains the sentences describing the properties to be fulfilled by the components of a removal triple:

$$(\forall x, y, z : \mathrm{P})\ x\ \pi_{\gtrsim}\ y \wedge y\ \pi_{\sqsupset}\ z \Rightarrow x\ \pi_{\sqsupset}\ z \tag{54}$$
$$(\forall x, y, z : \mathrm{P})\ x\ \pi_{\succeq}\ y \wedge y\ \pi_{\sqsupset}\ z \Rightarrow x\ \pi_{\sqsupset}\ z \tag{55}$$

   If $(\mathcal{P} \cup \mathcal{Q}) - \{\alpha\}$ is empty, then (55) is *omitted* and $\pi_{\succeq}$ is removed from $\mathit{\Pi}^{\check{\tau}}$.
2. $\mathcal{S}^{RT}_{CRel,\alpha}$ contains the sentences describing the *compatibility* between a removal triple and the open CTRS problem $\check{\tau}$ (Definition 6):

$$(\forall x, y : \mathrm{P})\ x \to_{\mathcal{R}} y \Rightarrow x\ \pi_{\gtrsim}\ y \tag{56}$$
$$(\forall x, y : \mathrm{P})\ x \to_{\mathcal{P}} y \Rightarrow x\ \pi_{\gtrsim}\ y \vee x\ \pi_{\succeq}\ y \vee x\ \pi_{\sqsupset}\ y \tag{57}$$
$$(\forall x, y : \mathrm{P})\ x \to_{\mathcal{Q}} y \Rightarrow x\ \pi_{\gtrsim}\ y \vee x\ \pi_{\succeq}\ y \vee x\ \pi_{\sqsupset}\ y \tag{58}$$

   If $\mathcal{P} - \{\alpha\} = \emptyset$ (resp. $\mathcal{Q} - \{\alpha\} = \emptyset$), then (57) (resp. (58)) can be *omitted* and $\to_{\mathcal{P}}$ (resp. $\to_{\mathcal{Q}}$) is removed from $\mathit{\Pi}^{\check{\tau}}$.
3. $\mathcal{S}^{RT}_{\mathcal{R}}$ describes one-step and many step rewritings with $\mathcal{R}$ according to Figure 2. $\mathcal{S}^{RT}_{\mathcal{R}}$ contains the formulas

$$(\forall x : \mathrm{S})\ x \to^*_{\mathcal{R}} x \tag{59}$$
$$(\forall x, y, z : \mathrm{S})\ x \to_{\mathcal{R}} y \wedge y \to^*_{\mathcal{R}} z \Rightarrow x \to^*_{\mathcal{R}} z \tag{60}$$

   corresponding to rules (Rf), i.e., *reflexivity*, and (T), i.e., *transitivity*. Now, provided that $\mathcal{R}$ is not empty, we add formulas $(C)_{f,i}$, i.e.,

$$(\forall \mathbf{x}, y_i : \mathrm{S})\ x_i \to_{\mathcal{R}} y_i \Rightarrow f(x_1, \ldots, x_i, \ldots, x_k) \to_{\mathcal{R}} f(x_1, \ldots, y_i, \ldots, x_k) \tag{61}$$

for each k ary symbol $f \in \Sigma^{\check{\tau}}_{\mathrm{S}\cdots\mathrm{S},\mathrm{P}} \cup \Sigma^{\check{\tau}}_{\mathrm{S}\cdots\mathrm{S},\mathrm{S}}$ with $\mathbf{x} = x_1, \ldots, x_k$ and i, $1 \leq i \leq k$, and a formula $(Rf)_\beta$

$$(\forall \mathbf{x} : \mathrm{S})\ s_1 \to^*_{\mathcal{R}} t_1 \wedge \cdots \wedge s_n \to^*_{\mathcal{R}} t_n \Rightarrow \ell \to_{\mathcal{R}} r \tag{62}$$

for each rule $\beta : \ell \to r \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n \in \mathcal{R}$, where $\mathbf{x}$ contains all variables occurring in the rule.

4. $\mathcal{S}^{RT}_{\mathcal{P},\alpha}$ describes reductions with $\overset{\Lambda}{\to}_{\mathcal{P},\mathcal{R}}$. $\mathcal{S}^{RT}_{\mathcal{P},\alpha}$ contains a formula

$$(\forall \mathbf{x} : \mathrm{S})\ s_1 \to^*_{\mathcal{R}} t_1 \wedge \cdots \wedge s_n \to^*_{\mathcal{R}} t_n \Rightarrow u \to_{\mathcal{P}} v \tag{63}$$

for each rule $u \to v \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n \in \mathcal{P} - \{\alpha\}$, where $\mathbf{x}$ contains all variables occurring in the rule.

5. $\mathcal{S}^{RT}_{\mathcal{Q},\alpha}$ describes reductions with $\overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}}$. $\mathcal{S}^{RT}_{\mathcal{Q},\alpha}$ contains a formula

$$(\forall \mathbf{x} : \mathrm{S})\ s_1 \to^*_{\mathcal{R}} t_1 \wedge \cdots \wedge s_n \to^*_{\mathcal{R}} t_n \Rightarrow u \to_{\mathcal{Q}} v \tag{64}$$

for each rule $u \to v \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n \in \mathcal{Q} - \{\alpha\}$, where $\mathbf{x}$ contains all variables occurring in the rule.

Example 24 In preparation to use P_{RT} to remove pair (52), i.e.,

$$\mathsf{G}(\mathsf{f}(\mathsf{b}), \mathsf{b}) \to \mathsf{G}(\mathsf{f}(\mathsf{c}), \mathsf{f}(\mathsf{b})) \Leftarrow \mathsf{f}(\mathsf{b}) \to \mathsf{c}$$

from $\tau^H_3 = (\{(52)\}, \emptyset, \mathcal{R}, f)$ in Example 23, we define $\mathcal{S}^{RT}_{\check{\tau}^H_3,(52)}$ as follows:

1. $\mathcal{S}^{RT}_{Rel,(52)} = \{(54)\}$ and $\pi_{\succeq}$ is not added to the signature.
2. $\mathcal{S}^{RT}_{CRel,(52)} = \{(56)\}$; predicates $\to_{\mathcal{P}}$ and $\to_{\mathcal{Q}}$ are not included in the signature.
3. $\mathcal{S}^{RT}_{\mathcal{R}}$ contains the formulas (59) and (60). Then, formulas (61) for each k-ary function f and $1 \leq i \leq k$:

$$\begin{aligned}
(\forall x_1, y_1 : \mathrm{S})\ x_1 \to_{\mathcal{R}} y_1 &\Rightarrow \mathsf{f}(x_1) \to_{\mathcal{R}} \mathsf{f}(y_1)\\
(\forall x_1, y_1 : \mathrm{S})\ x_1 \to_{\mathcal{R}} y_1 &\Rightarrow \mathsf{h}(x_1) \to_{\mathcal{R}} \mathsf{h}(y_1)\\
(\forall x_1, x_2, y_1 : \mathrm{S})\ x_1 \to_{\mathcal{R}} y_1 &\Rightarrow \mathsf{g}(x_1, x_2) \to_{\mathcal{R}} \mathsf{g}(y_1, x_2)\\
(\forall x_1, x_2, y_2 : \mathrm{S})\ x_2 \to_{\mathcal{R}} y_2 &\Rightarrow \mathsf{g}(x_1, x_2) \to_{\mathcal{R}} \mathsf{g}(x_1, y_2)\\
(\forall x_1, x_2, y_1 : \mathrm{S})\ x_1 \to_{\mathcal{R}} y_1 &\Rightarrow \mathsf{G}(x_1, x_2) \to_{\mathcal{R}} \mathsf{G}(y_1, x_2)\\
(\forall x_1, x_2, y_2 : \mathrm{S})\ x_2 \to_{\mathcal{R}} y_2 &\Rightarrow \mathsf{G}(x_1, x_2) \to_{\mathcal{R}} \mathsf{G}(x_1, y_2)
\end{aligned}$$

Note the *overloaded* versions of $\to_{\mathcal{R}}$ (with S in the first sentence and P in the last two sentences). Finally, formulas (62) for the rules in $\mathcal{R}$:

$$\begin{aligned}
\mathsf{a} &\to_{\mathcal{R}} \mathsf{h}(\mathsf{b})\\
\mathsf{a} &\to_{\mathcal{R}} \mathsf{h}(\mathsf{c})\\
(\forall x, y : \mathrm{S})\ \mathsf{a} \to^*_{\mathcal{R}} \mathsf{h}(y) &\Rightarrow \mathsf{f}(x) \to_{\mathcal{R}} y\\
(\forall x : \mathrm{S})\ \mathsf{f}(\mathsf{b}) \to^*_{\mathcal{R}} x \wedge x \to^*_{\mathcal{R}} \mathsf{c} &\Rightarrow \mathsf{g}(x, \mathsf{b}) \to_{\mathcal{R}} \mathsf{g}(\mathsf{f}(\mathsf{c}), x)
\end{aligned}$$

4. Since (52) is the only pair the component $\mathcal{P}$ of $\check{\tau}^H_3$, $\mathcal{S}^{RT}_{\mathcal{P},(52)}$ is empty. Also, $\mathcal{S}^{RT}_{\mathcal{Q},(52)}$ is empty.

Definition 8 (Semantic version of P_{RT}) Let $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an open CTRS problem, $\mathcal{A}$ be an $\Omega_{\check{\tau}}$-structure with $\mathcal{A}_{\mathrm{P}}$ nonempty, $\alpha : u \to v \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n \in \mathcal{P} \cup \mathcal{Q}$, and $\phi_\alpha^{\sqsupset}$ be $(\forall \mathbf{x} : \mathrm{S})\bigwedge_{i=1}^{n} s_i \to_{\mathcal{R}}^{*} t_i \Rightarrow u\ \pi_{\sqsupset}\ v$, where $\mathbf{x}$ lists all variables in α. Then, $\mathsf{P}_{RT}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) = \{(\mathcal{P}[\emptyset]_\alpha, \mathcal{Q}[\emptyset]_\alpha, \mathcal{R}, \varphi)\}$ if $\mathcal{A} \models \mathcal{S}_{\check{\tau},\alpha}^{RT} \cup \{\phi_\alpha^{\sqsupset}\}$ and $\pi_{\sqsupset}^{\mathcal{A}}$ is well-founded on $\mathcal{A}_{\mathrm{P}}$.

Definition 8 transforms the application of P_{RT} to $\check{\tau}$ into the problem of *finding a model* $\mathcal{A}$ of $\mathcal{S}_{\check{\tau},\alpha}^{RT} \cup \{\phi_\alpha^{\sqsupset}\}$ such that the interpretation $\sqsupset^{\mathcal{A}}$ of $\sqsupset$ is well-founded on $\mathcal{A}_{\mathrm{P}}$. Such models can be obtained from model generators like AGES or Mace4[8]. The examples in this paper are treated with AGES, which is able to generate models of many-sorted theories (Mace4 does not support sorts). Sort, function, and predicate symbols are interpreted as *parametric* domains, functions and predicates. Furthermore, binary predicates can be required to be interpreted by a *well-founded* relation. Sentences in the theory are then transformed into *constraints* over the parameters, which are solved by using standard constraint solving methods and tools (based on SAT, SMT, etc.) [19].

Example 25 (cont. Example 24) In order to remove pair (52) from $\check{\tau}_3^H$ in Example 23, we seek a model $\mathcal{A}$ of

$$\mathcal{S}_{\check{\tau}_3^H,(52)}^{RT} \cup \{\mathsf{f}(\mathsf{b}) \to_{\mathcal{R}}^{*} \mathsf{c} \Rightarrow \mathsf{G}(\mathsf{f}(\mathsf{b}), \mathsf{b})\ \pi_{\sqsupset}\ \mathsf{G}(\mathsf{f}(\mathsf{c}), \mathsf{f}(\mathsf{b}))\} \tag{65}$$

so that $\pi_{\sqsupset}^{\mathcal{A}}$ is well-founded on $\mathcal{A}_{\mathrm{P}}$. With AGES, we obtain a structure $\mathcal{A}$ with domains: $\mathcal{A}_{\mathrm{P}} = \mathbb{N} - \{0\}$ and $\mathcal{A}_{\mathrm{S}} = \{-1, 0, 1\}$ (see Appendix A for a complete description of the AGES encoding). Symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = -1 \qquad \mathsf{b}^{\mathcal{A}} = -1 \qquad \mathsf{c}^{\mathcal{A}} = 0$$
$$\mathsf{f}^{\mathcal{A}}(x) = -x \qquad \mathsf{g}^{\mathcal{A}}(x, y) = 0 \qquad \mathsf{h}^{\mathcal{A}}(x) = 0 \qquad \mathsf{G}^{\mathcal{A}}(x, y) = 5x + 6$$

$$x(\to_{\mathcal{R}})_{\mathrm{P}}^{\mathcal{A}} y \Leftrightarrow x \geq y \qquad x(\to_{\mathcal{R}})_{\mathrm{S}}^{\mathcal{A}} y \Leftrightarrow \mathit{true} \qquad x(\to_{\mathcal{R}}^{*})_{\mathrm{S}}^{\mathcal{A}} y \Leftrightarrow \mathit{true}$$

$$x\ \pi_{\gtrsim}^{\mathcal{A}}\ y \Leftrightarrow x \geq y \qquad x\ \pi_{\sqsupset}^{\mathcal{A}}\ y \Leftrightarrow x > y$$

Note that $\pi_{\sqsupset}^{\mathcal{A}}$ is well-founded on $\mathcal{A}_{\mathrm{P}}$, as required.

Remark 5 (Trivial interpretations) Note that relations $\to_{\mathcal{R}}$ and $\to_{\mathcal{R}}^{*}$ on sort S are both interpreted as *true*. This is not surprising: every definite Horn theory[9] admits a trivial model $\mathcal{A}$ where all predicates $P \in \Pi_w$ are interpreted as $\mathcal{A}_w$, i.e., the cartesian product of the domains $\mathcal{A}_{s_i}$ interpreting each sort s_i in w. This is denoted in AGES by giving $P_w^{\mathcal{A}}$ the truth value *true*. Note that (65) is indeed a definite Horn theory. Thus, the only extra requirement which disallows a trivial model $\mathcal{A}$ is well-foundedness of $\pi_{\sqsupset}^{\mathcal{A}}$ on $\mathcal{A}_{\mathrm{P}}$. Clearly, $\pi_{\sqsupset}^{\mathcal{A}}$ cannot then be interpreted as *true*. Thus, it is not surprising to obtain a structure $\mathcal{A}$ which is loose enough as to give $\pi_{\sqsupset}$ the required (well-founded) interpretation, whereas the interpretation of other relations remain close to *true*.

So, finally, $\mathsf{P}_{RT}(\check{\tau}_3^H) = \{\check{\tau}_4^H\}$ where $\check{\tau}_4^H = (\emptyset, \emptyset, \mathcal{R}, f)$, thus proving $\mathcal{R}$ operationally terminating (Figure 5).

[8] See [18, Section 5.5.1] for details about its practical use in proofs of termination by satisfiability.

[9] By a definite Horn theory we mean a set of universally quantified clauses $\neg A_1 \vee \cdots \vee \neg A_n \vee B$ (or $A_1 \wedge \cdots \wedge A_n \Rightarrow B$ in implication form), where $A_1, \ldots, A_n, B$ are atoms for some $n \geq 0$.

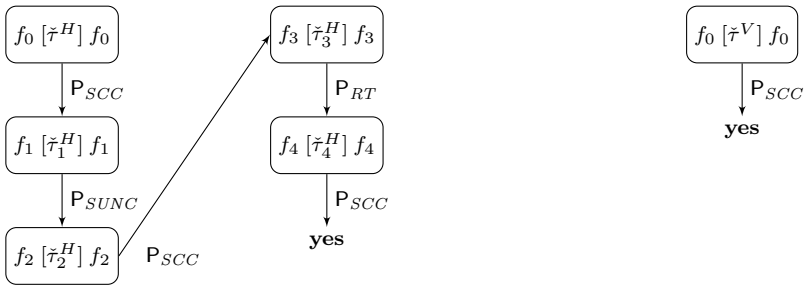


Fig. 5 Operational termination of $\mathcal{R}$ in Example 23 in the Open 2D DP Framework

4.4 Usable rules for CTRS Problems

The notion of *usable rule* was introduced in connection with *innermost termination* of TRSs[10] as (a superset of) those rules that may possibly be used in an innermost reduction of a normalized instance $\sigma(t)$ of a term t by a substitution σ [1].[11] Such rules are estimated by considering the function symbols in t only. This is possible because the substitution σ is assumed to be *normalized*.

