Termination of Generalized Term Rewriting Systems

Salvador Lucas
DSIC & VRAIN, Universitat Politècnica de València, Spain, Spain

Abstract

We investigate termination of *Generalized Term Rewriting Systems* (GTRSs), which extend *Conditional Term Rewriting Systems* by considering replacement restrictions on selected arguments of function symbols, as in *Context-Sensitive Rewriting*, and conditional rewriting rules whose conditional part may include not only a mix of the usual (reachability, joinability,...) conditions, but also atoms defined by a set of definite Horn clauses. GTRSs can be used to prove confluence and termination of *Generalized Rewrite Theories* and Maude programs. We have characterized confluence of terminating GTRSs as the joinability of a finite set of conditional pairs. Since termination of GTRSs is underexplored to date, this paper introduces a Dependency Pair Framework which is well-suited to automatically (dis)prove termination of GTRSs.

2012 ACM Subject Classification Theory of computation → Automated reasoning; Theory of computation → Logic and verification; Theory of computation → Equational logic and rewriting

Keywords and phrases Program Analysis, Reduction-Based Systems, Termination

Digital Object Identifier 10.4230/LIPIcs.FSCD.2024.32

Funding Supported by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe" (PID2021-122830OB-C42) and by the grant CIPROM/2022/6 funded by Generalitat Valenciana.

Acknowledgements I thank the reviewers for their useful comments leading to several improvements in the paper. I also thank Raúl Gutiérrez for his comments.

1 Introduction

Generalized Term Rewriting Systems (GTRS [21]) extend Conditional Term Rewriting Systems (CTRSs, see, e.g., [29, Chapter 7] and the references therein) by (i) restricting reductions on specific arguments of function symbols by means of a replacement map μ [17] and by also (ii) including atoms in the conditional part of rules which are defined by (iii) a set of definite Horn clauses. GTRSs and CTRSs are compared in [21, Section 7.3].

▶ **Example 1.** The following GTRS $\mathcal{R}$ to divide natural numbers in Peano's notation (adapted from [32, Example 9]), consists of clauses (1)-(5) and rules (6)-(10).

$$x \approx y \Leftarrow x \to^* y \quad (1) \qquad x - 0 \to x \quad (6)$$
$$s(x) > 0 \quad (2) \qquad 0 - y \to 0 \quad (7)$$
$$s(x) > s(y) \Leftarrow x > y \quad (3) \qquad \mathsf{s}(x) - \mathsf{s}(y) \to x - y \quad (8)$$
$$0 \le x \quad (4) \qquad \mathsf{div}(x, y) \to \mathsf{pair}(0, x) \Leftarrow y > x \quad (9)$$
$$s(x) \le s(y) \Leftarrow x \le y \quad (5)$$
$$\mathsf{div}(x, y) \to \mathsf{pair}(\mathsf{s}(q), r) \Leftarrow y \le x, \mathsf{div}(x - y, y) \approx \mathsf{pair}(q, r) \quad (10)$$

A call $\mathsf{div}(m, n)$ would return $\mathsf{pair}(q, r)$ with q and r the quotient and remainder.

Rewriting steps $s \to_{\mathcal{R}} t$ with GTRSs $\mathcal{R}$ are defined by *deduction* of goals $s \to t$ (where $\to$ is a *predicate symbol*) with respect to the first-order theory $\overline{\mathcal{R}}$ of $\mathcal{R}$, i.e., $s \to_{\mathcal{R}} t$ iff $\overline{\mathcal{R}} \vdash s \to t$ holds. Accordingly, main computational properties such as *confluence* and *termination* are

defined by applying the usual abstract notions to $\to_{\mathcal{R}}$. For instance, a GTRS $\mathcal{R}$ is *terminating* if $\to_{\mathcal{R}}$ is terminating, i.e., there is no infinite rewrite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots$. Confluence of terminating GTRSs $\mathcal{R}$ is characterized as the joinability of a number of conditional pairs $\langle s, t\rangle \Leftarrow c$ obtained from the rules in $\mathcal{R}$ [21]. In this paper we investigate how to prove *termination of GTRSs*. As explained in [24], the "termination behavior" (of CTRSs) can be investigated as having a *horizontal dimension* (H-termination, or just termination in the usual sense), which pays attention to *sequences* of rewriting steps only, and a *vertical dimension* (V-termination) which pays attention to the existence of *infinite proof trees*, eventually built to prove a rewriting goal $s \to t$ in a proof system as in natural deduction [30]. When *both* H- and V-termination are achieved, the CTRS is said to be *operationally terminating* [22]. Thus, termination is a weaker property, *easier* to achieve. For instance, $\mathcal{R}$ in Example 1 is *not* operationally terminating: the attempt to rewrite $\mathsf{div}(0,0)$ using rule (10) leads to build an infinite tree due to the (recurrent) need to prove that $\mathsf{div}(0,0) \approx \mathsf{pair}(q,r)$ (obtained from the second condition of the rule after simplifying $\mathsf{div}(0-0,0) \approx \mathsf{pair}(q,r)$) is satisfied by further rewriting on $\mathsf{div}(0,0)$. However, $\mathcal{R}$ is *terminating* (see Example 43).

The paper is organized as follows: after some preliminaries in Section 2, Section 3 recalls the notion of a GTRS. Section 4 investigates the structure of infinite rewrite sequences with GTRSs. Nowadays, proofs of termination of reduction-based systems are based on the notion of *dependency pair* (DP [2]) and *dependency pair framework* [8, 9] for Term Rewriting Systems (TRSs [3]). Section 5 introduces appropriate notions of dependency pairs of a GTRS $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$, which are viewed as a new set P of Horn clauses which are added to H to obtain a new GTRS $\mathsf{DP}_{HC}(\mathcal{R})$ which characterizes termination of $\mathcal{R}$. Section 6 introduces a framework for (dis)proving termination of GTRSs which is amenable for automation through appropriate adaptations of the usual notions of *termination problem* and *processor* introduced in the DP Framework for TRSs [8]. Section 7 introduces five processors to be used in our framework and illustrates their use by means of examples. Section 8 discusses some related work. Section 9 concludes.

2 Preliminaries

In the following, we often write *iff* instead of *if and only if*. We assume some familiarity with the basic notions of term rewriting [3, 29, 33] and first-order logic [7, 27]. We just summarize the main notions and notations we use.

Given a binary relation $\mathsf{R} \subseteq A \times A$ on a set A, we often write $a \ \mathsf{R} \ b$ instead of $(a,b) \in \mathsf{R}$. The *transitive* closure of R is denoted by R^+, and its *reflexive and transitive* closure by R^*. An element $a \in A$ is *irreducible* (or an R-*normal form*), if there exists no b such that $a \ \mathsf{R} \ b$. We say that R is *terminating* if there is no infinite sequence $a_1 \ R \ a_2 \ R \ a_3 \cdots$. In this paper, $\mathcal{X}$ denotes a countable set of *variables* and $\mathcal{F}$ denotes a *signature*, i.e., a set of *symbols* $\{f, g, \ldots\}$, each with 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 $\mathcal{T}(\mathcal{F}, \mathcal{X})$. The set of variables occurring in t is $\mathcal{V}ar(t)$. Terms are viewed as labeled trees in the usual way. *Positions* p are represented by chains of positive natural numbers used to address subterms $t|_p$ of t. The root position of a term is denoted as Λ; the root symbol as $root(t)$. The *set of positions* of a term t is $\mathcal{P}os(t)$. A term t is a *strict* subterm of s (written $s \rhd t$) iff $t = s|_p$ for some $p \in \mathcal{P}os(s) - \{\Lambda\}$. We write $s \trianglerighteq t$ if $s = t$ or $s \rhd t$.