In [15,34], usable rules were applied to prove termination of rewriting. The rules are also estimated by considering the function symbols in t only. The assumption of being a normalized instance is now replaced by *termination* of $\sigma(t)$. When dealing with *minimal* chains of dependency pairs, this can be fruitfully assumed. In this way, since $\sigma(t)$ is (assumed to be) terminating, one can (i) *precompute* all possible reducts of $\sigma(t)$ by $\mathcal{R}$ and store them in a list built by using a new function symbol c (the usual "cons", i.e., the *cons*tructor of lists), (ii) use those reducts involving usable rules for t only, and (iii) eventually retrieve from the list any other reduct of $\sigma(t)$ by using the following TRS $\mathcal{C}_\varepsilon$:

$$\mathsf{c}(x,y) \to x \tag{66}$$
$$\mathsf{c}(x,y) \to y \tag{67}$$

to traverse the list of reducts. Borrowing from [33, Definition 3.7], we define the Cap-function as a mapping from CTRSs $\mathcal{R}$ and terms t into terms as follows: for all terms t, $\textsc{Cap}_\mathcal{R}(t)$ is a term t' which is obtained from t by replacing all maximal subterms $t|_p$ of t by fresh variables whenever $t|_p$ is a variable or there is a substitution σ and a term u such that $\sigma(t) \to_\mathcal{R}^* \circ \xrightarrow{p}_\mathcal{R} u$. An *estimated* Cap*-function*, ECap, is a function with the following property: whenever Cap replaces a subterm at a position p by a fresh variable, then there is a subterm at a higher position $p' \leq p$ which is replaced by a fresh variable using ECap. As remarked by Thiemann, the essential property of an estimated Cap-function is that $\textsc{ECap}(t)$ contains the structure of $\sigma(t)$ after any number of reduction steps, i.e., if $\sigma(t) \to_\mathcal{R}^* u$ for some substitution σ, then $u = \sigma'(\textsc{ECap}_\mathcal{R}(t))$ for some substitution σ' which differs from

[10] A TRS is said to be *innermost terminating* if the (one-step) innermost rewriting relation $\to_i$, which rewrites terms only if they contain no other redex, is terminating.

[11] A 'normalized instance' of a term t is an instance $\sigma(t)$ by a substitution σ such that $\sigma(x)$ is a normal form for all variables $x \in \mathcal{V}ar(t)$.

σ only in the fresh variables that are introduced by ECap. The following definition concerns this previous discussion.

Definition 9 (Needed rules) Let $\mathcal{R}$ be a CTRS and ECap be an estimated Cap-function. The *needed rules* of a term t are defined as the smallest set $\mathsf{Needed}_{\mathcal{R}}(t) \subseteq \mathcal{R}$ such that

1. If $t = f(t_1, \ldots, t_k)$ and $f(\text{ECap}_{\mathcal{R}}(t_1), \ldots, \text{ECap}_{\mathcal{R}}(t_k))$ and ℓ (for some rule $\alpha : \ell \to r \Leftarrow c \in \mathcal{R}$) *unify* with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, then $\alpha \in \mathsf{Needed}_{\mathcal{R}}(t)$.
2. If $t = f(t_1, \ldots, t_k)$, then $\bigcup_{i=1}^{k} \mathsf{Needed}_{\mathcal{R}}(t_i) \subseteq \mathsf{Needed}_{\mathcal{R}}(t)$.
3. If $\ell \to r \Leftarrow c \in \mathsf{Needed}_{\mathcal{R}}(t)$, then $\mathsf{Needed}_{\mathcal{R}}(r) \cup \bigcup_{s \to t \in c} \mathsf{Needed}_{\mathcal{R}}(s) \subseteq \mathsf{Needed}_{\mathcal{R}}(t)$.

In principle, $\mathsf{Needed}_{\mathcal{R}}(t)$, which borrows from [33, Definition 4.5] for TRSs, suffices to capture all rules that are involved in $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chains and can be used as a basis for the definition of usable rules for CTRSs as given in [25, Definition 11]. However, in the following we refine this by distinguishing *two* subsets of rules from $\mathcal{R}$ which are relevant in $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chains:

1. The rules from $\mathcal{R}$ which are used to *evaluate* the instantiated conditional part $\sigma(c)$ of rules $\ell \to r \Leftarrow c \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{R}$ by the substitution σ associated to the $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain, and
2. The rules from $\mathcal{R}$ which can be used to *perform* the *rewriting steps* that *connect* an instance $\sigma(v)$ of the right-hand side v of a pair $u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$ and the instance $\sigma(u')$ of the left-hand side u' of the next pair $u' \to v' \Leftarrow c' \in \mathcal{P} \cup \mathcal{Q}$, disregarding the necessary evaluation of the conditions, which is considered in the previous item.

Exploiting this distinction is important. The rules in the first group above do *not* require comparisons using the components of a removal triple. Indeed, consider item 1 in Definition 6 and terms s and t such that $s \to_{\mathcal{S},\mathcal{R}} t$ holds due to the application of a rule $\ell \to r \Leftarrow \bigwedge_{i=1}^{n} s_i \to t_i \in \mathcal{S} \subseteq \mathcal{R}$. Then, s and t (which include instances of ℓ and r as subterms) should satisfy $s \gtrsim t$; however, terms $\sigma(s_i)$ and $\sigma(t_i)$ in auxiliary computations $\sigma(s_i) \to_{\mathcal{R}}^{*} \sigma(t_i)$ which are necessary to implement such a rewriting step (hopefully without involving rules in $\mathcal{S}$) are not required to be comparable with $\gtrsim$, $\succeq$, or $\sqsupset$. Thus, we try to take advantage of the precise identification of the rules in $\mathcal{S}$. However, this can be tricky.

Example 26 Consider the CTRS problem $(\mathcal{P}, \emptyset, \mathcal{R}, f)$ with $\mathcal{P} = \{\mathsf{F}(x, y) \to \mathsf{F}(y, y) \Leftarrow \mathsf{g}(x) \to y\}$ and $\mathcal{R} = \{\mathsf{g}(\mathsf{a}) \to \mathsf{a}\}$, and the following infinite sequence:

$$\mathsf{F}(\mathsf{a}, \mathsf{g}(\mathsf{a})) \xrightarrow{\Lambda}_{\mathcal{P},\mathcal{R}} \mathsf{F}(\underline{\mathsf{g}(\mathsf{a})}, \mathsf{g}(\mathsf{a})) \to_{\mathcal{R}} \mathsf{F}(\mathsf{a}, \mathsf{g}(\mathsf{a})) \xrightarrow{\Lambda}_{\mathcal{P},\mathcal{R}} \cdots$$

Without using rule $\mathsf{g}(\mathsf{a}) \to \mathsf{a}$, the previous infinite sequence is *not* possible. However, the need of this rule is discovered only when the *left-hand side of the conditional part* is examined (even though the rule is *not* used to evaluate the conditional part!). The point is that, due to the *unifiability* of the left- and right-hand sides $\mathsf{g}(x)$ and y of the condition in the rule of $\mathcal{P}$, symbol g becomes part of the symbols to be considered when instances of $\mathsf{F}(y, y)$ are evaluated. Thus, the needed rules which are obtained from the left-hand side of the conditions may also be relevant to generate the infinite chain.

Therefore, we can distinguish two different sets of needed rules if we can find a set of needed rules that is being used to evaluate the conditions only. This is the case when the right-hand side t of a condition $s \to t$ is either ground or nonground but its variables occur in t only.

Definition 10 Given a rule $\alpha : u \to v \Leftarrow c$ and $s_i \to t_i \in c$, we say that t_i has the fresh-var property in α, written $fv_{u\to v\Leftarrow c}(t_i)$, $fv_\alpha(t_i)$ or just $fv(t_i)$, if no variable in t_i occurs anywhere else in the rule, i.e., $\mathcal{V}ar(t_i)$ and $\mathcal{V}ar(u \to v \Leftarrow c[]_i) \cup \mathcal{V}ar(s_i)$ are *disjoint*. We say that α has the fresh-var property if for all $s \to t \in c$, t has the fresh-var property.

In functional programming, variables with the fresh-var property correspond to unsused variables (usually marked by an underscore). These variables can appear when we want to introduce reachability conditions for the application of the conditional rule but the reachability result is not longer used.

Example 27 All pairs in Example 2 and Example 15 have the fresh-var property. In contrast, pairs (46) and (47) in Example 23 lack it.

First, we define $\mathsf{Needed}^{\to}_{\mathcal{R}}(t)$ to be as in Definition 9, but with the third item replaced by the following:

3'. *If* $\alpha : \ell \to r \Leftarrow c \in \mathsf{Needed}^{\to}_{\mathcal{R}}(t)$, *then*

$$\mathsf{Needed}^{\to}_{\mathcal{R}}(r) \cup \bigcup_{s\to t\in c, \neg fv_\alpha(t)} \mathsf{Needed}^{\to}_{\mathcal{R}}(s) \subseteq \mathsf{Needed}^{\to}_{\mathcal{R}}(t).$$

i.e., in contrast to $\mathsf{Needed}_{\mathcal{R}}$, the conditional part c of the rules $\ell \to r \Leftarrow c \in \mathsf{Needed}^{\to}_{\mathcal{R}}(t)$ is *not* always considered to be adding new rules to $\mathsf{Needed}^{\to}(t)$ because we can distinguish a subset of needed rules that are going to be used to evaluate conditions only. Given a CTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$, we let

$$\mathsf{NeedR}^{\to}(\check{\tau}) = \bigcup_{u\to v\Leftarrow c\in\mathcal{P}\cup\mathcal{Q}} \mathsf{Needed}^{\to}_{\mathcal{R}_F}(v) \cup \bigcup_{s\to t\in c, \neg fv(t)} \mathsf{Needed}^{\to}_{\mathcal{R}_F}(s) \tag{68}$$

$$\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) = \bigcup_{u\to v\Leftarrow c\in\mathcal{P}\cup\mathcal{Q}\cup\mathsf{NeedR}^{\to}(\check{\tau})} \bigcup_{s\to t\in c} \mathsf{Needed}_{\mathcal{R}_F}(s) \tag{69}$$

Example 28 The CTRS in Figure 6 implements *quicksort* (see [29, Section 1]; we have added rules to compare natural numbers in Peano's notation with leq, and for the *appending* operator app for lists). $\mathsf{DP}_H(\mathcal{R})$ consists of the rules:

$$\mathsf{LEQ}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{LEQ}(x, y) \tag{80}$$

$$\mathsf{APP}(\mathsf{cons}(x, xs), ys) \to \mathsf{APP}(xs, ys) \tag{81}$$

$$\mathsf{QSORT}(\mathsf{cons}(x, xs)) \to \mathsf{APP}(\mathsf{qsort}(ys), \mathsf{cons}(x, \mathsf{qsort}(zs))) \Leftarrow \mathsf{split}(x, xs) \to \mathsf{pair}(ys, zs) \tag{82}$$

$$\mathsf{QSORT}(\mathsf{cons}(x, xs)) \to \mathsf{QSORT}(ys) \Leftarrow \mathsf{split}(x, xs) \to \mathsf{pair}(ys, zs) \tag{83}$$

$$\mathsf{QSORT}(\mathsf{cons}(x, xs)) \to \mathsf{QSORT}(zs) \Leftarrow \mathsf{split}(x, xs) \to \mathsf{pair}(ys, zs) \tag{84}$$

$$\mathsf{leq}(0, x) \to \mathsf{true} \tag{70}$$
$$\mathsf{leq}(\mathsf{s}(x), 0) \to \mathsf{false} \tag{71}$$
$$\mathsf{leq}(s(x), \mathsf{s}(y)) \to \mathsf{leq}(x, y) \tag{72}$$
$$\mathsf{app}(\mathsf{nil}, xs) \to xs \tag{73}$$
$$\mathsf{app}(\mathsf{cons}(x, xs), ys) \to \mathsf{cons}(x, \mathsf{app}(xs, ys)) \tag{74}$$
$$\mathsf{split}(x, \mathsf{nil}) \to \mathsf{pair}(\mathsf{nil}, \mathsf{nil}) \tag{75}$$
$$\mathsf{split}(x, \mathsf{cons}(y, ys)) \to \mathsf{pair}(xs, \mathsf{cons}(y, zs)) \tag{76}$$
$$\Leftarrow \mathsf{leq}(x, y) \to \mathsf{true}, \mathsf{split}(x, ys) \to \mathsf{pair}(xs, zs)$$
$$\mathsf{split}(x, \mathsf{cons}(y, ys)) \to \mathsf{pair}(\mathsf{cons}(y, xs), zs) \tag{77}$$
$$\Leftarrow \mathsf{leq}(x, y) \to \mathsf{false}, \mathsf{split}(x, ys) \to \mathsf{pair}(xs, zs)$$
$$\mathsf{qsort}(\mathsf{nil}) \to \mathsf{nil} \tag{78}$$
$$\mathsf{qsort}(\mathsf{cons}(x, xs)) \to \mathsf{app}(\mathsf{qsort}(ys), \mathsf{cons}(x, \mathsf{qsort}(zs))) \tag{79}$$
$$\Leftarrow \mathsf{split}(x, xs) \to \mathsf{pair}(ys, zs)$$

Fig. 6 Implementation of *quicksort* as a CTRS (Example 28)

$\mathsf{DP}_V(\mathcal{R})$ consists of the following rules:

$$\mathsf{SPLIT}(x, \mathsf{cons}(y, ys)) \to \mathsf{LEQ}(x, y) \tag{85}$$
$$\mathsf{SPLIT}(x, \mathsf{cons}(y, ys)) \to \mathsf{SPLIT}(x, ys) \Leftarrow \mathsf{leq}(x, y) \to \mathsf{true} \tag{86}$$
$$\mathsf{SPLIT}(x, \mathsf{cons}(y, ys)) \to \mathsf{SPLIT}(x, ys) \Leftarrow \mathsf{leq}(x, y) \to \mathsf{false} \tag{87}$$
$$\mathsf{QSORT}(\mathsf{cons}(x, xs)) \to \mathsf{SPLIT}(x, xs) \tag{88}$$

$\mathsf{DP}_{VH}(\mathcal{R})$ consists of a single rule (80). For $\check{\tau}^V$, we have $\mathsf{P}_{SCC}(\check{\tau}^V) = \{\check{\tau}_1^V\}$, where $\check{\tau}_1^V = (\{(86), (87)\}, \emptyset, \mathcal{R}, f)$. Then, $\mathsf{P}_\triangleright$ applied to $\check{\tau}_1^V$ and the subsequent application of P_{SCC} proves it finite.