Given a signature $\mathcal{F}$, a *replacement map* is a mapping μ from symbols in $\mathcal{F}$ to sets of positive numbers satisfying $\mu(f) \subseteq \{1, \ldots, ar(f)\}$ for all $f \in \mathcal{F}$ [17]. The set of replacement maps for $\mathcal{F}$ is $M_{\mathcal{F}}$. We use $\mu_\top(f) = \{1, \ldots, ar(f)\}$ and $\mu_\bot(f) = \emptyset$ for all $f \in \mathcal{F}$. The set of μ-

replacing (or active*) positions* of t is $\mathcal{P}os^{\mu}(t) = \{\Lambda\}$, if $t \in \mathcal{X}$, and $\mathcal{P}os^{\mu}(t) = \{\Lambda\} \cup \{i.p \mid i \in \mu(f), p \in \mathcal{P}os^{\mu}(t_i)\}$, if $t = f(t_1, \ldots, t_k)$. The set of *non-μ-replacing* (or *frozen*) positions of t is $\overline{\mathcal{P}os^{\mu}}(t) = \mathcal{P}os(t) - \mathcal{P}os^{\mu}(t)$. Accordingly, subterms $u = t|_p$ of a term t at an active (resp. frozen) position $p \in \mathcal{P}os^{\mu}(t)$ (resp. $p \in \overline{\mathcal{P}os^{\mu}}(t)$) of t are called *active* (resp. *frozen*). Positions of *active* non-variable subterms of t are denoted as $\mathcal{P}os^{\mu}_{\mathcal{F}}(t)$. Given a term t, the set of variables occurring at active positions in t is $\mathcal{V}ar^{\mu}(t) = \{x \in \mathcal{V}ar(t) \mid \exists p \in \mathcal{P}os^{\mu}(t), x = t|_p\}$. A term t is a *strict, active* subterm of s (written $s \rhd_{\mu} t$) iff $t = s|_p$ for some $p \in \mathcal{P}os^{\mu}(s) - \{\Lambda\}$. We write $s \trianglerighteq_{\mu} t$ if $s = t$ or $s \rhd_{\mu} t$.

Consider a signature $\mathcal{F}$ of *function symbols* and a signature Π of *predicate symbols*. Atoms $A \in Atoms(\mathcal{F}, \Pi, \mathcal{X})$ and first-order formulas $F \in Forms(\mathcal{F}, \Pi, \mathcal{X})$ on such signatures, with variables in $\mathcal{X}$, are built in the usual way. A (definite) Horn clause (with label α) is written $\alpha : A \Leftarrow A_1, \ldots, A_n$, for atoms $A, A_1, \ldots, A_n$; if $n = 0$, then α is written A rather than $A \Leftarrow$. A first-order theory (FO-theory for short) Th is a set of sentences (formulas whose variables are all *quantified*). An $\mathcal{F}, \Pi$-*structure* $\mathcal{A}$ (or just *structure* if no confusion arises) consists of a *non-empty* set $\mathsf{dom}(\mathcal{A})$, called *domain* and often denoted $\mathcal{A}$ if no confusion arises, together with an interpretation of symbols $f \in \mathcal{F}$ and $P \in \Pi$ as mappings $f^{\mathcal{A}}$ and relations $P^{\mathcal{A}}$ on $\mathcal{A}$, respectively. Then, the usual interpretation of first-order formulas with respect to $\mathcal{A}$ is considered [27, page 60]. An $\mathcal{F}, \Pi$-model for a theory Th is just a structure $\mathcal{A}$ that makes them all true, written $\mathcal{A} \models \mathsf{Th}$. A formula F is a *logical consequence* of a theory Th (written $\mathsf{Th} \models F$) iff every model $\mathcal{A}$ of Th is also a model of F. Also, $\mathsf{Th} \vdash F$ means that F is *deducible* from Th by using a correct and complete deduction procedure. In that case, $\vdash$ and $\models$ coincide.

An *f-condition* γ is an atom [13]. Sequences $\mathsf{F} = (\gamma_i)_{i=1}^{n} = (\gamma_1, \ldots, \gamma_n)$ of f-conditions are called *f-sequences*. We often drop "f-" when no confusion arises. Empty sequences are written $()$. Given an FO-theory Th, a condition γ is Th-*feasible* (or just *feasible* if no confusion arises) if $\mathsf{Th} \vdash \sigma(\gamma)$ holds for some substitution σ; otherwise, it is *infeasible*. Note that (in)feasibility is, in general, undecidable. A sequence F is Th-feasible (or just *feasible*) iff there is a substitution σ such that, for all $\gamma \in \mathsf{F}$, $\mathsf{Th} \vdash \sigma(\gamma)$ holds.

A CTRS is a pair $\mathcal{R} = (\mathcal{F}, R)$ where $\mathcal{F}$ is a signature and R is a set of rules $\ell \to r \Leftarrow c$, with c a sequence $s_1 \approx t_1, \cdots, s_n \approx t_n$ 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. *Labeled* rules are written $\alpha : \ell \to r \Leftarrow c$, where α is a *label*. In the following, given $\mathcal{R}$, we often write $\alpha \in \mathcal{R}$, instead of $\alpha \in R$, to say that α is a rule of $\mathcal{R}$.

3 Generalized Term Rewriting Systems

A *Generalized Term Rewriting System (GTRS)* is a tuple $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ where $\mathcal{F}$ is a signature of *function symbols*, Π is a signature of *predicate symbols*, including at least $\to$ and $\to^*$, $\mu \in M_{\mathcal{F}}$ ($\mu_{\top}$ is assumed if μ is not explicitly given, as in Example 1), H is a (possibly empty) set of clauses $A \Leftarrow c$, where $root(A) \notin \{\to, \to^*\}$, and R is a set of rewrite rules $\ell \to r \Leftarrow c$ such that $\ell \notin \mathcal{X}$. In both cases, c is a sequence of atoms. Note that rules in R are Horn clauses. We often give them a label α as follows: $\alpha : \ell \to r \Leftarrow c$. The rules in R permit the usual distinction of function symbols $f \in \mathcal{F}$ as *defined* (if $root(\ell) = f$ for some $\ell \to r \Leftarrow c \in R$) or *constructor* symbols (otherwise). The set of defined (resp. constructor) symbols of $\mathcal{R}$ is $\mathcal{D}_{\mathcal{R}}$ or just $\mathcal{D}$ if no confusion arises (resp. $\mathcal{C}_{\mathcal{R}}$ or $\mathcal{C}$). The FO-theory of $\mathcal{R}$ is

$$\overline{\mathcal{R}} = \{(\mathrm{Rf}), (\mathrm{Co})\} \cup \{(\mathrm{Pr})_{f,i} \mid f \in \mathcal{F}, i \in \mu(f)\} \cup \{(\mathrm{HC})_{\alpha} \mid \alpha \in H \cup R\}$$

Table 1 Generic sentences of the first-order theory of rewriting.

Label	Sentence
(Rf)	$(\forall x)\ x \to^* x$
(Co)	$(\forall x, y, z)\ x \to y \wedge y \to^* z \Rightarrow x \to^* z$
$(\mathrm{Pr})_{f,i}$	$(\forall x_1, \ldots, x_k, y_i)\ x_i \to y_i \Rightarrow f(x_1, \ldots, x_i, \ldots, x_k) \to f(x_1, \ldots, y_i, \ldots, x_k)$
$(\mathrm{HC})_{A \Leftarrow A_1, \ldots, A_n}$	$(\forall x_1, \ldots, x_p)\ A_1 \wedge \cdots \wedge A_n \Rightarrow A$ where $x_1, \ldots, x_p$ are the variables occurring in $A_1, \ldots, A_n$ and A

where (see Table 1), (Rf) expresses *reflexivity* of many-step rewriting; (Co) expresses *compatibility* of one-step and many-step rewriting; for each k-ary function symbol f and $i \in \mu(f)$, $(\mathrm{Pr})_{f,i}$ enables the *propagation* of rewriting steps in the i-th immediate *active* subterm $t|_i$ of a term t with root symbol f; finally, for each Horn clause $\alpha \in H \cup R$, $(\mathrm{HC})_\alpha$ provides the usual *implicative* form for them. For all terms s and t, we write $s \to_{\mathcal{R}} t$ (resp. $s \to^*_{\mathcal{R}} t$) iff $\overline{\mathcal{R}} \vdash s \to t$ (resp. $\overline{\mathcal{R}} \vdash s \to^* t$). Since $\overline{\mathcal{R}}$ is a Horn theory, the use of, e.g., *resolution* [31] provides a correct and complete proof method for the considered goals.

▶ Remark 2 (Infeasible rules). Only rules $\ell \to r \Leftarrow c \in R$ whose conditional part c is $\overline{\mathcal{R}}$-feasible can be used in rewriting steps. In the following, we assume that all rules in $\mathcal{R}$ can be proved $\overline{\mathcal{R}}$-feasible. Again, resolution can be used for this purpose, see also [13].

The following result provides a *position-based* view of rewriting with GTRSs.

▶ **Proposition 3** ([21, Proposition 58]). *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ be a GTRS and $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. Then, $s \to_{\mathcal{R}} t$ iff there is $p \in \mathcal{P}os^\mu(s)$ and $\ell \to r \Leftarrow c \in R$ such that (i) $s|_p = \sigma(\ell)$ for some substitution σ, (ii) for all $A \in c$, $\overline{\mathcal{R}} \vdash \sigma(A)$ holds, and (iii) $t = s[\sigma(r)]_p$.*

▶ **Definition 4.** *A GTRS $\mathcal{R}$ is terminating iff there is no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots$.*

▶ **Example 5.** Consider the GTRS $\mathcal{R} = (\mathcal{F}, \Pi, \mu_\top, H, R)$ (from COPS #342, [34, Ex. 4])

$$x \approx y \Leftarrow x \to^* y \tag{11}$$
$$\mathsf{f}(x', x'') \to \mathsf{h}(x, \mathsf{f}(x, \mathsf{b})) \Leftarrow x' \approx x, x'' \approx x \tag{12}$$
$$\mathsf{f}(\mathsf{g}(y'), y'') \to \mathsf{h}(y, \mathsf{f}(\mathsf{g}(y), \mathsf{a})) \Leftarrow y' \approx y, y'' \approx y \tag{13}$$
$$\mathsf{a} \to \mathsf{b} \tag{14}$$

where $\mathcal{F} = \{\mathsf{a}, \mathsf{b}, \mathsf{f}, \mathsf{g}, \mathsf{h}\}$, $\Pi = \{\approx, \to, \to^*\}$, $H = \{(11)\}$, and $R = \{(12), (13), (14)\}$. $\overline{\mathcal{R}}$ is:

$$\{(\mathrm{Rf}), (\mathrm{Co}), (\mathrm{Pr})_{\mathsf{f},1}, (\mathrm{Pr})_{\mathsf{f},2}, (\mathrm{Pr})_{\mathsf{g},1}, (\mathrm{Pr})_{\mathsf{h},1}, (\mathrm{Pr})_{\mathsf{h},2}, (\mathrm{HC})_{(11)}, (\mathrm{HC})_{(12)}, (\mathrm{HC})_{(13)}, (\mathrm{HC})_{(14)}\}$$

Note that $\mathcal{R}$ is *not* terminating: $\underline{\mathsf{f}(\mathsf{b}, \mathsf{b})} \to_{\mathcal{R}} \mathsf{h}(\mathsf{b}, \underline{\mathsf{f}(\mathsf{b}, \mathsf{b})}) \to_{\mathcal{R}} \cdots$ We prove it in Example 30.

The following example illustrates the use of replacement maps.

▶ **Example 6.** Consider the GTRS $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ with $\mathcal{F}$, Π, H and R as in Example 5 but with $\mu(\mathsf{f}) = \{1, 2\}$, $\mu(\mathsf{g}) = \{1\}$, and $\mu(\mathsf{h}) = \emptyset$. Now,

$$\overline{\mathcal{R}} = \{(\mathrm{Rf}), (\mathrm{Co}), (\mathrm{Pr})_{\mathsf{f},1}, (\mathrm{Pr})_{\mathsf{f},2}, (\mathrm{Pr})_{\mathsf{g},1}, (\mathrm{HC})_{(11)}, (\mathrm{HC})_{(12)}, (\mathrm{HC})_{(13)}, (\mathrm{HC})_{(14)}\}$$

Note the absence of $(\mathrm{Pr})_{\mathsf{h},1}$ and $(\mathrm{Pr})_{\mathsf{h},2}$ due to $\mu(\mathsf{h}) = \emptyset$. Since rewritings on the arguments of h are *forbidden* by $\mu(\mathsf{h}) = \emptyset$, $\mathcal{R}$ is *terminating* (see Example 8 below).

Termination of GTRSs is characterized by interpretation of the symbols as follows.

▶ **Proposition 7** (Termination of GTRSs by interpretation). *A GTRS $\mathcal{R}$ is terminating iff there is a model $\mathcal{A}$ of $\overline{\mathcal{R}}$ with non-empty domain $\mathcal{A}$ and $\to^{\mathcal{A}}$ is* terminating *on $\mathcal{A}$.*

▶ **Example 8.** Consider the GTRS $\mathcal{R}$ in Example 6. By Proposition 7, the following model $\mathcal{A}$ of $\overline{\mathcal{R}}$ (computed by the tool AGES [12] for the automatic generation of models of first-order theories) with domain $\mathcal{A} = \mathbb{N} \cup \{-1\}$, function symbols interpreted by

$$\mathsf{a}^{\mathcal{A}}{=}3 \qquad \mathsf{b}^{\mathcal{A}}{=}0 \qquad \mathsf{f}^{\mathcal{A}}(x){=}x+y+13 \qquad \mathsf{g}^{\mathcal{A}}(x){=}x \qquad \mathsf{h}^{\mathcal{A}}(x,y){=}1$$

and predicate symbols as follows:

$$x \approx^{\mathcal{A}} y \Leftrightarrow \mathit{true} \qquad x \to^{\mathcal{A}} y \Leftrightarrow x > y \qquad x(\to^*)^{\mathcal{A}} y \Leftrightarrow \mathit{true}$$

proves termination of $\mathcal{R}$, as $\to^{\mathcal{A}}$ is terminating on $\mathbb{N} \cup \{-1\}$.

Sometimes, termination of GTRSs $\mathcal{R}$ is difficult to prove by using Proposition 7 due to the need of (automatically) synthesizing appropriate interpretations. In the following, we develop a more powerful approach to prove termination of GTRSs. For instance, in Example 20 we prove termination of $\mathcal{R}$ in Example 6 without synthesizing any interpretation.

4 Infinite rewrite sequences starting from minimal terms

In this section, following [14], we consider the structure of infinite rewrite sequences as the starting point for the subsequent analysis of *termination* of GTRSs.

▶ **Definition 9.** *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ be a GTRS. A term t is* nonterminating *if there is an infinite sequence $t = t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots$. A nonterminating term t is* minimal *if all strict* active *subterms t' of t (i.e., $t \rhd_\mu t'$) are terminating. Let $\mathcal{M}_{\infty,\mu}$ be the set of such minimal nonterminating terms.*

Nonterminating terms contain active minimal nonterminating subterms (cf. [1, Lemma 3]).

▶ **Definition 10.** *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ be a GTRS. The set of* active defined subterms *of a term t is $\mathcal{D}_{\trianglerighteq_\mu}(\mathcal{R}, t) = \{t|_p \mid p \in \mathcal{P}os^\mu(t), root(t|_p) \in \mathcal{D}\}$.*

Infinite rewrite sequences starting from a minimal nonterminating term s (i) first rewrite s *below the root* to a term t (written $s \xrightarrow{>\Lambda}{}^*_{\mathcal{R}} t$) which (ii) matches the left-hand side ℓ of a rule $\ell \to r \Leftarrow c \in \mathcal{R}$, i.e., $t = \sigma(\ell)$ for some substitution σ such that $\sigma(c)$ holds, and hence (iii) t rewrites at the root to $\sigma(r)$ (written $t = \sigma(\ell) \xrightarrow{\Lambda}_{\mathcal{R}} \sigma(r)$), and then (iv) $\sigma(r)$ contains a minimal nonterminating subterm u on which the infinite sequence may continue.

▶ **Proposition 11.** *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ be a GTRS and $t \in \mathcal{M}_{\infty,\mu}$. There exist $\ell \to r \Leftarrow c \in R$, a substitution σ, and $u \in \mathcal{M}_{\infty,\mu}$ such that $t \xrightarrow{>\Lambda}{}^*_{\mathcal{R}} \sigma(\ell) \xrightarrow{\Lambda}_{\mathcal{R}} \sigma(r) \trianglerighteq_\mu u$, and either*

1. *there is $s \in \mathcal{D}_{\trianglerighteq_\mu}(\mathcal{R}, r)$, $\ell \not\trianglerighteq_\mu s$, such that $u = \sigma(s)$, or*
2. *there is $x \in \mathcal{V}ar^\mu(r) - \mathcal{V}ar^\mu(\ell)$ such that $\sigma(x) \trianglerighteq_\mu u$.*

The following definition collects the rules that are used in the first and second cases of Proposition 11, respectively.