For $\check{\tau}^H$, we have $\mathsf{P}_{SCC}(\check{\tau}^H) = \{\check{\tau}_{11}^H, \check{\tau}_{12}^H, \check{\tau}_{13}^H\}$, where $\check{\tau}_{11}^H = (\{(80)\}, \emptyset, \mathcal{R}, f)$, $\check{\tau}_{12}^H = (\{(81)\}, \emptyset, \mathcal{R}, f)$, and $\check{\tau}_{13}^H = (\{(83), (84)\}, \emptyset, \mathcal{R}, f)$. Then, $\mathsf{P}_\triangleright$ applied to $\check{\tau}_{11}^H$ and $\check{\tau}_{12}^H$ and a subsequent application of P_{SCC} proves them finite. However, $\mathsf{P}_\triangleright$ cannot be applied to $\check{\tau}_{13}^H$. We will use P_{RTN}. We have:

$$\mathsf{NeedR}^{\to}(\check{\tau}_{13}^H) = \emptyset \tag{89}$$
$$\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}_{13}^H) = \{(70), (71), (72), (75), (76), (77)\} \tag{90}$$

We use $\mathcal{N}^c = \mathsf{NeedR}^{\mathsf{c}}(\check{\tau})$ to evaluate the conditions whose right-hand sides have the fresh-var property and $\mathcal{N}^{\to} = \mathsf{NeedR}^{\to}(\check{\tau})$ otherwise. However, rules in $\mathcal{N}^{\to}$ can be applied in conditions whose right-hand sides have the fresh-var property:

Example 29 Consider $(\mathcal{P}, \emptyset, \mathcal{R}, f)$ with $\mathcal{P} = \{\mathsf{A} \to \mathsf{F}(\mathsf{b}), \mathsf{F}(x) \to \mathsf{A} \Leftarrow x \to \mathsf{c}, x \to \mathsf{d}\}$, $\mathcal{R} = \{\mathsf{b} \to \mathsf{c}, \mathsf{b} \to \mathsf{d}\}$, and the following infinite sequence:

$$\mathsf{A} \to_{\mathcal{P},\mathcal{R}} \mathsf{F}(\mathsf{b}) \to_{\mathcal{P},\mathcal{R}} \mathsf{A} \to_{\mathcal{P},\mathcal{R}} \cdots \tag{91}$$

Rules $\mathsf{b} \to \mathsf{c}$ and $\mathsf{b} \to \mathsf{d}$ are used to evaluate the reachability conditions $\mathsf{b} \to^* \mathsf{c}$ and $\mathsf{b} \to^* \mathsf{d}$ in the step $\mathsf{F}(\mathsf{b}) \to_{\mathcal{P},\mathcal{R}} \mathsf{A}$. The attempt to '*move*' the rewriting steps $\mathsf{b} \to \mathsf{c}$ and $\mathsf{b} \to \mathsf{d}$ to the *connection part* of the chain fails. For instance:

$$\mathsf{A} \to_{\mathcal{P},\mathcal{R}} \mathsf{F}(\underline{\mathsf{b}}) \to_{\mathcal{R}} \mathsf{F}(\mathsf{c}) \not\to_{\mathcal{P},\mathcal{R}} \mathsf{A} \to_{\mathcal{P},\mathcal{R}} \tag{92}$$

fails in the second $\to_{\mathcal{P},\mathcal{R}}$-step because the corresponding reachability conditions $\mathsf{c} \to^* \mathsf{c}$ and $\mathsf{c} \to^* \mathsf{d}$ are *not* simultaneously satisfiable now. The situation is similar if $\mathsf{F}(\underline{\mathsf{b}}) \to_{\mathcal{R}} \mathsf{F}(\mathsf{d})$ is issued in chain (92) above. Thus, in chain (91), the rules in $\mathcal{R}$ are needed to rewrite the (instantiated) *conditional part* of the pairs rather than to connect them.

Example 29 shows that we have to impose $\mathcal{N}^{\to} \subseteq \mathcal{N}^c$. This compatibility condition between $\mathcal{N}^c$ and $\mathcal{N}^{\to}$ can be dropped for conditions $s_i \to t_i \in c$ where t_i has the fresh-var property and the left-hand side s_i is ground. If this happens, we say that t_i has the *strong* fresh-var property, written $sfv_{u\to v\Leftarrow c}(t_i)$, $sfv_{\alpha}(t_i)$ or just $sfv(t_i)$.

Example 30 Consider the rule $\mathsf{a} \to \mathsf{b} \Leftarrow \mathsf{c} \to \mathsf{f}(x)$. Since x occurs in $\mathsf{f}(x)$ only, $\mathsf{f}(x)$ has the fresh-var property. Since c is ground, it has the strong fresh-var property.

In the following, $\mathcal{C}_{\varepsilon}^{\mathcal{N}}$ and $\mathcal{C}_{\varepsilon}^{\mathcal{R}}$ are TRSs with rules (66) and (67), where symbol c is replaced by $c_{\mathcal{N}}$ and $c_{\mathcal{R}}$, respectively, of rank $\mathrm{s}\,\mathrm{s} \to \mathrm{s}$ (which also belong to $\Sigma^{\check{\tau}}_{\mathrm{SS},\mathrm{S}}$). The intuition is that rules in $\mathcal{C}_{\varepsilon}^{\mathcal{N}}$ will be used to keep track of rules which are applied to *connect* dependency pairs, whilst rules in $\mathcal{C}_{\varepsilon}^{\mathcal{R}}$ will be used to keep track of rules which are applied to *evaluate* rule conditions. We also consider the rule:

$$c_{\mathcal{N}}(x, y) \to c_{\mathcal{R}}(x, y) \tag{93}$$

which is necessary to eventually take care of terms which are lifted to the 'evaluation' part of the conditions from the 'connection' part of pairs. The move from the 'evaluation' part to the 'connection' part (when needed) is treated by *collapsing* $\mathcal{C}_{\varepsilon}^{\mathcal{R}}$ and $\mathcal{C}_{\varepsilon}^{\mathcal{N}}$ into $\mathcal{C}_{\varepsilon}$ (see Definition 11). Also, $\mathcal{N}^c$ and $\mathcal{N}^{\to}$ refer to subsets of rules in $\mathcal{R}$ that are *needed* to evaluate the conditional part of rules in $\mathcal{P} \cup \mathcal{Q} \cup \mathcal{R}$ (in the case of $\mathcal{N}^c$) only, or perform the connection between pairs in $\mathcal{P}$ and $\mathcal{Q}$ (in the case of $\mathcal{N}^{\to}$). Depending on the shape of the rules in $\mathcal{Q}$ and $\mathcal{R}$ these sets $\mathcal{N}^c$ and $\mathcal{N}^{\to}$ will be required to include some of the sets of rules discussed above.

Definition 11 (Removal triple with needed rules processor) Let $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an OCTRS problem, $\alpha : u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$, $\mathcal{N}^c, \mathcal{N}^{\to} \subseteq \mathcal{R}$, and $(\gtrsim, \succeq, \sqsupset)$ be a $((\mathcal{P}, \mathcal{Q}, \mathcal{N}^c \cup \mathcal{C}_{\varepsilon}^{\mathcal{R}} \cup \{(93)\}, \varphi), \mathcal{N}^{\to} \cup \mathcal{C}_{\varepsilon}^{\mathcal{N}})$-compatible removal triple such that:

1. $\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \subseteq \mathcal{N}^c$,
2. $\mathsf{NeedR}^{\to}(\check{\tau}) \subseteq \mathcal{N}^{\to}$,
3. for all rules $\alpha' : \ell' \to r' \Leftarrow c'$,
   - if $\alpha' \in \mathcal{N}^{\to}$, then $\mathsf{Needed}^{\to}_{\mathcal{R}_F}(r') \subseteq \mathcal{N}^{\to}$ and, for all $s' \to t' \in c'$,
     (a) $\mathsf{Needed}^{\to}_{\mathcal{R}_F}(s') \subseteq \mathcal{N}^{\to}$ if $fv(t')$ does not hold, and
     (b) $\mathsf{Needed}_{\mathcal{R}_F}(s') \subseteq \mathcal{N}^c$ if $fv(t')$ holds,
   - if $\alpha' \in \mathcal{N}^c$, then $\mathsf{Needed}_{\mathcal{R}_F}(r') \cup \bigcup_{s'\to t' \in c'} \mathsf{Needed}_{\mathcal{R}_F}(s') \subseteq \mathcal{N}^c$.
4. if there is $u' \to v' \Leftarrow c' \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{N}^{\to}$ and $s' \to t' \in c'$ such that $sfv(t')$ does not hold, then $\mathcal{N}^{\to} \subseteq \mathcal{N}^c$.
5. if there is $u' \to v' \Leftarrow c' \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{N}^{\to}$ and $s' \to t' \in c'$ such that $fv(t')$ does not hold, then $\mathcal{C}_{\varepsilon}$ is used instead of $\mathcal{C}_{\varepsilon}^{\mathcal{N}}$ and $\mathcal{C}_{\varepsilon}^{\mathcal{R}}$ in the compatibility condition required for the removal triple (and (93) is therefore removed).

P_{RTN} is given by

$$\mathsf{P}_{RTN}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) = \{(\mathcal{P}[\emptyset]_{\alpha}, \mathcal{Q}[\emptyset]_{\alpha}, \mathcal{R}, \varphi)\}$$

iff $\sigma(u) \sqsupset \sigma(v)$ whenever $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ for all substitutions σ and all $s \to t \in c$.

Note that, $\mathsf{NeedR}^{\to}$ satisfies the conditions of $\mathcal{N}^{\to}$ in item 3 and $\mathsf{NeedR}^{\mathsf{c}}$ satisfies the conditions of $\mathcal{N}^{c}$ in item 3.

Theorem 5 (Soundness and completeness of P_{RTN}) *P_{RTN} is m-sound and complete. Therefore, $\Pi(\mathsf{P}_{RTN}) = \{\langle \mathsf{m}, f\rangle\}$.*

The proof of Theorem 5 is in Appendix B.

4.4.1 Implementing the use of P_{RTN} as a satisfiability problem

Given an OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ and $\alpha \in \mathcal{P} \cup \mathcal{Q}$, consider $\Omega^{\check{\tau}}_{RTN} = (S_{DP}, \Sigma^{\check{\tau}}_{RTN}, \Pi^{\check{\tau}}_{RTN})$, where $\Sigma^{\check{\tau}}_{RTN}$ is $\Sigma^{\check{\tau}}$ enriched with two new binary symbols $c_{\mathcal{R}}, c_{\mathcal{N}} : \mathrm{S}\ \mathrm{S} \to \mathrm{S}$; $\Pi^{\check{\tau}}_{RTN}$ consists of $\Pi^{\check{\tau}}_{\mathrm{PP}} = \{\to_{\mathcal{N}}, \to_{\mathcal{P}}, \to_{\mathcal{Q}}, \pi_{\gtrsim}, \pi_{\succeq}, \pi_{\sqsupset}\}$ and $\Pi^{\check{\tau}}_{\mathrm{SS}} = \{\to_{\mathcal{R}}, \to^{*}_{\mathcal{R}}, \to_{\mathcal{N}}\}$. Note that $\to_{\mathcal{N}}$ is overloaded but $\to_{\mathcal{R}}$ is *not* overloaded anymore. The introduction of the new predicate $\to_{\mathcal{N}}$ changes the interpretation of $\to_{\mathcal{R}}$:

- Symbols $\to_{\mathcal{R}}$ and $\to^{*}_{\mathcal{R}}$ represent the usual one-step and many-step rewriting relations on terms by using $\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon} \cup \{(93)\}$.
- Symbol $\to_{\mathcal{N}}$ represent one-step reductions with $\to_{\mathsf{NeedR}^{\to}(\check{\tau})\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathsf{NeedR}^{\mathsf{c}}(\check{\tau})\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}}$, i.e., the rules $\ell \to r \Leftarrow c$ issuing each step on a term s (possibly *below* the root of s) belong to $\mathsf{NeedR}^{\to}(\check{\tau}) \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}$ but the conditional part c is evaluated using $\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon} \cup \{(93)\}$.

Now we let $\mathcal{S}^{RTN}_{\check{\tau},\alpha} = \mathcal{S}^{RT}_{Rel,\alpha} \cup \mathcal{S}^{RTN}_{CRel,\alpha} \cup \mathcal{S}^{RTN}_{\mathcal{R}} \cup \mathcal{S}^{RTN}_{\mathcal{N},\alpha} \cup \mathcal{S}^{RT}_{\mathcal{P},\alpha} \cup \mathcal{S}^{RT}_{\mathcal{Q},\alpha}$. Note that the sets of sentences $\mathcal{S}^{RT}_{Rel}$, $\mathcal{S}^{RT}_{\mathcal{P},\alpha}$ and $\mathcal{S}^{RT}_{\mathcal{Q},\alpha}$ are as in Section 4.3.2, although their *interpretation* should take into account the interpretation of $\to_{\mathcal{R}}$ and $\to^{*}_{\mathcal{R}}$ as explained above. The new sets of sentences are as follows:

1. $\mathcal{S}^{RTN}_{CRel,\alpha}$ is $\mathcal{S}^{RT}_{CRel,\alpha}$ with rule (56) replaced by the following

$$(\forall x, y : \mathrm{P})\ x \to_{\mathcal{N}} y \Rightarrow x\ \pi_{\gtrsim}\ y \tag{94}$$

2. $\mathcal{S}^{RTN}_{\mathcal{R}}$ is similar to $\mathcal{S}^{RT}_{\mathcal{R}}$, but using rules in $\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon} \cup \{(93)\}$ only. $\mathcal{S}^{RTN}_{\mathcal{R}}$ contains the formulas (59) and (60). We add formulas (61) for each k ary symbol $f \in \Sigma^{\check{\tau}}_{\mathrm{S}\cdots\mathrm{S},\mathrm{S}}$ (except $c_{\mathcal{N}}$) and $1 \le i \le k$, and a formula (62) for each rule $\ell \to r \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n \in \mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}$.
3. $\mathcal{S}^{RTN}_{\mathcal{N}}$ describes reductions with $\to_{\mathsf{NeedR}^{\to}(\check{\tau})\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathsf{NeedR}^{\mathsf{c}}(\check{\tau})\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}}$. There is a formula

$$(\forall \mathbf{x}, y_i : \mathrm{S})\ x_i \to_{\mathcal{N}} y_i \Rightarrow f(x_1, \ldots, x_i, \ldots, x_k) \to_{\mathcal{N}} f(x_1, \ldots, y_i, \ldots, x_k) \tag{95}$$

for each k ary symbol $f \in \Sigma^{\check{\tau}}_{\mathrm{S}\cdots\mathrm{S},\mathrm{S}} \cup \Sigma^{\check{\tau}}_{\mathrm{S}\cdots\mathrm{S},\mathrm{P}}$ (except $c_{\mathcal{R}}$) and i, $1 \le i \le k$, with $\mathbf{x} = x_1, \ldots, x_k$, and a formula

$$(\forall \mathbf{x} : \mathrm{S})\ s_1 \to^{*}_{\mathcal{R}} t_1 \wedge \cdots \wedge s_n \to^{*}_{\mathcal{R}} t_n \Rightarrow \ell \to_{\mathcal{N}} r \tag{96}$$

for each rule $\ell \to r \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n \in \mathsf{NeedR}^{\to}(\check{\tau}) \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}$, where $\mathbf{x}$ contains all variables occurring in the rule.

Example 31 For $\check{\tau}^{H}_{13} = (\{(83), (84)\}, \emptyset, \mathcal{R}, f)$ in Example 28, we have:

1. $\mathcal{S}^{RT}_{Rel,(83)} = \{(54), (55)\}$ and $\mathcal{S}^{RTN}_{CRel,(83)} = \{(57), (94)\}$.

$$
\begin{aligned}
(\forall x : \mathrm{S})\ x &\to^*_{\mathcal{R}} x \\
(\forall x, y, z : \mathrm{S})\ x \to_{\mathcal{R}} y \wedge y \to^*_{\mathcal{R}} z \Rightarrow x &\to^*_{\mathcal{R}} z \\
(\forall x : \mathrm{S})\ \mathsf{leq}(\mathsf{0}, x) &\to_{\mathcal{R}} \mathsf{true} \\
(\forall x : \mathrm{S})\ \mathsf{leq}(\mathsf{s}(x), \mathsf{0}) &\to_{\mathcal{R}} \mathsf{false} \\
(\forall x, y : \mathrm{S})\ \mathsf{leq}(s(x), \mathsf{s}(y)) &\to_{\mathcal{R}} \mathsf{leq}(x, y) \\
(\forall x : \mathrm{S})\ \mathsf{split}(x, \mathsf{nil}) &\to_{\mathcal{R}} \mathsf{pair}(\mathsf{nil}, \mathsf{nil}) \\
(\forall x, y, xs, ys, zs : \mathrm{S})\ \mathsf{leq}(x, y) \to^*_{\mathcal{R}} \mathsf{true} \wedge \mathsf{split}(x, ys) &\to^*_{\mathcal{R}} \mathsf{pair}(xs, zs) \ \Rightarrow \\
\mathsf{split}(x, \mathsf{cons}(y, ys)) &\to_{\mathcal{R}} \mathsf{pair}(xs, \mathsf{cons}(y, zs)) \\
(\forall x, y, xs, ys, zs : \mathrm{S})\ \mathsf{leq}(x, y) \to^*_{\mathcal{R}} \mathsf{false} \wedge \mathsf{split}(x, ys) &\to^*_{\mathcal{R}} \mathsf{pair}(xs, zs) \ \Rightarrow \\
\mathsf{split}(x, \mathsf{cons}(y, ys)) &\to_{\mathcal{R}} \mathsf{pair}(\mathsf{cons}(y, xs), zs) \\
(\forall x, y : \mathrm{S})\ c_{\mathcal{R}}(x, y) &\to_{\mathcal{R}} x \\
(\forall x, y : \mathrm{S})\ c_{\mathcal{R}}(x, y) &\to_{\mathcal{R}} y \\
(\forall x, y : \mathrm{S})\ c_{\mathcal{N}}(x, y) &\to_{\mathcal{R}} c_{\mathcal{R}}(x, y)
\end{aligned}
$$

Fig. 7 Sentences $\mathcal{S}^{RTN}_{\mathcal{R}}$ in Example 31.