▶ **Definition 12.** *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ be a GTRS. Let $DRules(\mathcal{R})$ be the set of rules in $\mathcal{R}$ which* depend *on other defined symbols in $\mathcal{R}$ and $ERules(\mathcal{R})$ be the set of rules in $\mathcal{R}$ whose right-hand sides contain* active *variables which are* not *active in the corresponding left-hand side:*

$$\begin{aligned} DRules(\mathcal{R}) &= \{\ell \to r \Leftarrow c \in \mathcal{R} \mid \mathcal{D}_{\trianglerighteq_\mu}(\mathcal{R}, r) \neq \emptyset\} \\ ERules(\mathcal{R}) &= \{\ell \to r \Leftarrow c \in \mathcal{R} \mid \mathcal{V}ar^\mu(r) - \mathcal{V}ar^\mu(\ell) \neq \emptyset\} \end{aligned}$$

5 Dependency pairs for termination of GTRSs

In the *dependency pair approach* (or just *DP approach*) for TRSs [2], termination of a TRS $\mathcal{R}$ is characterized as the absence of infinite *chains* $\langle u_1, v_1\rangle, \langle u_2, v_2\rangle, \ldots$,[1] where for all $i \geq 1$, $\langle u_i, v_i\rangle$ are *dependency pairs*, which are obtained from the rules $\ell \to r \in \mathcal{R}$ [2, Definition 3] by *marking* the root symbols f (as, e.g., $f^\sharp$ but often *capitalizing* it as F) in the left-hand side ℓ of the rule and also the root symbol of the *defined* subterm s of the right-hand side r which is referred in Proposition 11. Hence, $\langle u, v\rangle$, where $u = \ell^\sharp$ and $v = s^\sharp$ is a *dependency pair* of $\mathcal{R}$. Furthermore, there is a substitution σ such that, for all $i \geq 1$, consecutive pairs are *connected* as follows $\sigma(v_i) \to_{\mathcal{R}}^* \sigma(u_{i+1})$ [2, Definition 5]. This corresponds to the sequence $t \xrightarrow{>\Lambda}_{\mathcal{R}}^* \sigma(\ell)$ referred in Proposition 11. In our generalization of the DP approach to GTRSs $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$, we use a fresh predicate symbol $\overset{\mathsf{t}}{\to} \notin \Pi$ to represent *conditional pairs* $u \overset{\mathsf{t}}{\to} v \Leftarrow c$ which are then added as new *definite Horn clauses* to H to capture the *termination* behavior of $\mathcal{R}$ (hence the "t" over the arrow). In this way, we obtain a new GTRS $\mathcal{R}'$ whose set of rules is R as well. Since the reduction relation of $\mathcal{R}$ does *not* depend on $\overset{\mathsf{t}}{\to}$, both $\to_{\mathcal{R}}$ and $\to_{\mathcal{R}'}$ *coincide*; thus, we take the pairs from the set of Horn clauses in $\mathcal{R}'$ while using $\to_{\mathcal{R}'}^* = \to_{\mathcal{R}}^*$ to *connect* them.

▶ **Notation 13.** *Given a GTRS $\mathcal{R}$, we write $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H \uplus P, R)$ making explicit the (possibly empty) set P of all clauses $u \overset{\mathsf{t}}{\to} v \Leftarrow c$ for some terms u and v and conditional part c which are used in $\mathcal{R}$. We also write $P_{\mathcal{R}}$ to refer to such a subset P of Horn clauses of $\mathcal{R}$.*

As usual in the DP approach, unless established otherwise, in the following we assume that considered pairs are pairwise variable-disjoint renamings of pairs taken from $P_{\mathcal{R}}$.

▶ **Definition 14** (Chain of pairs of a GTRS). *Let $\mathcal{R}$ be a GTRS. An $\mathcal{R}$-chain is a finite or infinite sequence of pairs $u_i \overset{\mathsf{t}}{\to} v_i \Leftarrow c_i \in P_{\mathcal{R}}$ together with a substitution σ such that, for all $i \geq 1$ ($1 \leq i < n$ for sequences of $n > 1$ pairs), $\overline{\mathcal{R}} \vdash \sigma(c_i), \sigma(v_i) \to^* \sigma(u_{i+1}), \sigma(c_{i+1})$ holds. An $\mathcal{R}$-chain is called* minimal *if for all $i \geq 1$, $\sigma(v_i)$ is terminating.*

According to Definition 14, the existence of a *finite* chain $\alpha_1, \ldots, \alpha_n$ for pairs $\alpha_i : u_i \overset{\mathsf{t}}{\to} v_i \Leftarrow c_i$, $1 \leq i \leq n$, is equivalent to the *feasibility* of the sequence

$$c_1, v_1 \to^* u_2, \ldots, c_{n-1}, v_{n-1} \to^* u_n, c_n \tag{15}$$

▶ Remark 15 (Infeasible pairs). Since only $\overline{\mathcal{R}}$-feasible pairs can be used in $\mathcal{R}$-chains, we can safely *remove* $\overline{\mathcal{R}}$-infeasible pairs from $P_{\mathcal{R}}$ in termination proofs.

As discussed above, we use *marked* symbols $f^\sharp$ associated to (*defined*) symbols f. In general, given a signature $\mathcal{F}$, we let $\mathcal{F}^\sharp = \{f^\sharp \mid f \in \mathcal{F}\}$; and given $\mu \in M_{\mathcal{F}}$ and $\mathcal{D} \subseteq \mathcal{F}$, we let $\mu^\sharp_{\mathcal{D}} \in M_{\mathcal{F} \cup \mathcal{D}^\sharp}$ (or just $\mu^\sharp$ if no confusion arises) be as follows: for all $f \in \mathcal{F} \cup \mathcal{D}^\sharp$,

$$\mu^\sharp(f) = \begin{cases} \mu(f) & \text{if } f \in \mathcal{F} \\ \mu(g) & \text{if } f = g^\sharp \text{ for some } g \in \mathcal{D} \end{cases}$$

As for the DP approach for TRSs, "classical" dependency pairs $u \overset{\mathsf{t}}{\to} v \Leftarrow c$, obtained from rules $\ell \to r \Leftarrow c$ by letting $u = \ell^\sharp$ and $v = s^\sharp$ for some active defined subterm s of v, capture the situation described in the first item of Proposition 11 (see, e.g., [14, Lemma 1] for TRSs).

[1] We use this early notation instead of the current rule-based one, $u \to v$, to prepare the inclusion of our dependency pairs in the component H, rather than R, of the considered GTRS. See Definition 14.

However, dealing with GTRSs, we also need to consider item 2 in Proposition 11 for which "classical" dependency pairs are not appropriate. In this case, Proposition 11 guarantees that a minimal nonterminating subterm s can be found in $\sigma(x)$. In order to *find* and then *mark* s, we use (i) the *active subterm* relation $\trianglerighteq_\mu$ defined by a set of clauses $\mathcal{S}ubt(\mathcal{F},\mu)$ and (ii) a binary predicate symbol Mk defined by a set of clauses $\mathcal{M}ark(\mathcal{F})$ for a signature $\mathcal{F}$ and $\mu \in M_{\mathcal{F}}$:

1. Let $\varpi_{\trianglerighteq_\mu}$ be a new predicate symbol. Then, $\mathcal{S}ubt(\mathcal{F},\mu)$ consists of clauses
$$x\ \varpi_{\trianglerighteq_\mu}\ x \tag{16}$$
$$f(x_1,\ldots,x_i,\ldots,x_k)\ \varpi_{\trianglerighteq_\mu}\ x_i' \ \Leftarrow\ x_i\ \varpi_{\trianglerighteq_\mu}\ x_i' \tag{17}$$
for each $f \in \mathcal{F}$, $k = ar(f)$, $i \in \mu(f)$, and variables x and x_i, x_i' for $1 \le i \le k$.
2. Let Mk be a new predicate symbol. Then, $\mathcal{M}ark(\mathcal{F})$ consists of the clauses :
$$\mathsf{Mk}(f(x_1,\ldots,x_k), f^\sharp(x_1,\ldots,x_k)) \tag{18}$$
for each $f \in \mathcal{F}$, $k = ar(f)$, and fresh variables $x_1,\ldots,x_k$.