2. Since $\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}^H_{13}) = \{(70), (71), (72), (75), (76), (77)\}$ (see the last part of Example 28), $\mathcal{S}^{RTN}_{\mathcal{R}}$ is displayed in Figure 7.
3. Since $\mathsf{NeedR}^{\rightarrow}(\check{\tau}^H_{13}) = \emptyset$ (see also Example 28), $\mathcal{S}^{RTN}_{\mathcal{N}}$ contains formulas (95) for each k-ary symbol f and $1 \le i \le k$, together with

$$
\begin{aligned}
(\forall x, y : \mathrm{S})\ c_{\mathcal{N}}(x, y) &\to_{\mathcal{N}} x \\
(\forall x, y : \mathrm{S})\ c_{\mathcal{N}}(x, y) &\to_{\mathcal{N}} y
\end{aligned}
$$

4. $\mathcal{S}^{RT}_{\mathcal{P},(83)}$ contains a single formula, corresponding to (84):

   $$(\forall x, xs, ys, zs : \mathrm{S})\ \mathsf{split}(x, xs) \to^*_{\mathcal{R}} \mathsf{pair}(ys, zs) \Rightarrow \mathsf{QSORT}(\mathsf{cons}(x, xs)) \to_{\mathcal{P}} \mathsf{QSORT}(zs)$$

   and $\mathcal{S}^{RT}_{\mathcal{Q},(83)}$ is empty.

Definition 12 (Semantic version of P_{RTN}) Let $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$ be an open CTRS problem, $\mathcal{A}$ be an $\Omega_{\check{\tau}}$-structure with $\mathcal{A}_{\mathrm{P}}$ nonempty, and $\alpha : u \to v \Leftarrow s_1 \to t_1, \ldots, s_n \to t_n \in \mathcal{P} \cup \mathcal{Q}$. Then, $\mathsf{P}_{RTN}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi) = \{(\mathcal{P}[\emptyset]_\alpha, \mathcal{Q}[\emptyset]_\alpha, \mathcal{R}, \varphi)\}$ iff $\mathcal{A} \models \mathcal{S}^{RTN}_{\check{\tau},\alpha} \cup \{(\forall \mathbf{x} : \mathrm{S}) \bigwedge_{i=1}^{n} s_i \to^*_{\mathcal{R}} t_i \Rightarrow u\ \pi_{\sqsupset}\ v\}$ (where $\mathbf{x}$ lists all variables in α) and $\pi^{\mathcal{A}}_{\sqsupset}$ is well-founded on $\mathcal{A}_{\mathrm{P}}$.

Example 32 (Operational termination of $\mathcal{R}$ in Example 28) We use processor P_{RTN} to remove pair (83), i.e.,

$$\mathsf{QSORT}(\mathsf{cons}(x, xs)) \to \mathsf{QSORT}(ys) \Leftarrow \mathsf{split}(x, xs) \to \mathsf{pair}(ys, zs)$$

from $\check{\tau}^H_{13}$ in Example 28. With AGES we obtain the following structure $\mathcal{A}$ with $\mathcal{A}_{\mathrm{P}} = \mathbb{N}$ and $\mathcal{A}_{\mathrm{S}} = \mathbb{Z} - \mathbb{N}$: constant symbols are interpreted as follows: $\mathsf{true}^{\mathcal{A}} = \mathsf{false}^{\mathcal{A}} = \mathsf{0}^{\mathcal{A}} = \mathsf{nil}^{\mathcal{A}} = -1$. Also,

$$
\begin{array}{llll}
\mathsf{s}^{\mathcal{A}}(x) = x - 1 & \mathsf{qsort}^{\mathcal{A}}(x) = 2x + 1 & \mathsf{app}^{\mathcal{A}}(x, y) = x & \mathsf{cons}^{\mathcal{A}}(x, y) = x + y \\
\mathsf{leq}^{\mathcal{A}}(x, y) = x + y + 1 & \mathsf{pair}^{\mathcal{A}}(x, y) = x + y & \mathsf{split}^{\mathcal{A}}(x, y) = x + y & \\
\mathsf{QSORT}^{\mathcal{A}}(x) = -x & c^{\mathcal{A}}_{\mathcal{N}}(x, y) = x + y & c^{\mathcal{A}}_{\mathcal{R}}(x, y) = x + y &
\end{array}
$$

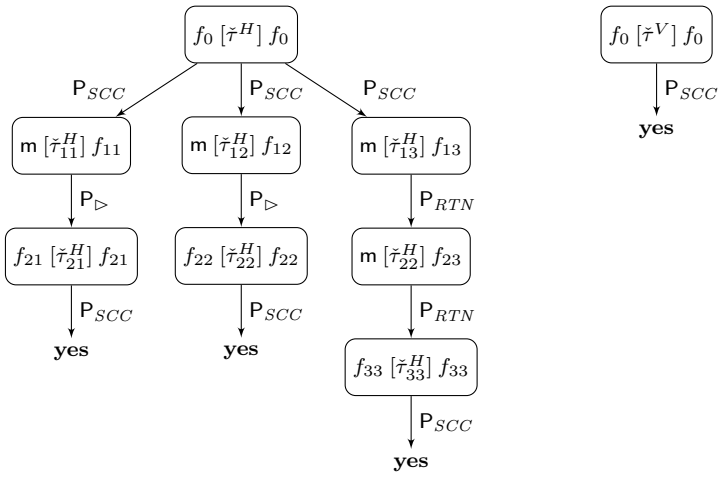


Fig. 8 Operational termination of $\mathcal{R}$ in Example 28 in the Open 2D DP Framework

$$\begin{array}{lll} x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathrm{P}} y \Leftrightarrow \mathit{true} & x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathrm{S}} y \Leftrightarrow y \geq x & x(\to^*_{\mathcal{R}})^{\mathcal{A}}_{\mathrm{S}} y \Leftrightarrow y \geq x \\ x(\to_{\mathcal{N}})^{\mathcal{A}}_{\mathrm{P}} y \Leftrightarrow x \geq y & x(\to_{\mathcal{N}})^{\mathcal{A}}_{\mathrm{S}} y \Leftrightarrow y \geq x & x(\to_{\mathcal{P}})^{\mathcal{A}}_{\mathrm{S}} y \Leftrightarrow x > y \\ x\ \pi^{\mathcal{A}}_{\gtrsim}\ y \Leftrightarrow x \geq y & x\ \pi^{\mathcal{A}}_{\succsim}\ y \Leftrightarrow x \geq y & x\ \pi^{\mathcal{A}}_{\sqsupset}\ y \Leftrightarrow x > y \end{array}$$

is a model of $\mathcal{S}^{RTN}_{\check{\tau}^H_{13},(83)}$ in Example 28 and also of

$$(\forall x, xs, ys, zs : \mathrm{S})\ \mathsf{split}(x, xs) \to^*_{\mathcal{R}} \mathsf{pair}(ys, zs) \Rightarrow \mathsf{QSORT}(\mathsf{cons}(x, xs))\ \pi_{\sqsupset}\ \mathsf{QSORT}(ys)$$

Therefore, $\mathsf{P}_{RTN}(\check{\tau}^H_{13}) = \{\check{\tau}^H_{23}\}$, where $\check{\tau}^H_{23} = (\{(84)\}, \emptyset, \mathcal{R}, f)$. We proceed similarly to remove (84) from $\check{\tau}^H_{23}$ to finally obtain a proof of operational termination of the quicksort CTRS $\mathcal{R}$ in Example 28 (Figure 8).

4.4.2 P_{RTN} *vs.* P_{RT}

Since $\mathsf{NeedR}^{\to}(\check{\tau}) \cup \mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \subseteq \mathcal{R}$, processor P_{RTN} usually considers *fewer* rules than P_{RT} when dealing with an OCTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, f)$. The following question naturally arises: is P_{RT} somehow 'subsumed' by P_{RTN} and therefore P_{RT} could be dismissed? In general, this is *not* possible!

In the literature about the DP Framework for TRSs (see, e.g., [11, Section 4.1] and [33, Section 4.2]) it is shown that usable and needed rules improve on the use of reduction pairs[12] $(\succsim, \succ)$ whenever $\succsim$ is a *simplification ordering*, i.e., such that for all terms s and subterms t of s, $s \succsim t$ holds (see, e.g., [11, footnote 7]); this is usually called the *subterm property*. Simplification orderings (e.g., polynomial orderings over the naturals [17], path orderings [7], etc.) are often used in most termination tools as *base* orderings to implement proofs of termination.

A requirement of use of DP Processors based on usable rules is the so-called $\mathcal{C}_\varepsilon$-compatibility of $\succsim$ with the rules in $\mathcal{C}_\varepsilon$ (rules (66) and (67)), i.e., $\mathsf{c}(x, y) \succsim x$ and

[12] In a reduction pair $(\succsim, \succ)$ consists of a *reflexive, transitive, monotonic (closed under contexts) and stable (closed under substitutions) relation* $\succsim$*, and a stable well-founded ordering* $\succ$ such that $\succsim \circ \succ \subseteq \succ$ or $\succ \circ \succsim \subseteq \succ$ [12, Section 2.3].

$\mathsf{c}(x,y) \gtrsim y$ (see, e..g., [11, Theorems 3 and 6]). Clearly, when using reduction pairs based on simplification orderings, such a requirement is automatically fulfilled. Therefore, tools which are mostly based on using simplification orderings promote the use of processors based on usable rules rather than the 'raw' reduction pair processor. This practice is somehow justified on the following grounds (we use the notation in this paper):

> If $\gtrsim$ in a removal pair $(\gtrsim, \succeq, \sqsupset)$ is a simplification ordering, then P_{RTN} succeeds whenever P_{RT} succeeds.

Our experience with the semantic technique introduced here (where no subterm property is required) has been different, though. For instance, despite the fact that many rules are dismissed by using P_{RTN} in the proof of finiteness of $\check{\tau}^H_{13}$ in Example 32, such a proof can also be achieved with P_{RT}. Moreover, we found examples where we could use P_{RT} but *failed* to use P_{RTN}.

Example 33 Consider the following CTRS $\mathcal{R}$ [29, Example 7.2.51]

$$\mathsf{h}(\mathsf{d}) \to \mathsf{c}(\mathsf{a}) \tag{97}$$
$$\mathsf{h}(\mathsf{d}) \to \mathsf{c}(\mathsf{b}) \tag{98}$$
$$\mathsf{f}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), x) \to \mathsf{f}(x,x,x) \tag{99}$$
$$\mathsf{g}(x) \to \mathsf{k}(y) \Leftarrow \mathsf{h}(x) \to \mathsf{d}, \mathsf{h}(x) \to \mathsf{c}(y) \tag{100}$$

We have:

$$\mathsf{DP}_H(\mathcal{R}): \quad \mathsf{F}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), x) \to \mathsf{F}(x,x,x) \tag{101}$$
$$\mathsf{DP}_V(\mathcal{R}): \quad \mathsf{G}(x) \to \mathsf{H}(x) \tag{102}$$
$$\mathsf{G}(x) \to \mathsf{H}(x) \Leftarrow \mathsf{h}(x) \to \mathsf{d} \tag{103}$$

and $\mathsf{DP}_{VH}(\mathcal{R}) = \emptyset$. Note that $\check{\tau}^H = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, f) = (\{(101)\}, \emptyset, \mathcal{R}, f)$. Note also that $\mathsf{NeedR}^{\to}(\check{\tau}^H) = \mathsf{NeedR}^{\mathsf{c}}(\check{\tau}^H) = \emptyset$. In order to remove (101) from $\check{\tau}^H$, with AGES we obtain a model $\mathcal{A}$ of

$$\mathcal{S}^{RT}_{\check{\tau}^H,(101)} \cup \{(\forall x : \mathrm{S})\, \mathsf{F}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), x)\, \pi_{\sqsupset}\, \mathsf{F}(x,x,x)\}.$$

Domains are $\mathcal{A}_{\mathrm{P}} = \mathbb{N}$ and $\mathcal{A}_{\mathrm{S}} = \{0,1\}$. With regard to function symbols: $\mathsf{a}^{\mathcal{A}} = 0$, $\mathsf{b}^{\mathcal{A}} = \mathsf{d}^{\mathcal{A}} = \mathsf{g}^{\mathcal{A}}(x) = \mathsf{f}^{\mathcal{A}}(x,y,z) = 1$, $\mathsf{h}^{\mathcal{A}}(x) = \mathsf{k}^{\mathcal{A}}(x) = x$, and $\mathsf{F}^{\mathcal{A}}(x,y,z) = y - x + 1$. As for predicate symbols:

$$x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathrm{P}} y \Leftrightarrow x \geq y \qquad x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathrm{S}} y \Leftrightarrow x = 1 \qquad x(\to^*_{\mathcal{R}})^{\mathcal{A}}_{\mathrm{S}} y \Leftrightarrow \mathit{true}$$
$$x\, \pi^{\mathcal{A}}_{\gtrsim}\, y \Leftrightarrow x \geq y \qquad x\, \pi^{\mathcal{A}}_{\sqsupset}\, y \Leftrightarrow x > y$$

Then, $\mathsf{P}_{RT}(\check{\tau}^H) = \{\check{\tau}^H_1\}$ where $\check{\tau}^H_1 = (\emptyset, \emptyset, \mathcal{R}, f)$.

As remarked above, we failed to remove (101) from $\check{\tau}^H$ in Example 33 by applying P_{RTN} as in Section 4.4.1 with AGES and Mace4. Indeed, if the rules in $\mathcal{C}^{\mathcal{N}}_{\varepsilon}$ are used to connect pairs (as assumed by P_{RTN}), then we have an infinite chain

$$\mathsf{F}(\underline{c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}))}, c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b})), c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b})) \to \mathsf{F}(\mathsf{k}(\mathsf{a}), \underline{c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}))}, c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}))$$
$$\underline{\mathsf{F}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}))} \to_{\mathcal{P}} \mathsf{F}(\underline{c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}))}, c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b})), c_{\mathcal{N}}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b})) \to \cdots$$

Note that, since $\pi^{\mathcal{A}}_{\gtrsim}$ in Example 33 is the *equality*, the relation $\gtrsim$ on terms generated by the interpretation (i.e., $s \gtrsim t \Leftrightarrow \mathcal{A} \models s\, \pi_{\gtrsim}\, t$ for all terms s and t) *lacks* the subterm property. For instance, $\mathsf{g}(\mathsf{b}) \gtrsim \mathsf{b}$ does *not* hold.

As a conclusion, we can say that P_{RTN} outperforms P_{RT} if simplification orderings are used. However, P_{RT} should also be considered if not only simplification orderings are considered (as we do here).

5 Implementation and experimental evaluation

The framework and the processors described in [27] and in this paper have been implemented as a part of the tool MU-TERM [2]. The development consists of 20 modules and more than 4000 lines of code written in Haskell. Overall, MU-TERM consists of 181 modules and more than 38000 lines of Haskell code. We tested the 2D DP Framework in practice on the 117 examples in the TRS Conditional subcategory of the International Termination Competition during the years 2014–2019[13]. These 117 examples are part of the Termination Problem Data Base (TPDB[14], version 10.6). Currently, MU-TERM can solve 104 of these 117 examples automatically and is the most successful tool for proving operational termination of CTRSs. Furthermore, it is the only tool that proves operational nontermination of CTRSs.

This section is divided in two subsections: first, we describe the proof strategy implemented in MU-TERM; in particular, the one we used to participate in the Termination Competition. Then, we compare the improvements introduced by this paper with respect to the results described in [27].

5.1 Proof strategy and use of the processors

In the Termination Competition, participants have a limit of 300s for each input program to return a proof of operational (non-)termination (or a *don't know* answer). The arbitrary application of processors can generate a huge search space. For this reason, we need to choose a fixed strategy where processors that reduce the number of rules (and the search space) are used at the beginning and processors that can increase the number of rules (and also the search space), are used when former processors fail. The frequency of use for the different processors, hence, depends on the chosen strategy.

The strategy used by MU-TERM to find a proof in the termination competition is the following:

1. We check that the CTRS is valid in our framework.
2. We obtain the 2D-DP problems and recursively (when a processor succeeds we start again from the beginning of the item): (a) Decision point between the Basic Processors (check whether the system is trivially operationally terminating), P_{SCC} or shift to the Dependency Pair Framework (if all rules and pairs have no conditional part); (b) $\mathsf{P}_{\triangleright}$; (c) P_{Inf}; (d) P_{RTN} with linear polynomials (LPoly) and coefficients in $N_2 = \{0, 1, 2\}$; (e) Semantic version of P_{RTN} with convex domains and integer coefficients in $\mathbb{Z}$; (f) Semantic version of P_{RT} with

[13] `http://zenon.dsic.upv.es/muterm/?page_id=82`

[14] `http://termination-portal.org/wiki/TPDB`

convex domains and integer coefficients in $\mathbb{Z}$; (g) P_{RTN} with 2-square matrices with entries in $N_1 = \{0, 1\}$; (h) P_{IR} with LPoly and coefficients in N_2; (i) P_{SUC}; (j) P_{NC}; (k) P_{SUNC}; (l) P_{NQ}; and (m) P_{NR}.
3. If the techniques above fail, we apply the usual unravelling transformation $\mathcal{U}$ for 3-CTRSs (see, e.g, [29, Definition 7.2.48]) and shift to the Dependency Pair Framework.

Interestingly, *all* processors are used at least once during the proofs: P_{SCC} is used 281 times, $\mathsf{P}_{\triangleright}$ is used 47 times, P_{RTN} and P_{RT} are used 30 times, P_{IR} is used once, P_{Inf} is used 17 times, P_{SUC} is used once, P_{NR} and P_{NC} are used 4 times, P_{SUNC} is used once, P_{NQ} is used twice. We shifted to the DP framework and applied transformation $\mathcal{U}$ only once. As occurs in the DP framework, $\mathsf{P}_{\triangleright}$ and P_{RT} (and its variants) are the most successful processors to prove finiteness of CTRS problems. When those processors fail, the processors based on narrowing pairs, rules and conditions are very useful, because they make progress on finding a proof in two directions: extracting possible loops on infinite problems that can be captured by P_{Inf} and unrolling paths in the graph that allow to simplify the input problem.

5.2 Plugging schemes in practice

When open CTRS processors are applied to open CTRS problems, a set of different plugging schemes can be chosen to construct the proof tree. This selection raises the question of which strategy is recommended with respect to plugging schemes in practice.

If we are interested in a particular proof, we can guide the proof by applying processors with plugging schemes compatible with our goal. For example, if we are interested on the operational nontermination of the input problem, we would choose processors requiring plugging schemes whose completeness component is not $\bullet$.

With respect to the processors we present in this paper and in [27], the strategy used to choose a particular plugging scheme is simple, because we have either processors with one plugging scheme or processors whose plugging schemes can be inferred from its application conditions. For example, let's consider P_{SUC} and its possible plugging schemes $\mathit{\Pi}(\mathsf{P}_{SUC}) = \{\langle f, f\rangle, \langle f, \mathsf{a}\rangle, \langle \bullet, f\rangle, \langle \bullet, \mathsf{a}\rangle\}$. Given an open CTRS problem, the plugging scheme inferred after applying P_{SUC} is:

1. if s_i is linear, $\mathit{NRules}(\mathcal{R}, s_i) = \emptyset$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(r) = \emptyset$, for all $s \to t \in c[\diamond]_i$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(s) = \emptyset$, and for all $s \to t \in c$, $\mathcal{V}ar(s_i) \cap \mathcal{V}ar(t) = \emptyset$, then
   (a) if $\alpha \notin \mathcal{R}$, or $\alpha \in \mathcal{R}_F$, we use $\langle f, f\rangle$.
   (b) else we use $\langle f, \mathsf{a}\rangle$.
2. else
   (a) if $\alpha \notin \mathcal{R}$, we use $\langle \bullet, f\rangle$
   (b) else we use $\langle \bullet, \mathsf{a}\rangle$.

Notice that if there is more than one suitable plugging scheme applicable, we can always choose a plugging schema that is more general. In the future, if we define processors where disjoint plugging schemes are applicable, a more complex strategy should be used.

5.3 Basic and new processors

In this section, we show the impact of the new processors presented in the paper. We compare two versions of MU-TERM: a *basic* version with the processors described in [27] only, and a *new*, extended version which also includes the processors and techniques presented in this paper.

The strategy is the same as the one described in the previous subsection. In the basic version the processors presented in this article are disabled and P_{RT} is used instead of P_{RTN}. In the extended version, P_{RTN} uses two sorts to distinguish pair and rule symbols (see Section 4.4.1).

Table 3 New processors Comparison via 117 examples

Tool Version	Proved (YES/NO)	Av. YES	Av. NO
MU-TERM **Basic**	91 (77/14)	0.38s	0.96s
MU-TERM **New**	102 (85/17)	1.09s	0.98s

Full details about the experiments reported in Table 3 can be found here:

`http://zenon.dsic.upv.es/muterm/benchmarks/jar-19/`

With the implementation of the new techniques we are able to prove 11 more examples, which are essential in the termination competition to be the most successful tool for proving operational termination of CTRSs. Two examples (labeled `logic` and `ohl`) were solved in the 2018 Termination Competition, but not by MU-TERM *New*. The reason is the use (in the termination competition) of a now *deprecated* processor which transforms CTRS problems into DP problems from the DP Framework for TRSs under some conditions [23, Theorem 8]. Indeed, we are able to solve these two examples by using our new techniques (without any transference to the DP Framework for TRSs). For instance, `ohl` is the CTRS $\mathcal{R}$ in Example 33, which is proved operationally terminating by using P_{RT}. We can similarly deal with `logic` as well. This explains why we now handle 104 examples.

5.4 Comparing direct and transformational approaches

In this section, we compare the actual version of MU-TERM against two tranformational versions: a version of MU-TERM that applies the transformation $\mathcal{U}$ to the initial CTRS transforming it into a TRS and proving its termination using the DP framework, and a version of MU-TERM that applies the context-sensitive transformation $\mathcal{U}_{cs}$ to the initial CTRS transforming it into a CS-TRS and proving its termination using the CSDP framework.

Results are presented in Table 4. Full details about the experiments reported in the table can be found here:

`http://zenon.dsic.upv.es/muterm/benchmarks/jar-19/`

Performance of both transformational approaches is very similar, the difference in one example comes from a processor that is implemented in the DP framework

Table 4 Direct and Transformational Approaches via 117 examples

Tool Version	Proved (YES/NO)	Av. YES	Av. NO
MU-TERM	106 (89/17)	9.96s	31.79s
MU-TERM $\mathcal{U}$	83 (83/0)	1.53s	0.0s
MU-TERM $\mathcal{U}_{cs}$	82 (82/0)	0.06s	0.0s

but not extended to the CSDP framework yet. With regard to the direct version, apart from the posibility of disproving operational termination, the direct approach shows how P_{RT} and its variants are successfully applied to CTRS problems where the transformed DP problems and CS problems fail. Furthermore, narrowing is crucial to prove [30, Example 19].

6 Conclusion

We briefly explain the new contributions of this paper, also in connection with some related work.

In Section 4.1 we use the conditional narrowing introduced in [27, Section 7.5] to define processor P_{NC}, which can be used to refine the *conditional part* of the pairs occurring in a CTRS problem. In this way, the reachability conditions in the rules are made more 'precise' with regard to the underlying rules that are used to evaluate them. The new definition is different from the one in [26, Section 6.2], which was actually buggy.[15]

In Section 4.2 we have introduced processor P_{SUC} which is useful to simplify the conditional part of the rules and pairs of an OCTRS problem by *removing* conditions $s \to t$ whose components s and t unify. We have introduced a processor P_{SUNC} which combines the transformations introduced by P_{NC} and P_{SUC}. With these processors we can eventually *remove* the entire conditional part of rules or pairs.

In Section 4.3 we have shown how the use of P_{RT} in [27, Section 7.3] can be transformed into a (many-sorted, first-order) *satisfiability problem*, which can be mechanized by using existing model generation tools. The many-sorted logic-based treatment in Section 4.3 and 4.4.1 is new (with regard to the usual *algebraic* treatment, see below) and could be used in any implementation of the corresponding processor in the DP Framework for TRSs [11,12]. In [9], the automated generation of monotone many-sorted *algebras* to use reduction pairs $(\succsim, \succ)$ together with the reduction pair processor (with usable rules) is considered. However, only interpretations of *function* symbols are synthesized. Sorts P and S are interpreted as the set $\mathbb{N}$ of natural numbers (finite subsets are not considered) and the set $\mathbb{N}^d$ of tuples of natural numbers (for some $d > 0$), respectively; the components $\succsim$ and $\succ$ of the reduction pair (which we would treat as predicate symbols $\pi_{\succsim}$ and $\pi_{\succ}$ which can be given an interpretation satisfying an underlying theory) are based on the usual orderings $\geq$ and $>$ on (tuples of) natural numbers, see [9, Section 5]. The flexibility of our approach is useful to capture examples which would otherwise

[15] The use of P_{NC} in [26, Section 6.2] with the CTRS in Example 15 would lead to a wrong proof of operational termination. The corrected version of P_{NC} which is presented in this paper, though, is the one implemented in the Termination Competition version of MU-TERM.

P	$\Pi(\mathsf{P})$	P	$\Pi(\mathsf{P})$
P_{NC}	$\{\langle f, f\rangle, \langle \bullet, f\rangle\}$	P_{SUC}	$\{\langle f, f\rangle, \langle f, \mathsf{a}\rangle, \langle \bullet, f\rangle, \langle \bullet, \mathsf{a}\rangle\}$
P_{SUNC}	$\{\langle f, f\rangle, \langle \bullet, f\rangle\}$	P_{RTN}	$\{\langle \mathsf{m}, f\rangle\}$

Table 5 Plugging schemes for the new processors

be out of reach. For instance, the proof in Example 33 requires the use of the *equality* predicate (not considered in [9], for instance) which we do *not* impose or select. Instead, it is automatically obtained by the corresponding model generator. Although the use of finite algebras (i.e., based on finite domains) is not precluded by the theoretical approaches to the use of the reduction pair processor based on the automatic generation of algebras (see, e.g., [6]), this is also missing in most implementations due to the emphasis in using algebras over the whole set of natural numbers together with specific families of functions (e.g., polynomials over the naturals, linear mappings based on matrices over the naturals, etc.) for the interpretation of function symbols as mappings. For instance, a polynomial expression like $x + 1$ is well-defined (as a mapping) over $\mathbb{N}$ but fails to be a mapping if x ranges on a finite subset of $\mathbb{N}$.

In Section 4.4 we have introduced a new processor P_{RTN} which is a refinement of P_{RT}, where *two* subsets of rules from $\mathcal{R}$ are considered to model the *connection* between pairs and the *evaluation* of the conditions in pairs or rules. We have also shown how to deal with the application of this processor as a (many-sorted, first-order) *satisfiability problem*. As discussed in Section 4.4.2, the ability to encode the application of these processors as a satisfiability problem introduces new avenues of application for both of them and shows that, depending on the particular implementation of the processors, their priority in the proof strategy which is applied in a termination tool may differ.

Table 5 summarizes the five new processors introduced in this paper together with their plugging schemes.

As discussed in Section 5, the 2D DP Framework has been implemented as part of the tool MU-TERM. Since 2014, we have participated in the yearly *International Termination Competition*, each year obtaining the first position among the tools in the *TRS Conditional* subcategory, see

`http://zenon.dsic.upv.es/muterm/?page_id=82`

for a summary. Still, the new processors and implementation techniques introduced here improve over the results obtained from the processors introduced in [27]. Our running examples (the CTRSs $\mathcal{R}$ in Figure 1 and Example 23) could not be handled by using processors in [27] only; we also failed to handle them with AProVE [10]. In this paper we have provided proofs of operational (non-)termination for all of them, thanks to the use of the different results in this paper (see Figures 4 and 5).

All these techniques are available for tools like MTT [8], which may use MU-TERM as a backend for achieving proofs of operational termination of more general theories like membership equational programs or order-sorted rewrite theories (like those used in Maude [5]) by means of *transformations* to CTRSs. However, an interesting subject for future work is the integration into a unified framework of the 2D DP Framework together with other dependency pair frameworks which are useful to capture specific 'components' (sorts, equational theories, context-sensitivity,

etc.) which may also play a role in the termination behavior of such sophisticated programs.

Acknowledgments We thank María Alpuente for her suggestions and comments. We thank the anonymous referees for many remarks and suggestions that helped us to improve the paper. In particular, Section 3 benefitted from comments and suggestions by one of the referees; Example 19 was suggested by another referee.

References

1. T. Arts and J. Giesl. Termination of Term Rewriting Using Dependency Pairs. *Theoretical Computer Science*, 236(1–2):133–178, 2000.
2. B. Alarcón, R. Gutiérrez, S. Lucas, R. Navarro-Marset. Proving Termination Properties with MU-TERM. In *Proc. of AMAST'10*, LNCS 6486:201-208, 2011.
3. F. Baader and T. Nipkow. Term Rewriting and All That. Cambridge University Press, 1998.
4. J. Barwise. An Introduction to First-Order Logic. In J. Barwise, editor, Handbook of Mathematical Logic, North-Holland, 1977.
5. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. All About Maude – A High-Performance Logical Framework. LNCS 4350, Springer-Verlag, 2007.
6. E. Contejean, C. Marché, A.-P. Tomás, and X. Urbain. Mechanically proving termination using polynomial interpretations. *Journal of Automated Reasoning*, 34(4):325-363, 2006.
7. N. Dershowitz. A note on simplification orderings. *Information Processing Letters*, 9(5):212-215, 1979.
8. F. Durán, S. Lucas, and J. Meseguer. MTT: The Maude Termination Tool (System Description). In *Proc. of IJCAR'08*, LNAI 5195:313–319, 2008.
9. J. Endrullis, J. Waldmann, and H. Zantema. Matrix Interpretations for Proving Termination of Term Rewriting. *Journal of Automated Reasoning* 40(2-3):195-220, 2008.
10. J. Giesl, P. Schneider-Kamp, and R. Thiemann. AProVE 1.2: Automatic Termination Proofs in the Dependency Pair Framework. In *Proc. of IJCAR'06*, LNAI 4130:281–286, 2006.
11. J. Giesl, R. Thiemann, and P. Schneider-Kamp. The Dependency Pair Framework: Combining Techniques for Automated Termination Proofs. In *Proc. of LPAR'04*, LNAI 3452:301–331, 2004.
12. J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Mechanizing and Improving Dependency Pairs. *Journal of Automatic Reasoning*, 37(3):155–203, 2006.
13. J. Goguen and J. Meseguer. Models and Equality for Logical Programming. In *Proc. of TAPSOFT'87*, LNCS 250:1-22, 1987.
14. R. Gutiérrez and S. Lucas. Automatic Generation of Logical Models with AGES. In *Proc. of CADE 2019*, LNCS 11716:287-299, 2019. Tool page: `http://zenon.dsic.upv.es/ages/`.
15. N. Hirokawa and A. Middeldorp. Dependency Pairs Revisited. In *Proc. of RTA'04*, LNCS 3091:249–268, 2004.
16. W. Hodges. Elementary Predicate Logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic* Volume 1, pages 1-131. Reidel Publishing Company, 1983.
17. D.S. Lankford. On proving term rewriting systems are noetherian. Technical Report, Louisiana Technological University, Ruston, LA, 1979.
18. S. Lucas. Using Well-Founded Relations for Proving Operational Termination. *Journal of Automated Reasoning*, to appear, 2020. doi:10.1007/s10817-019-09514-2.
19. S. Lucas and R. Gutiérrez. Automatic Synthesis of Logical Models for Order-Sorted First-Order Theories. *Journal of Automated Reasoning*, 60(4):465–501, 2018.
20. S. Lucas and R. Gutiérrez. Use of logical models for proving infeasibility in term rewriting. *Information Processing Letters*, 136:90-95, 2018.
21. S. Lucas, C. Marché, and J. Meseguer. Operational termination of conditional term rewriting systems. *Information Processing Letters* 95:446–453, 2005.
22. S. Lucas and J. Meseguer. Models for Logics and Conditional Constraints in Automated Proofs of Termination. In Proc. of *AISC'14*, LNAI 8884:9-20, 2014.