▶ **Definition 16.** *Let $\mathcal{R} = (\mathcal{F},\Pi,\mu,H,R)$ be a GTRS whose set of defined symbols is $\mathcal{D}$. The GTRS $\mathsf{DP}_{HC}(\mathcal{R})$ is:*

$$(\mathcal{F}\cup\mathcal{D}^\sharp, \Pi\cup\{\varpi_{\trianglerighteq_\mu}, \mathsf{Mk}\}, \mu^\sharp, H\cup\mathcal{S}ubt(\mathcal{F}\cup\mathcal{D}^\sharp,\mu^\sharp)\cup\mathcal{M}ark(\mathcal{D})\cup H_{DP}(\mathcal{R})\cup H_{DPC}(\mathcal{R}), R)$$

where

$$\begin{array}{rcl} H_{DP}(\mathcal{R}) & = & \{\ell^\sharp \xrightarrow{t} v^\sharp \Leftarrow c \mid \ell \to r \Leftarrow c \in DRules(R), v \in \mathcal{D}_{\trianglerighteq_\mu}(\mathcal{R},r), \ell \not\trianglerighteq_\mu v\} \\ H_{DPC}(\mathcal{R}) & = & \{\ell^\sharp \xrightarrow{t} x'' \Leftarrow c, x\ \varpi_{\trianglerighteq_\mu}\ x', \mathsf{Mk}(x',x'') \mid \ell \to r \Leftarrow c \in R, x \in \mathcal{V}ar^\mu(r) - \mathcal{V}ar^\mu(\ell) \\ & & \quad \textit{and } x' \textit{ and } x'' \textit{ are fresh variables}\} \end{array}$$

The set of clauses $H_{DP}(\mathcal{R})$ correspond to "classical" conditional dependency pairs for $\mathcal{R}$. $H_{DPC}(\mathcal{R})$ contains *collapsing* dependency pairs for rules $\ell \to r \Leftarrow c$ in $\mathcal{R}$, as the right-hand side of one of such pairs is a *variable* x'' corresponding to a variable $x \in \mathcal{V}ar(r)$ which is active in r but missing, or not active, in ℓ; the conditional part of such pairs uses $\varpi_{\trianglerighteq_\mu}$ to extract active minimal nonterminating subterms s to which x' gets bounded, and finally mark it using Mk so that x'' is finally bound to $s^\sharp$ (see item 2 of Proposition 11). Note that $P_{\mathsf{DP}_{HC}(\mathcal{R})} = H_{DP}(\mathcal{R}) \cup H_{DPC}(\mathcal{R})$.

▶ **Example 17.** For $\mathcal{R}$ in Example 1, $P_{\mathsf{DP}_{HC}(\mathcal{R})} = H_{DP}(\mathcal{R}) \cup H_{DPC}(\mathcal{R})$ where $H_{DP}(\mathcal{R}) = \{(19)\}$ and $H_{DPC}(\mathcal{R}) = \{(20),(21)\}$, with

$$\mathsf{s}(x) -^\sharp \mathsf{s}(y) \xrightarrow{t} x -^\sharp y \tag{19}$$
$$\mathsf{DIV}(x,y) \xrightarrow{t} q'' \Leftarrow y \le x, \mathsf{div}(x-y,y) \approx \mathsf{pair}(q,r), q\ \varpi_{\trianglerighteq_\mu}\ q', \mathsf{Mk}(q',q'') \tag{20}$$
$$\mathsf{DIV}(x,y) \xrightarrow{t} r'' \Leftarrow y \le x, \mathsf{div}(x-y,y) \approx \mathsf{pair}(q,r), r\ \varpi_{\trianglerighteq_\mu}\ r', \mathsf{Mk}(r',r'') \tag{21}$$

where, as usual in the DP approach, the uppercase identifier DIV corresponds to $\mathsf{div}^\sharp$. Besides, $\mathcal{S}ubt(\mathcal{F},\mu) = \{(22),\ldots,(33)\}$ and $\mathcal{M}ark(\mathcal{D}) = \{(34),(35)\}$, with

$$x\ \varpi_{\trianglerighteq_\mu}\ x \tag{22}$$
$$s(x)\ \varpi_{\trianglerighteq_\mu}\ x' \ \Leftarrow\ x\ \varpi_{\trianglerighteq_\mu}\ x' \tag{23}$$
$$x - y\ \varpi_{\trianglerighteq_\mu}\ x' \ \Leftarrow\ x\ \varpi_{\trianglerighteq_\mu}\ x' \tag{24}$$
$$x - y\ \varpi_{\trianglerighteq_\mu}\ y' \ \Leftarrow\ y\ \varpi_{\trianglerighteq_\mu}\ y' \tag{25}$$
$$\mathsf{div}(x,y)\ \varpi_{\trianglerighteq_\mu}\ x' \ \Leftarrow\ x\ \varpi_{\trianglerighteq_\mu}\ x' \tag{26}$$
$$\mathsf{div}(x,y)\ \varpi_{\trianglerighteq_\mu}\ y' \ \Leftarrow\ y\ \varpi_{\trianglerighteq_\mu}\ y' \tag{27}$$
$$\mathsf{pair}(x,y)\ \varpi_{\trianglerighteq_\mu}\ x' \ \Leftarrow\ x\ \varpi_{\trianglerighteq_\mu}\ x' \tag{28}$$
$$\mathsf{pair}(x,y)\ \varpi_{\trianglerighteq_\mu}\ y' \ \Leftarrow\ y\ \varpi_{\trianglerighteq_\mu}\ y' \tag{29}$$
$$x -^\sharp y\ \varpi_{\trianglerighteq_\mu}\ x' \ \Leftarrow\ x\ \varpi_{\trianglerighteq_\mu}\ x' \tag{30}$$
$$x -^\sharp y\ \varpi_{\trianglerighteq_\mu}\ y' \ \Leftarrow\ y\ \varpi_{\trianglerighteq_\mu}\ y' \tag{31}$$
$$\mathsf{DIV}(x,y)\ \varpi_{\trianglerighteq_\mu}\ x' \ \Leftarrow\ x\ \varpi_{\trianglerighteq_\mu}\ x' \tag{32}$$
$$\mathsf{DIV}(x,y)\ \varpi_{\trianglerighteq_\mu}\ y' \ \Leftarrow\ y\ \varpi_{\trianglerighteq_\mu}\ y' \tag{33}$$
$$\mathsf{Mk}(x-y, x -^\sharp y) \tag{34}$$
$$\mathsf{Mk}(\mathsf{div}(x,y), \mathsf{DIV}(x,y)) \tag{35}$$

▶ **Example 18.** For $\mathcal{R}$ in Example 5, $P_{\mathsf{DP}_{HC}(\mathcal{R})} = H_{DP}(\mathcal{R}) \cup H_{DPC}(\mathcal{R})$, where $H_{DP}(\mathcal{R}) = \{(36), (37), (38)\}$ and $H_{DPC}(\mathcal{R}) = \{(39), (40)\}$, with

$$\mathsf{F}(x', x'') \xrightarrow{\mathsf{t}} \mathsf{F}(x, \mathsf{b}) \quad \Leftarrow \quad x' \approx x, x'' \approx x \tag{36}$$

$$\mathsf{F}(\mathsf{g}(y'), y'') \xrightarrow{\mathsf{t}} \mathsf{F}(\mathsf{g}(y), \mathsf{a}) \quad \Leftarrow \quad y' \approx y, y'' \approx y \tag{37}$$

$$\mathsf{F}(\mathsf{g}(y'), y'') \xrightarrow{\mathsf{t}} \mathsf{A} \quad \Leftarrow \quad y' \approx y, y'' \approx y \tag{38}$$

$$\mathsf{F}(x', x'') \xrightarrow{\mathsf{t}} z' \quad \Leftarrow \quad x' \approx x, x'' \approx x, x \; \varpi_{\trianglerighteq_\mu} \; z, \mathsf{Mk}(z, z') \tag{39}$$

$$\mathsf{F}(\mathsf{g}(y'), y'') \xrightarrow{\mathsf{t}} z' \quad \Leftarrow \quad y' \approx y, y'' \approx y, y \; \varpi_{\trianglerighteq_\mu} \; z, \mathsf{Mk}(z, z') \tag{40}$$

▶ **Theorem 19** (Termination). *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H, R)$ be a GTRS.*
1. *$\mathcal{R}$ is* terminating *if there is no infinite minimal $\mathsf{DP}_{HC}(\mathcal{R})$-chain.*
2. *$\mathcal{R}$ is* nonterminating *if there is an infinite $\mathsf{DP}_{HC}(\mathcal{R})$-chain.*

▶ **Example 20** (Termination of $\mathcal{R}$ in Example 6 by absence of $\mathsf{DP}_{HC}(\mathcal{R})$-chains). For $\mathcal{R}$ in Example 6, $H_{DP}(\mathcal{R})$ is empty as the (definite) subterms $\mathsf{f}(x, \mathsf{b})$, $\mathsf{f}(\mathsf{g}(y), \mathsf{a})$, and a in the right-hand sides of rules (13) and (14) are *frozen*. Also, $H_{DPC}(\mathcal{R}) = \emptyset$ as variables x and y in the right-hand sides of rules (13) and (14) are *frozen* too. Thus, no $\mathsf{DP}_{HC}(\mathcal{R})$-chain is possible. By Theorem 19, $\mathcal{R}$ is terminating. In contrast to the proof of termination in Example 8, based on Proposition 7, no synthesis of any model is required here.

The following section introduces a *Dependency Pair Framework* for the *mechanization* of proofs of termination of GTRSs using dependency pairs.

6 The Dependency Pair Framework for GTRSs

The Dependency Pair (DP) Framework is a *divide-and-conquer* technique to prove "termination problems". They are decomposed into *smaller* or *simpler* ones to finally obtain "trivial" problems which can be easily solved. Then, we combine the obtained answers to provide a solution to the initial problem. Thus, the first ingredient of the DP Framework for GTRSs is a suitable notion of *termination problem*. Let $\mathsf{F} = \{\mathsf{a}, \mathsf{m}\}$ be a signature of *flag constants* φ referring to arbitrary (resp. minimal) $\mathcal{R}$-chains if $\varphi = \mathsf{a}$ (resp. $\varphi = \mathsf{m}$).

▶ **Definition 21** (GTRS problem). *A GTRS problem $\tau = (\mathcal{R}, \varphi)$ consists of a GTRS $\mathcal{R}$ and $\varphi \in \mathsf{F}$.*

We often speak of $(\mathcal{R}, \mathsf{m})$-chains (or just τ-chains if $\tau = (\mathcal{R}, \mathsf{m})$) instead of *minimal* $\mathcal{R}$-chains; and of $(\mathcal{R}, \mathsf{a})$-chains (resp. τ-chains if $\tau = (\mathcal{R}, \mathsf{a})$) instead of *arbitrary* $\mathcal{R}$-chains.

▶ **Definition 22** (Finite GTRS problem). *A GTRS problem $\tau = (\mathcal{R}, \varphi)$ is* finite *iff there is no infinite τ-chain; otherwise, τ is* infinite.

Accordingly, the following result rephrases Theorem 19.

▶ **Theorem 23.** *A GTRS $\mathcal{R}$ is terminating iff $(\mathsf{DP}_{HC}(\mathcal{R}), \varphi)$ is finite for some $f \in \mathsf{F}$.*

Processors *transform* GTRS problems $\tau = (\mathcal{R}, \varphi)$ into possibly empty sets $\{\tau_1, \ldots, \tau_n\}$ of GTRS problems $\tau_i = (\mathcal{R}_i, \varphi_i)$, hopefully easier to deal with. Processors may also return "no", with the intended meaning of τ being *infinite*.

▶ **Definition 24** (GTRS processor). *A* GTRS processor P *is a partial function from GTRS problems into sets of GTRS problems. Alternatively, it can return* "no". $\mathsf{Dom}(\mathsf{P})$ *is the set of GTRS problems τ for which P is defined.*

In the following we often speak of "processors" rather than "GTRS processors". For the sake of readability, we often write $\mathsf{P}(\mathcal{R},\varphi)$ rather than $\mathsf{P}((\mathcal{R},\varphi))$. The most relevant properties to be established when using processors are *soundness* and *completeness*. Soundness is essential to prove GTRS problems *finite* by using combinations of processors; completeness for proving *infiniteness*.

▶ **Definition 25** (Soundness and completeness). *Let* P *be a processor and* $\tau \in \mathsf{Dom}(\mathsf{P})$*. We say that* P *is:*

- τ-sound *iff* τ *is finite whenever* $\mathsf{P}(\tau) \neq \mathsf{no}$ *and for all* $\tau' \in \mathsf{P}(\tau)$*,* τ' *is finite.*
- τ-complete *iff* τ *is infinite whenever* $\mathsf{P}(\tau) = \mathsf{no}$ *or there is* $\tau' \in \mathsf{P}(\tau)$ *such that* τ' *is infinite.*

Accordingly, given $\varphi \in \mathsf{F}$*, we say that* P *is* φ-sound *(resp.* φ-complete*) if it is* τ*-sound (*τ*-complete) for all* $\tau = (\mathcal{R},\varphi') \in \mathsf{Dom}(\mathsf{P})$ *such that* $\varphi = \varphi'$*.* P *is* sound *(resp.* complete*) if it is* φ*-sound (*φ*-complete) for all* $\varphi \in \mathsf{F}$*.*

Processors are used in a *divide and conquer* scheme to incrementally simplify a *target* GTRS problem τ_0, possibly decomposing it into (a tree of) smaller problems which are independently treated in the same way.

▶ **Definition 26** (GTRS Proof Tree). *Let* τ_0 *be a GTRS problem. A* GTRS Proof tree $\mathcal{T}$ *(GTRSP-tree for short) for* τ_0 *is a tree whose nodes are labeled with GTRS problems; the leaves may also be labeled with either* "yes" *or* "no"*. The root of* $\mathcal{T}$ *is labeled with* τ_0*. For every inner node* n *with label* τ*, there is a processor* P *such that* $\tau \in \mathsf{Dom}(\mathsf{P})$ *and:*

1. *If* $\mathsf{P}(\tau) = \mathsf{no}$*, then* n *has just one child* n' *with label* "no"*.*
2. *If* $\mathsf{P}(\tau) = \emptyset$*, then* n *has just one child* n' *with label* "yes"*.*
3. *If* $\mathsf{P}(\tau) = \{\tau_1,\ldots,\tau_k\}$ *with* $k > 0$*, then* n *has exactly* k *children* $\mathsf{n}_1,\ldots,\mathsf{n}_k$ *with labels* $\tau_1,\ldots,\tau_k$*, respectively.*

▶ **Theorem 27.** *Let* τ *be a GTRS problem and* $\mathcal{T}$ *be a GTRSP-tree for* τ*. Then,*

1. *If all leaves in* $\mathcal{T}$ *are labeled with "*yes*", and all involved processors are* sound *for the GTRS problems they are applied to, then* τ *is* finite*.*
2. *If* $\mathcal{T}$ *has a leaf with label "*no*" and all processors from* τ *to the leaf are* complete *for the GTRS problems they are applied to, then* τ *is* infinite*.*

Given a GTRS $\mathcal{R}$, $\tau_I = (\mathsf{DP}_{HC}(\mathcal{R}),\varphi)$, where $\varphi \in \mathsf{F}$, is called an *initial problem*, from which a proof of (non)termination of $\mathcal{R}$ is initiated. The specific choice of φ is important in practice, as soundness/completeness of some processor may depend of this choice. However, from an implementation point of view (which we do not address in this paper), we could easily adapt the *open DP framework* discussed in [25, Section 6], which leaves φ unespecified until a particular use of processors establishes some requirements to fix it.

7 Processors for the DP Framework for GTRSs

In this section we introduce several *processors* for their use in proofs of termination of GTRSs and illustrate their application with some examples. Processors P_{Inf} and P_{Cyc} are used to prove GTRS problems infinite (Section 7.1). The *SCC processor* P_{SCC} permits the use of graph techniques to *decompose* GTRS problems (Section 7.2). The *subterm processor* $\mathsf{P}_{\pi,\triangleright_\mu}$ removes pairs from P without paying attention to rules in R (Section 7.3). The *Removal Pair Processor* P_{RP} uses terminating relations to *remove* pairs from P (Section 7.4).

7.1 Proving GTRS problems infinite

The following processor detects a simple kind of infinite GTRS problems.