23. S. Lucas and J. Meseguer. 2D Dependency Pairs for Proving Operational Termination of CTRSs. In S. Escobar, editor, *Proc. of the 10th International Workshop on Rewriting Logic and its Applications, WRLA'14*, LNCS 8663:195-212, 2014.
24. S. Lucas and J. Meseguer. Dependency pairs for proving termination properties of conditional term rewriting systems. *Journal of Logical and Algebraic Methods in Programming*, 86:236-268, 2017.
25. S. Lucas and J. Meseguer. Normal forms and normal theories in conditional rewriting. *Journal of Logical and Algebraic Methods in Programming*, 85(1):67-97, 2016.
26. S. Lucas, J. Meseguer, and R. Gutiérrez. Extending the 2D DP Framework for Conditional Term Rewriting Systems. In Selected papers from *LOPSTR'14*, LNCS 8981:113-130, 2015.
27. S. Lucas, J. Meseguer, and R. Gutiérrez. The 2D Dependency Pair Framework for conditional rewrite systems. Part I: Definition and basic processors. *Journal of Computer Systems Sciences* 96:74–106, 2018.
28. W. McCune Prover9 & Mace4. `http://www.cs.unm.edu/~mccune/prover9/`, 2005–2010.
29. E. Ohlebusch. *Advanced Topics in Term Rewriting*. Springer-Verlag, Apr. 2002.
30. F. Schernhammer and B. Gramlich. Characterizing and proving operational termination of deterministic conditional term rewriting systems. *Journal of Logic and Algebraic Programming* 79:659-688, 2010.
31. T. Sternagel and A. Middeldorp. Conditional Confluence (System Description). In *Proc. of RTA-TLCA'14*, LNCS 8560:456-465, 2014.
32. T. Sternagel and A. Middeldorp. Infeasible Conditional Critical Pairs. In *Proc. of IWC'15*, pages 13–18, 2014.
33. R. Thiemann. The DP Framework for Proving Termination of Term Rewriting. PhD Thesis, RWTH Aachen, Technical Report AIB-2007-17, 2007.
34. R. Thiemann, J. Giesl, and P. Schneider-Kamp. Improved Modular Termination Proofs Using Dependency Pairs. In *Proc. of IJCAR'04*, LNAI 3097:75–90, 2004.
35. H. Wang. Logic of many-sorted theories. *Journal of Symbolic Logic* 17(2):105-116, 1952.

A Use of $\mathbf{P}_{RT}$ with AGES (Example 23)

AGES *specification*

```
mod SchGra10_Example17_tH3 is
  sort S .
  ops a b c : -> S .
  ops f h : S -> S .
  op g : S S -> S .
  vars x y : S .
  rl a => h(b) .
  rl a => h(c) .
  crl f(x) => y if a => h(y) .
  crl g(x,b) => g(f(c),x) if f(b) => x /\ x => c .

  sort P .
  op G : S S -> P .

  op redR : P P -> Bool .  *** ->_R for sort P
  op redR : S S -> Bool .  *** ->_R for sort S
  op redsR : S S -> Bool . *** ->*_R

  op r : P P -> Bool . *** \gtrsim
  op wfr : P P -> Bool [wellfounded] . *** \sqsupset
endm
```

AGES *goal*

```
*** S^RT_Rel
r(x:P,y:P) /\ wfr(y:P,z:P) => wfr(x:P,z:P)

*** S^RT_CRel,alpha
  redR(x:P,y:P) => r(x:P,y:P)

*** S^RT_R :
*** Reflexivity and transitivity
  redsR(x:S,x:S)
  redR(x:S,y:S) /\ redsR(y:S,z:S) => redsR(x:S,z:S)

*** Congruence
  redR(x:S,y:S) => redR(f(x:S),f(y:S))
  redR(x:S,y:S) => redR(h(x:S),h(y:S))
  redR(x:S,y:S) => redR(g(x:S,z:S),g(x:S,z:S))
  redR(x:S,y:S) => redR(g(z:S,x:S),g(z:S,y:S))
  redR(x:S,y:S) => redR(G(x:S,z:S),G(x:S,z:S))
  redR(x:S,y:S) => redR(G(z:S,x:S),G(z:S,y:S))

*** Rewrite rules
  redR(a,h(b))
  redR(a,h(c))
  redsR(a,h(y:S)) => redR(f(x:S),y:S)
  (redsR(f(b),x:S) /\ redsR(x:S,c)) => redR(g(x:S,b),g(f(c),x:S))

*** Removing target pair:

redsR(f(b),c) => wfr(G(f(b),b),G(f(c),f(b)))
```

AGES *output*

```
Domains:
S: {-1 , 0 , 1}
```

```
P: |N \ {0}

Function Interpretations:
|[a]| =  - 1
|[f(x_1_1:S)]| =  - x_1_1:S
|[g(x_1_1:S,x_2_1:S)]| = 0
|[G(x_1_1:S,x_2_1:S)]| = 6+5.x_1_1:S
|[b]| =  - 1
|[c]| = 0
|[h(x_1_1:S)]| = 0

Predicate Interpretations:
r(x_1_1:P,x_2_1:P) <=> (x_1_1:P >= x_2_1:P)
redR(x_1_1:S,x_2_1:S) <=> (0 >= 0)
redR(x_1_1:P,x_2_1:P) <=> (x_1_1:P >= x_2_1:P)
redsR(x_1_1:S,x_2_1:S) <=> (1 + x_1_1:S >= 0)
wfr(x_1_1:P,x_2_1:P) <=> (x_1_1:P >= 1 + x_2_1:P)
```

B Proof of Theorem 5

First, we introduce a slight generalization of the notion of O-chain in Definition 1.

Definition 13 Let $\mathcal{P}, \mathcal{Q}, \mathcal{R}$, and $\mathcal{S}$ be CTRSs. A $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{S})$-O-chain is a finite or infinite sequence of (renamed) rules $u_i \to v_i \Leftarrow c_i \in \mathcal{P}$, which are viewed as *conditional dependency pairs*, together with a substitution σ satisfying that, for all $i \geq 1$, (i) for all $s \to t \in c_i$, $\sigma(s) \to_{\mathcal{R}}^* \sigma(t)$ and (ii) $\sigma(v_i)(\to_{\mathcal{S},\mathcal{R}} \cup \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}})^* \sigma(u_{i+1})$. A $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{S})$-O-chain is called *minimal* if for all $i \geq 1$, whenever

$$\sigma(v_i) = w_{i1}(\to_{\mathcal{S},\mathcal{R}}^* \circ \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}}) \cdots (\to_{\mathcal{S},\mathcal{R}}^* \circ \overset{\Lambda}{\to}_{\mathcal{Q},\mathcal{R}})\, w_{im_i} \to_{\mathcal{S},\mathcal{R}}^* \sigma(u_{i+1}),$$

in the chain, then for all j, $1 \leq j \leq m_i$, w_{ij} is $\mathcal{R}$-operationally terminating.

Clearly, a (minimal) $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-O-chain can be viewed as a (minimal) $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{R})$-O-chain. Given a CTRS problem $\check{\tau} = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, \varphi)$, we prove that for every infinite minimal $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{R})$-O-chain we can construct an infinite $(\mathcal{P}, \mathcal{Q}, \mathcal{N}^c \cup \mathcal{C}_\varepsilon^{\mathcal{R}}, \mathcal{N}^{\to} \cup \mathcal{C}_\varepsilon^{\mathcal{N}})$-O-chain, where $\mathcal{N}^c \subseteq \mathcal{R}$ can be different from $\mathcal{N}^{\to} \subseteq \mathcal{R}$ is some cases. First, we define the following interpretation:

Definition 14 (Interpretation) Let $\mathcal{R} = (\mathcal{F}, R)$ be a CTRS and $\mathcal{N} \subseteq \mathcal{R}$. Let $>$ be an arbitrary total ordering on terms in $\mathcal{T}(\mathcal{F} \cup \{\bot, \mathsf{c}^\Delta\}, \mathcal{X})$, where $\bot$ is a fresh constant symbol and c^Δ is a fresh binary symbol. The interpretation $\Phi_{\mathcal{R},\mathcal{N}}^\Delta$ is a mapping from operationally terminating terms in $\mathcal{T}(\mathcal{F}, \mathcal{X})$ to terms in $\mathcal{T}(\mathcal{F} \cup \{\bot, \mathsf{c}^\Delta\}, \mathcal{X})$ defined as follows:

1. $\Phi_{\mathcal{R},\mathcal{N}}^\Delta(x) = x$ if $x \in \mathcal{X}$,
2. $\Phi_{\mathcal{R},\mathcal{N}}^\Delta(f(t_1, \ldots, t_k)) = f(\Phi_{\mathcal{R},\mathcal{N}}^\Delta(t_1), \ldots, \Phi_{\mathcal{R},\mathcal{N}}^\Delta(t_k))$ if there is no rule in $\ell \to r \Leftarrow c \in \mathcal{R} - \mathcal{N}$ such that $f(\mathrm{ECAP}_{\mathcal{R}}(t_1), \ldots, \mathrm{ECAP}_{\mathcal{R}}(t_k))$ unifies with ℓ with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible,
3. $\Phi_{\mathcal{R},\mathcal{N}}^\Delta(f(t_1, \ldots, t_k)) = \mathsf{c}^\Delta(f(\Phi_{\mathcal{R},\mathcal{N}}^\Delta(t_1), \ldots, \Phi_{\mathcal{R},\mathcal{N}}^\Delta(t_k)), t')$ otherwise.

where $t' = \mathit{order}\left(\{\Phi_{\mathcal{R},\mathcal{N}}^\Delta(u) \mid t \to_{\mathcal{R}} u\}\right)$

$$\mathit{order}(T) = \begin{cases} \bot, & \text{if } T = \emptyset \\ \mathsf{c}^\Delta(t, \mathit{order}(T - \{t\})) & \text{if } t \text{ is minimal in the totally ordered set } (T, >) \end{cases}$$

The interpretation $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}$ does not generate an infinite term when the input term is operationally terminating.

Lemma 2 *Let $\mathcal{R} = (\mathcal{F}, R)$ be a CTRS, $\mathcal{N} \subseteq \mathcal{R}$ and t in $\mathcal{T}(\mathcal{F}, \mathcal{X})$. If t is $\mathcal{R}$-operationally terminating then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t)$ is a finite term.*

Proof Suppose not. Without loss of generality we choose a t minimal, i.e. $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t)$ is an infinite term but $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t')$ is a finite term for every proper subterm t' of t. By cases:

1. If $t = x \in \mathcal{X}$, the $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(x) = x$, yielding a contradiction.
2. If $t = f(t_1, \ldots, t_k)$ and there is no rule in $\ell \to r \Leftarrow c \in \mathcal{R} - \mathcal{N}$ such that ℓ and $f(\text{ECAP}_{\mathcal{R}}(t_1), \ldots, \text{ECAP}_{\mathcal{R}}(t_k))$ unify with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t) = f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t_1), \ldots, \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t_k))$. By the minimality assumption, for every $1 \leq i \leq n$, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t_i)$ is finite, yielding again a contradiction.
3. If $t = f(t_1, \ldots, t_k)$ and there is $\ell \to r \Leftarrow c \in \mathcal{R} - \mathcal{N}$ such that the lhs ℓ and $f(\text{ECAP}_{\mathcal{R}}(t_1), \ldots, \text{ECAP}_{\mathcal{R}}(t_k))$ unify with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t) = \mathsf{c}^{\Delta}(s, s')$, where $s = f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t_1), \ldots, \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t_k))$. By the minimality assumption, for every $1 \leq i \leq n$, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(t_i)$ is finite and s is finite. We have to analyze s'. We have two possibilities:
   - There are infinite rewritings starting from t, which implies t is not operationally terminating, leading to a contradiction.
   - There is a term u such that $t \to_{\ell \to r \Leftarrow c} u$, $\ell \to r \Leftarrow c \in \mathcal{R}$ and $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(u)$ is infinite. We know that t is operationally terminating, t is V-terminating (conditions do not generate infinite computations), u is operationally terminating and $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(u)$ is infinite. We can find a subterm u' of u such that $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(u')$ is infinite and u' is minimal. Since any other option generates a finite term for $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(u')$, the only possibility is infinitely iterate over this step, but, if this occur, we can construct an infinite sequence of the form:

     $$t \to_{\mathcal{R}} u \trianglerighteq u' \to_{\mathcal{R}} v \trianglerighteq v' \to_{\mathcal{R}} \cdots$$

     contradicting the operational termination of t.

□

If we have a substitution σ such that for all $x \in \mathcal{D}om(\sigma)$, $\sigma(x)$ is operationally terminating, we can construct a substitution $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}$.

Definition 15 Let $\mathcal{R}$ be a CTRS, $\mathcal{N} \subseteq \mathcal{R}$ and σ an $\mathcal{R}$-operationally terminating substitution. We denote by $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}$ a substitution that replaces occurrences of $x \in \mathcal{D}om(\sigma)$ by $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(x))$.

We can extract the following properties from $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}$.

Lemma 3 (Properties of $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}$) *Let $\mathcal{R} = (\mathcal{F}, R)$ be a CTRS, $\mathcal{N} \subseteq \mathcal{R}$, $s, t, \sigma(s), \sigma(t) \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ be $\mathcal{R}$-operationally terminating terms and σ an $\mathcal{R}$-operationally terminating substitution. We have:*

1. *If $\mathsf{Needed}_{\mathcal{R}_F}(t) \subseteq \mathcal{N}$ then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t)) = \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$.*

2. $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$.
3. *Assuming that whenever* $\ell' \to r' \Leftarrow c' \in \mathcal{N}$, *we have* $\mathsf{Needed}_{\mathcal{R}_F}(r') \subseteq \mathcal{N}$ *and* $\mathsf{Needed}_{\mathcal{R}_F}(s'_i) \subseteq \mathcal{N}$, *where* $s'_i \to t'_i \in c'$. *If* $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ *and* $\mathsf{Needed}_{\mathcal{R}_F}(s) \subseteq \mathcal{N}$ *then* $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^*_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$.

Proof 1. Similar to Item 2.

2. By structural induction on t:
   - If t is a variable x then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(x)) = \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(x)$.
   - If $t = f(t_1, \ldots, t_k)$ then:
     - if there is no rule in $\ell \to r \Leftarrow c \in \mathcal{R} - \mathcal{N}$ such that $f(\textsc{ECap}_{\mathcal{R}}(t_1), \ldots, \textsc{ECap}_{\mathcal{R}}(t_k))$ unifies with ℓ with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, then
     $$\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t)) = f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t_1)), \ldots, \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t_k))).$$
     By the I.H., we have $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t_i)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t_i)$, for all $1 \le i \le k$. Hence, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$;
     - if there is a rule in $\ell \to r \Leftarrow c \in \mathcal{R} - \mathcal{N}$ such that $f(\textsc{ECap}_{\mathcal{R}}(t_1), \ldots, \textsc{ECap}_{\mathcal{R}}(t_k))$ unifies with ℓ with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, then, for some t', we have
     $$\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t)) = \mathsf{c}^{\Delta}(f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t_1)), \ldots, \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t_k))), t')$$
     Using the $\mathcal{C}^{\Delta}_{\varepsilon}$-rule $\mathsf{c}^{\Delta}(x, y) \to x$, we obtain
     $$f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t_1)), \ldots, \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t_k))$$
     again and, therefore, the same conclusion $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$.