▶ **Definition 28** (Infiniteness processor). *Let $\mathcal{R}$ be a GTRS and $\varphi \in \mathsf{F}$. Then, P_{Inf} is given by $\mathsf{P}_{Inf}(\mathcal{R}, \varphi) = \mathsf{no}$ iff there is $u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P_{\mathcal{R}}$ and substitutions η and θ such that $\eta(c)$ and $\eta(v) = \theta(\eta(u))$ hold.*

▶ **Theorem 29.** *P_{Inf} is sound and a-complete. If $\tau = (\mathcal{R}, \mathsf{m})$ and v in Definition 28 is ground and contains no symbol from $\mathcal{D}_{\mathcal{R}}$, then P_{Inf} is τ-complete.*

▶ **Example 30** (Nontermination of $\mathcal{R}$ in Example 5). For $\mathsf{DP}_{HC}(\mathcal{R})$ in Example 18, and $\tau_I = (\mathsf{DP}_{HC}(\mathcal{R}), \mathsf{a})$, consider (36) in Example 18, i.e., $\mathsf{F}(x', x'') \xrightarrow{\mathsf{t}} \mathsf{F}(x, \mathsf{b}) \Leftarrow x' \approx x, x'' \approx x$, $\eta = \{x \mapsto \mathsf{b}, x' \mapsto \mathsf{b}, x'' \mapsto \mathsf{b}\}$, and θ be the empty substitution. Since η satisfies the conditional part of (36) and $\eta(\mathsf{F}(x, \mathsf{b})) = \mathsf{F}(\mathsf{b}, \mathsf{b}) = \eta(\mathsf{F}(x', x''))$, we have $\mathsf{P}_{Inf}(\tau_I) = \mathsf{no}$. Hence, by completeness of P_{Cyc} (Theorem 32), τ_I is infinite, and $\mathcal{R}$ is not terminating.

Just requiring (i) *feasibility* of c and that (ii) v matches u in Definition 28 does *not* guarantee completeness. For instance, $\alpha : \mathsf{F}(x) \xrightarrow{\mathsf{t}} \mathsf{F}(\mathsf{s}(x)) \Leftarrow x \to^* \mathsf{0}$ is feasible (use $\sigma(x) = \mathsf{0}$). The right-hand side $\mathsf{F}(\mathsf{s}(x))$ matches the left-hand side $\mathsf{F}(x)$. However, there is no infinite chain using α only. In practice, P_{Inf} will be used with pairs $u \xrightarrow{\mathsf{t}} v$ without conditional part. In this way, checking that v matches u, i.e., $v = \theta(u)$ for some substitution θ, suffices.

▶ **Definition 31** (Cycle processor). *Let $\mathcal{R}$ be a GTRS, and $\varphi \in \mathsf{F}$. Then, P_{Cyc} is given by $\mathsf{P}_{Cyc}(\mathcal{R}, \varphi) = \mathsf{no}$ iff there are $n \geq 1$ pairs $\alpha_1, \ldots, \alpha_n \in P_{\mathcal{R}}$ such that the following sequence is $\overline{\mathcal{R}}$-feasible*

$$c_1, v_1 \to^* u_2, \ldots, c_i, v_i \to^* u_{i+1}, \ldots, c_n, v_n \to^* u_1 \tag{41}$$

▶ **Theorem 32.** *P_{Cyc} is sound and a-complete. If $\tau = (\mathcal{R}, \mathsf{m})$ and for all $1 \leq i \leq n$ and v_i in pairs α_i in Definition 31 is ground and contains no symbol from $\mathcal{D}_{\mathcal{R}}$, then P_{Cyc} is τ-complete.*

▶ **Example 33.** Consider $\mathcal{R} = (\mathcal{F}, \Pi, \mu_\top, H, R)$, where $H = \{(42)\}$ and $R = \{(43), (44)\}$:

$$x \approx y \;\Leftarrow\; x \to^* y \tag{42}$$

$$\mathsf{a} \to \mathsf{c}(x) \;\Leftarrow\; x \approx \mathsf{b} \tag{43}$$

$$\mathsf{b} \;\to\; \mathsf{a} \tag{44}$$

Regarding $\mathsf{DP}_{HC}(\mathcal{R})$, $\mathcal{S}ubt(\mathcal{F}, \mu)$ and $\mathcal{M}ark(\mathcal{F})$ are defined as explained above, and we have $H_{DP}(\mathcal{R}) = \{(45)\}$ and $H_{DPC}(\mathcal{R}) = \{(46)\}$, with

$$\mathsf{B} \;\to\; \mathsf{A} \tag{45}$$

$$\mathsf{A} \;\to\; x'' \Leftarrow x \approx \mathsf{b}, x \varpi_{\trianglerighteq_\mu} x', \mathsf{Mk}(x', x'') \tag{46}$$

We prove $\mathcal{R}$ nonterminating by applying P_{Cyc} to $\tau_I = (\mathsf{DP}_{HC}(\mathcal{R}), \mathsf{a})$. The following sequence built using the pairs in $H_{DP}(\mathcal{R})$ and $H_{DPC}(\mathcal{R})$ according to (41):

$$\mathsf{A} \to^* \mathsf{A}, x \approx \mathsf{b}, x \varpi_{\trianglerighteq_\mu} x', \mathsf{Mk}(x', x''), x'' \to^* \mathsf{B} \tag{47}$$

is $\overline{\mathsf{DP}_{HC}(\mathcal{R})}$-feasible, as it is satisfied by $\sigma = \{x \mapsto \mathsf{b}, x' \mapsto \mathsf{b}, x'' \mapsto \mathsf{B}\}$. Thus, $\mathsf{P}_{Cyc}(\tau_I) = \mathsf{no}$.

Note that P_{Inf} does *not* apply to $\mathcal{R}$ in Example 33: the only substitution satisfying the conditional part of (46) is σ in the example. But the instance $\sigma(x'') = \mathsf{B}$ of the right-hand side of (46) does *not* match the left-hand side A of (46). However, P_{Cyc} does *not* subsume P_{Inf}. For instance, $\mathsf{F}(x) \xrightarrow{\mathsf{t}} \mathsf{F}(\mathsf{s}(x))$, where s is not a defined symbol, can be used to define an infinite chain which would be easily detected by P_{Inf}. However, $\mathsf{F}(\mathsf{s}(x)) \to^* \mathsf{F}(x)$ is infeasible and P_{Cyc} could not be used to detect infiniteness of a GTRS-problem involving such a pair.

7.2 The SCC processor

In this section we provide a notion of graph which captures infinite (*minimal*) chains of (dependency) pairs as given in Definition 14.

▶ **Definition 34** (GTRS Graph of Pairs). *Let $\mathcal{R}$ be a GTRS. The* GTRS-graph $\mathsf{G}(\mathcal{R})$ *has $P_{\mathcal{R}}$ as the set of nodes. There is an arc from $\alpha \in P_{\mathcal{R}}$ to $\alpha' \in P_{\mathcal{R}}$ iff α, α' is an $\mathcal{R}$-chain for some substitution σ.*

The following result *approximates* the (in general incomputable) dependency graph using the *infeasibility* of some sequences. Although (in)feasibility is undecidable, a number of techniques and tools have been developed to automatically (dis)prove it, thus providing a practical approach to *approximate* the graph.

▶ **Definition 35** (Estimated GTRS Graph). *Let $\mathcal{R}$ be a GTRS. The* estimated *GTRS-graph* $\mathsf{EG}(\mathcal{R})$ *has $P_{\mathcal{R}}$ as the set of nodes. Let $\alpha : u \xrightarrow{\mathsf{t}} v \Leftarrow c, \alpha' : u' \xrightarrow{\mathsf{t}} v' \Leftarrow c' \in P_{\mathcal{R}}$ be such that $\mathcal{V}ar(\alpha) \cap \mathcal{V}ar(\alpha') = \emptyset$ (rename if necessary). There is an arc from α to α' in* $\mathsf{EG}(\mathcal{R})$ *iff $c, v \to^* u', c'$* cannot *be proved $\overline{\mathcal{R}}$-infeasible.*

The sequence considered in Definition 35 actually characterizes the GTRS chain required in Definition 34 to draw an arc from a node α_1 to a node α_2 (see (15)): there is such an arc iff $c, c', v \to^* u'$ is feasible. Definition 35 provides an *estimation* as only an $\overline{\mathcal{R}}$*-infeasibility proof* is attempted (by using some method, tool, etc.); if it succeeds, the arc is *dismissed*; otherwise, the arc is *included* in the graph. Thus, all nodes and arcs of $\mathsf{G}(\mathcal{R})$ are in $\mathsf{EG}(\mathcal{R})$.