3. By induction on the length of the sequence, we have:
   - if $\sigma(s) = \sigma(t)$ then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t))$. By hypothesis, $\mathsf{Needed}_{\mathcal{R}_F}(s) \subseteq \mathcal{N}$ and, therefore, $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))$. By Lemma 3(2), $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(t)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$. Therefore, we get $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$ and, since $\mathcal{C}^{\Delta}_{\varepsilon} \subseteq \mathcal{N} \cup \mathcal{C}^{\Delta}_{\varepsilon}$, $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^*_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$.
   - if $\sigma(s) \to_{\mathcal{R}} \sigma(s') \to^*_{\mathcal{R}} \sigma(t)$, where p is the position of redex $\sigma(s)|_p$ in $\sigma(s) \stackrel{p}{\to}_{\{\ell\to r\Leftarrow c\},\mathcal{R}} \sigma(s')$. Without loss of generality, we can make $s' = y$, where y is a fresh new variable:
     - First assume that there is no position q such that $q \le p$, $\sigma(s)|_q = f(s_1, \ldots, s_k)$ and there is no rule in $\ell \to r \Leftarrow c \in \mathcal{R} - \mathcal{N}$ such that $f(\textsc{ECap}_{\mathcal{R}}(s_1), \ldots, \textsc{ECap}_{\mathcal{R}}(s_k))$ unifies with ℓ with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible. Applying Definition 14,
     $$\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)[\sigma(s)|_p]_p) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)|_p)]_p.$$
     Then, $\ell \to r \Leftarrow c \in \mathcal{N}$, $\sigma(s)|_p = \sigma(\ell)$, $\sigma(s)|_p$ is V-terminating and $\sigma(s')|_p = \sigma(r)$ for some substitution σ. By definition, $\mathsf{Needed}_{\mathcal{R}_F}(r) \subseteq \mathcal{N}$ and for each $s_i \to t_i \in c$, $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}$. Since $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)|_p) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(\ell))$, by Lemma 3 (2), we get $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(\ell)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(\ell)$. Now, for every $s_i \to t_i \in c$, we can apply the I.H. to $\sigma(s_j) \to^*_{\mathcal{R}} \sigma(t_j)$ ($\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}$ because $\ell \to r \Leftarrow c \in \mathcal{N}$ and, by hypothesis,

if $s_i \to t_i \in c$, $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}$), obtaining $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s_j) \to^*_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t_j)$. Furthermore, we have $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(r) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(r))$ (by hypothesis $\mathsf{Needed}_{\mathcal{R}_F}(r) \subseteq \mathcal{N}$ because $\ell \to r \Leftarrow c \in \mathcal{N}$). We have:

$$\begin{array}{l}\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)|_p)]_p = \\ \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(\ell))]_p \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(\ell)]_p \to^+_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \\ \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(r)]_p = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(r))]_p = \\ \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)[\sigma(r)]_p)\end{array}$$

and $\sigma(s') = \sigma(s)[\sigma(r)]_p$. Since $\mathsf{Needed}_{\mathcal{R}_F}(s) \subseteq \mathcal{N}$, by Lemma 3 (1), we have $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)) = \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s)$ and since $s' \in \mathcal{X}$, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s')) = \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s')$. Therefore $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^+_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s')$.

– Now consider the case where there is a position q such that $q \leq p$, $\sigma(s)|_q = f(s_1,\ldots,s_k)$ and there is a rule in $\ell \to r \Leftarrow c \in \mathcal{R}-\mathcal{N}$ such that ℓ and $f(\mathrm{ECAP}_{\mathcal{R}}(t_1),\ldots,\mathrm{ECAP}_{\mathcal{R}}(t_k))$ unify with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible. Applying Definition 14, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)[\sigma(s)|_q]_q) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)|_q)]_q$. By Definition 14,

$$\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s')|_q) \in \mathit{order}\left(\{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(u') \mid \sigma(s)|_q \to_{\mathcal{R}} u'\}\right).$$

By applying $\mathcal{C}^{\Delta}_{\varepsilon}$ rules, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)|_q) \to^+_{\mathcal{C}^{\Delta}_{\varepsilon}} \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s')|_q)$. We have:

$$\begin{array}{l}\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)|_q)]_q \to^+_{\mathcal{C}^{\Delta}_{\varepsilon}} \\ \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s))[\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s')|_q)]_q = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)[\sigma(s')|_q]_q)\end{array}$$

and $\sigma(s') = \sigma(s)[\sigma(s')|_q]_q$. Since $\mathsf{Needed}_{\mathcal{R}_F}(s) \subseteq \mathcal{N}$, by Lemma 3 (1), we have $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s)) = \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s)$ and since $s' \in \mathcal{X}$, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}(\sigma(s')) = \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s')$. Therefore $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^+_{\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s')$, but since $\mathcal{C}^{\Delta}_{\varepsilon} \subseteq \mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}$ then we can also write $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^+_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s')$.

Finally, applying again the I.H. to $\sigma(s') \to^*_{\mathcal{R}} \sigma(t)$ (note that $s' \in \mathcal{X}$, therefore $\mathsf{Needed}_{\mathcal{R}_F}(s') \subseteq \mathcal{N}$ is vacuously true), we get

$$\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^+_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s') \to^*_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$$

and, therefore, $\sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(s) \to^*_{\mathcal{N}\cup\mathcal{C}^{\Delta}_{\varepsilon}} \sigma_{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}}}(t)$.

□

Lemma 4 (Compatibility between needed rules) *Let $\mathcal{R} = (\mathcal{F}, R)$ be a CTRS, $\mathcal{N}^c, \mathcal{N}^{\to} \subseteq \mathcal{R}$ and $t \in \mathcal{T}(\mathcal{F},\mathcal{X})$ is $\mathcal{R}$-operationally terminating. If $\mathcal{N}^{\to} \subseteq \mathcal{N}^c$, then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t)$.*

Proof By structural induction on t:

– if $t \in \mathcal{X}$ then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t)$.
– if $t = f(t_1,\ldots,t_k)$ then:
  – if there is no rule $\ell \to r \Leftarrow c \in \mathcal{R}-\mathcal{N}^{\to}$ such that $f(\mathrm{ECAP}_{\mathcal{R}}(t_1),\ldots,\mathrm{ECAP}_{\mathcal{R}}(t_k))$ and ℓ unify with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, then we have $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t) = f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_1),\ldots,\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_k))$. By the I.H., $f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_1),\ldots,\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_k)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t_1),\ldots,\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t_k)) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t)$

– if there is no rule $\ell \to r \Leftarrow c \in \mathcal{R}{-}\mathcal{N}^c$ such that $f(\mathrm{ECAP}_{\mathcal{R}}(t_1),\ldots,\mathrm{ECAP}_{\mathcal{R}}(t_k))$ and ℓ unify with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, but there is $\ell \to r \Leftarrow c \in \mathcal{R} - \mathcal{N}^{\to}$ such that $f(\mathrm{ECAP}_{\mathcal{R}}(t_1),\ldots,\mathrm{ECAP}_{\mathcal{R}}(t_k))$ and ℓ unify with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, then $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_1),\ldots,\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_k))$. This is because, by Definition 14,

$$\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(f(t_1,\ldots,t_n)) = c(f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_1),\ldots,\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_k)),t').$$

By I.H., $f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_1),\ldots,\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t_k)) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} f(\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t_1),\ldots,\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t_k)) = \Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t)$.

– if there is a rule $\ell \to r \Leftarrow c \in \mathcal{R}{-}\mathcal{N}^c$ such that $f(\mathrm{ECAP}_{\mathcal{R}}(t_1),\ldots,\mathrm{ECAP}_{\mathcal{R}}(t_k))$ and ℓ unify with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible, $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t) = \mathit{order}(\{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(u) \mid t \to_{\mathcal{R}} u\})$ and $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t) = \mathit{order}(\{\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(u) \mid t \to_{\mathcal{R}} u\})$. By applying the I.H. to the $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(u)$ terms, we have $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(u) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(u)$ and, therefore, we can conclude $\Phi^{\Delta}_{\mathcal{R},\mathcal{N}^{\to}}(t) \to^*_{\mathcal{C}^{\Delta}_{\varepsilon}} \Phi^{\Delta}_{\mathcal{R},\mathcal{N}^c}(t)$.

□

Lemma 5 (Properties of $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}$ and $\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}$) *Let $\mathcal{R} = (\mathcal{F},R)$ be a CTRS, $\mathcal{N}^c,\mathcal{N}^{\to} \subseteq \mathcal{R}$, $t,u \in \mathcal{T}(\mathcal{F},\mathcal{X})$ be $\mathcal{R}$-operationally terminating. Furthermore, we assume that*

– *if a rule $\ell' \to r' \Leftarrow c' \in \mathcal{N}^{\to}$, then*
  1. $\mathsf{Needed}^{\to}_{\mathcal{R}_F}(r') \subseteq \mathcal{N}^{\to}$,
  2. $\mathsf{Needed}^{\to}_{\mathcal{R}_F}(s'_i) \subseteq \mathcal{N}^{\to}$ *if* $s'_i \to t'_i \in c'$ *and* $\mathit{fv}(t'_i)$ *does not hold,*
  3. $\mathsf{Needed}_{\mathcal{R}_F}(s'_i) \subseteq \mathcal{N}^c$ *if* $s'_i \to t'_i \in c'$ *and* $\mathit{fv}(t'_i)$ *holds;*
– *if a rule $\ell' \to r' \Leftarrow c' \in \mathcal{N}^c$, then*
  1. $\mathsf{Needed}_{\mathcal{R}_F}(r') \subseteq \mathcal{N}^c$,
  2. $\mathsf{Needed}_{\mathcal{R}_F}(s'_i) \subseteq \mathcal{N}^c$;
– *if there exists a a rule $\ell' \to r' \Leftarrow c' \in \mathcal{N}^{\to}$, $s'_i \to t'_i \in c$ and $\mathit{sfv}(t'_i)$ does not hold then $\mathcal{N}^{\to} \subseteq \mathcal{N}^c$; and,*
– *if there exists a a rule $\ell' \to r' \Leftarrow c' \in \mathcal{N}^{\to}$, $s'_i \to t'_i \in c$ and $\mathit{fv}(t'_i)$ does not hold then $\mathcal{C}_{\varepsilon}$ is used instead of $\mathcal{C}^{\mathcal{N}}_{\varepsilon}$ and $\mathcal{C}^{\mathcal{R}}_{\varepsilon}$ in the following result (and is* (93) *therefore removed).*

If $t \to^*_{\mathcal{R}} u$ then $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(t) \to^*_{\mathcal{N}^{\to}\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}\cup\{(93)\}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(u)$.

Proof We proceed by induction on the length of the sequence $t \to^*_{\mathcal{R}} u$.

– if $t = u$ then $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(t) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(u)$.
– if $t \xrightarrow{p}_{\mathcal{R}} t' \to^*_{\mathcal{R}} u$, where p is the position p of the redex $t|_p$ in $t \xrightarrow{p}_{\ell\to r\Leftarrow c} t'$:
  – First assume that there is no position q with $q \leq p$, $t|_q = f(t_1,\ldots,t_k)$ and rule $\ell' \to r' \Leftarrow c' \in \mathcal{R} - \mathcal{N}^{\to}$ such that $f(\mathrm{ECAP}_{\mathcal{R}}(t_1),\ldots,\mathrm{ECAP}_{\mathcal{R}}(t_k))$ and ℓ' unify with *mgu* θ and $\theta(c')$ is $\mathcal{R}$-feasible. Applying Definition 14, $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(t) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(t[t|_p]_p) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(t)[\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(t|_p)]_p$. We have $\ell \to r \Leftarrow c \in \mathcal{N}^{\to}$, $t|_p = \sigma(\ell)$, $t|_p$ is V-terminating and $t'|_p = \sigma(r)$ for some substitution σ. Moreover, by definition we have $\mathsf{Needed}^{\to}_{\mathcal{R}_F}(r) \subseteq \mathcal{N}^{\to}$ and for every $s_i \to t_i \in c$ we have $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^c$ if $\mathit{fv}(t_i)$ and $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^{\to}$, otherwise. Starting with $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(t|_p) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(\sigma(\ell))$, by Lemma 3 (2), we get $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}(\sigma(\ell)) \to^*_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\to}}}(\ell)$. Now, for every $s_i \to t_i \in c$, we consider three cases:

1. If $sfv(t_i)$ holds ($fv(t_i)$ also holds), then $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s_i) = \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^c}}(s_i) = \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(s_i) = s_i$ because s_i is ground. Since $sfv(t_i)$ implies $fv(t_i)$, by hypothesis $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^c$ (if $\ell \rightarrow r \Leftarrow c \in \mathcal{N}^{\rightarrow}$ then $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^c$ if $fv(t_i)$ holds). Since $\mathcal{N}^c$ satisfies the conditions of $\mathcal{N}$ in Lemma 3 (3) (thanks to the second item in the lemma), we can apply Lemma 3 (3), where $\mathcal{N} = \mathcal{N}^c$, getting $s_i \rightarrow^*_{\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(t_i)$. Note that t_i and r share no variables because $fv(t_i)$ holds.
2. If $fv(t_i)$ holds but $sfv(t_i)$ does not, by hypothesis $\mathcal{N}^{\rightarrow} \subseteq \mathcal{N}^c$ (there is a $s_i \rightarrow t_i \in c$ such that $sfv(t_i)$ does not hold). Therefore, by applying Rule (93) repeatedly to transform every occurrence of $\mathsf{c}^{\mathcal{N}}$ by $\mathsf{c}^{\mathcal{R}}$ (only when $\mathcal{C}^{\mathcal{N}}_{\varepsilon} \neq \mathcal{C}^{\mathcal{R}}_{\varepsilon}$, i.e. when there is no rule $\ell' \rightarrow r' \Leftarrow c' \in \mathcal{N}^{\rightarrow}$, $s'_i \rightarrow t'_i \in c$ and $fv(t'_i)$ does not hold) and Lemma 4, we obtain $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s_i) \rightarrow^*_{\{(93)\}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s_i) \rightarrow^*_{\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(s_i)$. Furthermore, $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^c$ (if $\ell \rightarrow r \Leftarrow c \in \mathcal{N}^{\rightarrow}$, then $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^c$ if $fv(t_i)$ holds). Since $\mathcal{N}^c$ satisfies the conditions of $\mathcal{N}$ in Lemma 3 (3) (thanks to the second item in the lemma), we can apply Lemma 3 (3), where $\mathcal{N} = \mathcal{N}^c$, getting $\sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(s_i) \rightarrow^*_{\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(t_i)$. Note that t_i and r share no variables because $fv(t_i)$ holds.
3. If $fv(t_i)$ (and, therefore, $sfv(t_i)$) does not hold, $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^{\rightarrow}$ (if $\ell \rightarrow r \Leftarrow c \in \mathcal{N}^{\rightarrow}$ then $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^{\rightarrow}$ if $fv(t_i)$ does not holds) and $\mathcal{C}^{\mathcal{N}}_{\varepsilon} = \mathcal{C}^{\mathcal{R}}_{\varepsilon} = \mathcal{C}_{\varepsilon}$. Since we have $\mathsf{Needed}_{\mathcal{R}_F}(s_i) \subseteq \mathcal{N}^{\rightarrow}$, then $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(s_i)) = \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s_i)$ by Lemma 3(1) and by the I.H., we have $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(s_i)) \rightarrow^*_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(t_i))$. Finally, by Lemma 3 (2), $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(t_i)) \rightarrow^*_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(t_i)$. Thus, $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s_i) \rightarrow^*_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(t_i)$. But, since $\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon} \subseteq \mathcal{N}^c \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}$ by hypothesis (there is a $s_i \rightarrow t_i \in c$ such that $sfv(t_i)$ does not hold), we can conclude that $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s_i) \rightarrow^*_{\mathcal{N}^c \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(t_i)$. Note that t_i and r can share variables, but they are instances of interpreted terms using $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}$, as desired.

Now, consider r. For every variable x in r, we know that $x \in \ell$ or $x \in t_i$ such that $s_i \rightarrow t_i \in c$ and $fv(t_i)$ (and also $sfv(t_i)$) does not hold. We obtain $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(r)$ and, by hypothesis ($\mathsf{Needed}_{\mathcal{R}_F}(r) \in \mathcal{N}^{\rightarrow}$), $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(r) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(r)) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t'|_p)$. We have:

$$\begin{array}{l}\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t)[\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t|_p)]_p = \\ \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t)[\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\ell))]_p \rightarrow^*_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t)[\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(\ell)]_p \rightarrow^+_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \\ \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t)[\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(r)]_p = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t)[\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(r))]_p = \\ \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t[\sigma(r)]_p)\end{array}$$

and $t' = t[\sigma(r)]_p$. Therefore, we have that $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t) \rightarrow^+_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t')$.

– Now consider the case where there is a position q such that $q \leq p$ and $t|_q = f(t_1, \ldots, t_k)$ and there is a rule in $\ell \rightarrow r \Leftarrow c \in \mathcal{R} - \mathcal{N}^{\rightarrow}$ such that $f(\text{ECAP}_{\mathcal{R}}(t_1), \ldots, \text{ECAP}_{\mathcal{R}}(t_k))$ unifies with ℓ with *mgu* θ and $\theta(c)$ is $\mathcal{R}$-feasible. By Definition 14, $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}}(t) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}}(t[t|_q]_q) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}}(t)[\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}}(t|_q)]_q$.

By Definition 14, we have $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t'|_q) \in \mathit{order}\left(\{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(u') \mid t|_q \rightarrow_{\mathcal{R}} u'\}\right)$. By applying $\mathcal{C}^{\mathcal{N}}_{\varepsilon}$ rules, we have $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t|_q) \rightarrow^{+}_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t'|_q)$. Thus:

$$\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t)[\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t|_q)]_q \rightarrow^{+}_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t)[\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t'|_q)]_q = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t[t'|_q]_q)$$

and $t' = t[t'|_q]_q$. Therefore $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t) \rightarrow^{+}_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t')$, but since $\mathcal{C}^{\mathcal{N}}_{\varepsilon} \subseteq \mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}$ and $\mathcal{C}^{\mathcal{N}}_{\varepsilon} \subseteq \mathcal{N}^{c} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}$, then $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t) \rightarrow^{+}_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^{c} \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t')$. Finally, applying the I.H. to $t' \rightarrow^{*}_{\mathcal{R}} u$ ($\mathcal{N}^{\rightarrow}$ and $\mathcal{N}^{c}$ do not change),

$$\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t) \rightarrow^{+}_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^{c} \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t') \rightarrow^{*}_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^{c} \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(u)$$

and, therefore, $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(t) \rightarrow^{*}_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^{c} \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(u)$.

□

Theorem 5. P_{RTN} *is* m*-sound and complete.*

Proof Completeness follows by Corollary 1. Regarding soundness, we proceed by contradiction. Assume that A is an infinite minimal $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{R})$-O-chain, but there is no infinite minimal $(\mathcal{P}[\emptyset]_{\alpha}, \mathcal{Q}[\emptyset]_{\alpha}, \mathcal{R}, \mathcal{R})$-O-chain. By finiteness of $\mathcal{P}$ and $\mathcal{Q}$, there are $\mathcal{P}' \subseteq \mathcal{P}$ and $\mathcal{Q}' \subseteq \mathcal{Q}$ such that A has a tail B that we can write as a reduction sequence as follows:

$$\sigma(u^1) \rightarrow_{\mathcal{P}',\mathcal{R}} \sigma(v^1)(\rightarrow_{\mathcal{R}} \cup \overset{\Lambda}{\rightarrow}_{\mathcal{Q}',\mathcal{R}})^{*}\sigma(u^2) \rightarrow_{\mathcal{P}',\mathcal{R}} \sigma(v^2)(\rightarrow_{\mathcal{R}} \cup \overset{\Lambda}{\rightarrow}_{\mathcal{Q}',\mathcal{R}})^{*}\sigma(u^3) \rightarrow_{\mathcal{P}',\mathcal{R}} \cdots$$

for some substitution σ, where all pairs in $\mathcal{P}'$ and $\mathcal{Q}'$ are used infinitely often and terms u^i, v^i, $i \geq 1$, are operationally terminating on $\mathcal{R}$.

Now, we consider the different steps with the different sets in the infinite sequence.

1. If $\sigma(u^i) \rightarrow_{\mathcal{P}',\mathcal{R}} \sigma(v^i)$ for $\alpha_i : u^i \rightarrow v^i \Leftarrow c^i \in \mathcal{P}'$, we have $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^i)) \rightarrow^{*}_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(u^i)$, by Lemma 3 (2). For all $s^i_j \rightarrow t^i_j \in c^i$, we must have $\sigma(s^i_j) \rightarrow^{*}_{\mathcal{R}} \sigma(t^i_j)$. We consider three cases:
   (a) If $\mathit{sfv}_{\alpha_i}(t^i_j)$ holds ($\mathit{fv}(t_i)$ also holds), then $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j) = \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{c}}}(s^i_j) = \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^{c}}}(s^i_j) = s^i_j$ because s^i_j is ground. Since $\mathit{sfv}(t^i_j)$ implies $\mathit{fv}(t^i_j)$, by definition of $\mathsf{NeedR}^{\mathsf{c}}$ (see (69)), we have that $\mathsf{Needed}_{\mathcal{R}_F}(s^i_j) \subseteq \mathsf{NeedR}^{\mathsf{c}}(\check{\tau})$. By item 1 in Definition 11, $\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \subseteq \mathcal{N}^{c}$. Then, $\mathsf{Needed}_{\mathcal{R}_F}(s^i_j) \subseteq \mathcal{N}^{c}$. By Lemma 3(3), where $\mathcal{N} = \mathcal{N}^{c}$, $s^i_j \rightarrow^{*}_{\mathcal{N}^{c} \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^{c}}}(t^i_j)$. Note that t^i_j and r share no variables because $\mathit{fv}(t^i_j)$ holds.
   (b) If $\mathit{fv}(t^i_j)$ holds but $\mathit{sfv}(t^i_j)$ does not hold, then, by definition of $\mathsf{NeedR}^{\mathsf{c}}$ (see (69)), we have that $\mathsf{Needed}_{\mathcal{R}_F}(s^i_j) \subseteq \mathsf{NeedR}^{\mathsf{c}}(\check{\tau})$. By Definition 11, item 1, $\mathsf{NeedR}^{\mathsf{c}}(\check{\tau}) \subseteq \mathcal{N}^{c}$. Therefore, $\mathsf{Needed}_{\mathcal{R}_F}(s^i_j) \subseteq \mathcal{N}^{c}$. If $\mathcal{C}^{\mathcal{N}}_{\varepsilon} \neq \mathcal{C}^{\mathcal{R}}_{\varepsilon} \neq \mathcal{C}_{\varepsilon}$, i.e. when there is no rule $\ell' \rightarrow r' \Leftarrow c' \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{N}^{\rightarrow}$, $s'_i \rightarrow t'_i \in c$ and $\mathit{fv}(t'_i)$

does not hold, applying Rule (93) repeatedly we transform every occurrence of $\mathsf{c}^{\mathcal{N}}$ by $\mathsf{c}^{\mathcal{R}}$, obtaining $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j) \rightarrow^*_{\{(93)\}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j)$; otherwise $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j) = \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j)$. Furthermore, $\mathcal{N}^{\rightarrow} \subseteq \mathcal{N}^c$ because there is a $s_i \rightarrow t_i \in c$ such that $sfv(t_i)$ does not hold. Since $\mathcal{N}^{\rightarrow} \subseteq \mathcal{N}^c$, applying Lemma 4, we get $\sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j) \rightarrow^*_{\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(s^i_j)$. Since $\mathsf{Needed}_{\mathcal{R}_F}(s^i_j) \subseteq \mathcal{N}^c$, by Lemma 3 (3), where $\mathcal{N} = \mathcal{N}^c$, we obtain $\sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(s^i_j) \rightarrow^*_{\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{R}}_{\mathcal{R},\mathcal{N}^c}}(t^i_j)$. Note that t^i_j and r share no variables because $fv(t^i_j)$ holds.

(c) If $fv(t^i_j)$ (and, therefore, $sfv(t_i)$) does not hold, by definition of $\mathsf{NeedR}^{\rightarrow}$ (see (68)), we have that $\mathsf{Needed}_{\mathcal{R}_F}(s^i_j) \subseteq \mathsf{NeedR}^{\rightarrow}(\check{\tau})$. By item 5 in Definition 11, we have $\mathcal{C}^{\mathcal{N}}_{\varepsilon} = \mathcal{C}^{\mathcal{R}}_{\varepsilon} = \mathcal{C}_{\varepsilon}$. By item 2 in Definition 11, $\mathsf{NeedR}^{\rightarrow}(\check{\tau}) \subseteq \mathcal{N}^{\rightarrow}$. Therefore, $\mathsf{Needed}_{\mathcal{R}_F}(s^i_j) \subseteq \mathcal{N}^{\rightarrow}$. Note that the conditions of Lemma 5 are fulfilled by item 3 in Definition 11. Then, by Lemma 5, $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(s^i_j)) = \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j) \rightarrow^*_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(t^i_j))$ and, by Lemma 3 (2),

$$\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(t^i_j)) \rightarrow^*_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(t^i_j),$$

and therefore $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(s^i_j) \rightarrow^*_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(t^i_j)$. Note that t^i_j and v^i_j can share variables, but they are instances of interpreted terms using $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}$, as we want.

Now, consider v^i_j. For every variable x in v^i_j, we know that $x \in u^i$ or $x \in t^i_j$ such that $s^i_j \rightarrow t^i_j \in c^i_j$ and $fv(t^i_j)$ (and also $sfv(t^i_j)$) does not hold. Therefore, we obtain $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(v^i_j)$ and, by Definition 11, $(\mathsf{Needed}_{\mathcal{R}_F}(v^i_j) \subseteq \mathsf{NeedR}^{\rightarrow}(\check{\tau}))$, $\sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(v^i_j) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^i_j))$.

2. If $\sigma(u^i_j) \rightarrow_{\mathcal{Q}',\mathcal{R}} \sigma(v^i_j)$ for $u^i_j \rightarrow v^i_j \Leftarrow c^i_j \in \mathcal{Q}'$, we can follow the exact same reasoning.
3. If $w^i_j \rightarrow^*_{\mathcal{R}} w^i_{j+1}$, by Lemma 5 (conditions of Lemma 5 are fulfilled by item 3 in Definition 11) we have $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(w^i_j) \rightarrow^+_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(w^i_{j+1})$.

Therefore, we can construct an infinite sequence of the form:

$$\begin{array}{lll}
\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^1)) & \rightarrow^*_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} & \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(u^1) \\
& \rightarrow_{\mathcal{P}',\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} & \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(v^1) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^1)) \\
& (\rightarrow^*_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \cup \overset{\Lambda}{\rightarrow}_{\mathcal{Q}',\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}})^* & \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^2)) \\
& \rightarrow^*_{\mathcal{C}^{\mathcal{N}}_{\varepsilon}} & \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(u^2) \\
& \rightarrow_{\mathcal{P}',\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} & \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(v^2) = \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^2)) \\
& (\rightarrow^*_{\mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon}, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} \cup \overset{\Lambda}{\rightarrow}_{\mathcal{Q}',\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}})^* & \sigma_{\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}}(u^3) \\
& \rightarrow_{\mathcal{P}',\mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}} & \cdots
\end{array}$$

Although $\mathcal{Q}'$ could be *empty* (if no pair in $\mathcal{Q}'$ is used to *connect* pairs in $\mathcal{P}'$) $\mathcal{P}'$ is not empty. By Definition 6.2, for all $i \geq 1$ and $u^i \rightarrow v^i \Leftarrow c^i \in \mathcal{P}'$,

$$\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^i))) \; (\gtrsim \cup \succeq \cup \sqsupseteq) \; \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}\sigma(v^i))) \tag{104}$$

Note that $\alpha \notin \mathcal{P}' \cup \mathcal{Q}'$. Otherwise, we get a contradiction as follows. Since

$$\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^i))(\rightarrow^*_{\mathcal{N}^{\rightarrow}\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \circ \xrightarrow{\Lambda}_{\overline{\mathcal{Q}}',\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}})^*\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^{i+1})),$$

there are pairs $\overline{u}^k_i \rightarrow \overline{v}^k_i \Leftarrow \bigwedge_{j=1}^{n_{ik}} \overline{u}^k_{ij} \rightarrow \overline{v}^k_{ij} \in \mathcal{Q}'$ for k, $1 \leq k \leq \kappa_i$ ($\kappa_i = 0$ indicates that no pair in $\mathcal{Q}'$ is necessary to connect $\sigma(v^i)$ and $\sigma(u^{i+1})$; if $\mathcal{Q}' = \emptyset$, then $\kappa_i = 0$ for all $i \geq 0$) such that $\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{u}^k_{ij})) \rightarrow^*_{\mathcal{N}^{\rightarrow}\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{v}^k_{ij}))$ for all j, $1 \leq j \leq n_{ik}$ and k, $1 \leq k \leq \kappa_i$, and

$$\begin{array}{l}
\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^i)) \rightarrow^*_{\mathcal{N}^{\rightarrow}\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{u}^1_i)) \xrightarrow{\Lambda}_{\mathcal{Q},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{v}^1_i)) \\
\rightarrow^*_{\mathcal{N}^{\rightarrow}\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \cdots \rightarrow^*_{\mathcal{N}^{\rightarrow}\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{u}^{\kappa_i}_i)) \xrightarrow{\Lambda}_{\mathcal{Q},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{v}^{\kappa_i}_i)) \\
\rightarrow^*_{\mathcal{N}^{\rightarrow}\cup\mathcal{C}^{\mathcal{N}}_{\varepsilon},\mathcal{N}^c\cup\mathcal{C}^{\mathcal{R}}_{\varepsilon}} \Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^{i+1}))
\end{array}$$

for $i \geq 1$. By Definition 6.1, for all k, $1 \leq k < \kappa_i$ we have

$$\begin{array}{c}
\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^i))) \gtrsim^* \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{u}^1_{i+1}))) \text{ and} \\
\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{v}^k_i))) \gtrsim^* \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{u}^{k+1}_i))) \qquad (105)
\end{array}$$

and $\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(\overline{v}^{\kappa_i}_i))) \gtrsim^* \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^{i+1})))$ (or $\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^i))) \gtrsim^* \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u_{i+1})))$ if $\kappa_i = 0$). Also,

$$\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^k_i))) \ (\gtrsim \cup \succeq \cup \sqsupseteq) \ \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(v^k_i))) \qquad (106)$$

for all k, $1 \leq k < \kappa_i$. By compatibility among $\gtrsim$, $\succeq$, and $\sqsupseteq$, from (104) and (106) we conclude that $\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^i))) \ (\gtrsim \cup \succeq \cup \sqsupseteq)^+ \ \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^{i+1})))$. Since α occurs infinitely often in B, there is an infinite set $\mathcal{J} \subseteq \mathbb{N}$ such that $\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^j))) \sqsupseteq \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^{j+1})))$ for all $j \in \mathcal{I}$. And we have $\pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^i))) \ (\gtrsim \cup \succeq \cup \sqsupseteq) \ \pi(\Phi^{\mathcal{N}}_{\mathcal{R},\mathcal{N}^{\rightarrow}}(\sigma(u^{i+1})))$ for all other $u^i \rightarrow v^i \Leftarrow c^i \in \mathcal{P}'$ with $i \in \mathbb{N} - \mathcal{J}$. Thus, by using the compatibility conditions of the removal triple, we obtain an infinite decreasing $\sqsupseteq$-sequence that contradicts the well-foundedness of $\sqsupseteq$.

Thus, $\mathcal{P}' \subseteq \mathcal{P}[\emptyset]_\alpha$ and $\mathcal{Q}' \subseteq \mathcal{Q}[\emptyset]_\alpha$, i.e., B is transformed into an infinite (minimal) $(\mathcal{P}[\emptyset]_\alpha, \mathcal{Q}[\emptyset]_\alpha, \mathcal{N}^c \cup \mathcal{C}^{\mathcal{R}}_{\varepsilon}, \mathcal{N}^{\rightarrow} \cup \mathcal{C}^{\mathcal{N}}_{\varepsilon})$-O-chain, leading to a contradiction. □