▶ **Example 36.** Consider the GTRS $\mathcal{R}$ in Example 1 and $\mathsf{DP}_{HC}(\mathcal{R})$ in Example 17. The estimated dependency graph $\mathsf{EG}(\mathsf{DP}_{HC}(\mathcal{R}))$ is

For instance, the following sequence, that corresponds to an arc from (20) to (20),

$$y_1 \le x_1, \mathsf{div}(x_1 - y_1, y_1) \approx \mathsf{pair}(q_1, r_1), q_1 \varpi_{\trianglerighteq_\mu} q_1', \mathsf{Mk}(q_1', q_1''),$$
$$y_2 \le x_2, \mathsf{div}(x_2 - y_2, y_2) \approx \mathsf{pair}(q_2, r_2), q_2 \varpi_{\trianglerighteq_\mu} q_2', \mathsf{Mk}(q_2', q_2''), q_1'' \to_{\mathcal{R}}^* \mathsf{DIV}(x_2, y_2)$$

can be proved $\overline{\mathsf{DP}_{HC}(\mathcal{R})}$-infeasible (see [13] for a general treatment of such infeasibility problems, including mechanization issues).

▶ **Definition 37** (SCC processor). *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H \uplus P, R)$ be a GTRS and $\tau = (\mathcal{R}, \varphi)$ be an $\mathcal{R}$-problem. Then, P_{SCC} is given by*

$$\mathsf{P}_{SCC}(\tau) = \{((\mathcal{F}, \Pi, \mu, H \uplus P', R), \varphi) \mid P' \textit{are the nodes of an SCC in } \mathsf{EG}(\mathcal{R})\}$$

▶ **Example 38.** For $\mathcal{R}_0 = \mathsf{DP}_{HC}(\mathcal{R}) = (\mathcal{F}, \Pi, \mu^\sharp, H \uplus \{(19), (20), (21)\}, R)$ in Example 17 and $\mathsf{EG}(\mathcal{R}_0)$ displayed in Example 36, we have $\mathsf{P}_{SCC}(\mathcal{R}_0, \varphi) = \{\tau_1\}$, where $\tau_1 = (\mathcal{R}_1, \varphi)$, with $\mathcal{R}_1 = (\mathcal{F}, \Pi, \mu^\sharp, H \uplus \{(19)\}, R)$.

▶ **Theorem 39.** P_{SCC} *is sound and complete.*

With P_{SCC} we can *separately* work with the strongly connected components of $\mathsf{EG}(\mathcal{R}, \varphi)$, disregarding other parts of the graph.

7.3 Subterm processor

This section generalizes the *subterm processor* for TRSs [14] to GTRSs $\mathcal{R}$. Such a processor removes pairs $u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P_{\mathcal{R}}$ if some immediate subterm $v|_j$ of v is a *strict* active subterm of an immediate subterm $u|_i$ of u, i.e., $u|_i \rhd_\mu v|_j$ holds, where i and j are determined by the root symbols of u and v, respectively, by means of a so-called *simple projection*. The processor does *not* pay attention to the conditional part c of α or to the rules in R. The set of *root symbols* associated to $P_{\mathcal{R}}$ is:

$$Root(P_{\mathcal{R}}) \quad = \quad \{root(u) \mid u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P_{\mathcal{R}}\} \cup \{root(v) \mid u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P_{\mathcal{R}}, v \notin \mathcal{X}\}$$

▶ **Definition 40** (Simple projection). *Let $\mathcal{R}$ be a GTRS. A* simple projection *for $\mathcal{R}$ is a mapping $\pi : Root(P_{\mathcal{R}}) \to \mathbb{N}$ such that $\pi(f) \in \{1, \ldots, ar(f)\}$. The mapping that assigns a subterm $\pi(t) = t|_{\pi(f)}$ to each term t with $root(t) \in Root(P_{\mathcal{R}})$ is also denoted by π; we also let $\pi(x) = x$ if $x \in \mathcal{X}$.*

▶ **Definition 41** (Subterm processor). *Let $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H \uplus P, R)$ be a GTRS, π be a simple projection for $\mathcal{R}$, and $\alpha : u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P$. Then, $\mathsf{P}_{\pi,\rhd_\mu}$ is given by $\mathsf{P}_{\pi,\rhd_\mu}(\mathcal{R}, \varphi) = \{(\mathcal{F}, \Pi, \mu, H \uplus (P - \{\alpha\}), R)\}$, if (i) for all $u' \xrightarrow{\mathsf{t}} v' \Leftarrow c' \in P - \{\alpha\}$ we have that $\pi(u') \trianglerighteq_\mu \pi(v')$ holds and (ii) $\pi(u) \rhd_\mu \pi(v)$ holds.*

Note that $\mathsf{P}_{\pi,\rhd_\mu}$ can *not* be applied to a GTRS problem $(\mathcal{R}, \varphi)$ if $P_{\mathcal{R}}$ contains pairs $u \xrightarrow{\mathsf{t}} v \Leftarrow c$ from $H_{DPC}(\mathcal{R})$. This is because, by definition, v is a fresh variable *not* belonging to u. Thus, neither (i) nor (ii) in Definition 41 are fulfilled and $(\mathcal{R}, \varphi) \notin \mathsf{Dom}(\mathsf{P}_{\pi,\rhd_\mu})$.

▶ **Theorem 42** (Soundness and completeness of $\mathsf{P}_{\pi,\rhd_\mu}$). *$\mathsf{P}_{\pi,\rhd_\mu}$ is complete and τ-sound for all $\tau = (\mathcal{R}, \varphi) \in \mathsf{Dom}(\mathsf{P}_{\pi,\rhd_\mu})$ such that for all $u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P_{\mathcal{R}}$ $v \notin \mathcal{X}$, $\varphi = \mathsf{m}$, and $Root_P(\mathcal{R}) \cap \mathcal{D}_{\mathcal{R}} = \emptyset$.*

▶ **Example 43** (Termination of $\mathcal{R}$ in Example 1). Consider $\tau_1 = (\mathcal{R}_1, \varphi)$ in Example 38 with $P_{\mathcal{R}_1} = \{(19)\}$, where (19) is $\mathsf{s}(x) -^\sharp \mathsf{s}(y) \xrightarrow{\mathsf{t}} x -^\sharp y$. Since $Root(P_{\mathcal{R}_1}) = \{-^\sharp\}$, with $\pi(-^\sharp) = 1$ we obtain $\pi(\mathsf{s}(x) -^\sharp \mathsf{s}(y)) = \mathsf{s}(x) \rhd_\mu x = \pi(x -^\sharp y)$. Hence, $\mathsf{P}_{\pi,\rhd_\mu}(\tau_1) = \{(\mathcal{R}_{11}, \varphi)\}$ where $P_{\mathcal{R}_{11}} = \emptyset$. This proves termination of $\mathcal{R}$ in Example 1, after a "formal" application of P_{SCC} to obtain an empty set of GTRS problems, see Figure 1 left.

7.4 Use of terminating relations

The absence of infinite $\mathcal{R}$-chains can be ensured by using terminating relations on terms.

▶ **Definition 44** (Removal pair). *A* removal pair $(\gtrsim, \sqsupset)$ *consists of relations $\gtrsim$ and $\sqsupset$ on terms such that $\sqsupset$ is terminating and $\sqsupset \circ \gtrsim \subseteq \sqsupset$.*

▶ **Definition 45** (Compatible removal pair). *Let $\mathcal{R}$ be a GTRS. A removal pair $(\gtrsim, \sqsupset)$ is* compatible *with $\mathcal{R}$ if (i) $\to_{\mathcal{R}}^* \subseteq \gtrsim$ and (ii) if $P_{\mathcal{R}}$ is not a singleton, then for all $u \xrightarrow{\mathsf{t}} v \Leftarrow c, u' \xrightarrow{\mathsf{t}} v' \Leftarrow c' \in P_{\mathcal{R}}$ and for all substitutions σ, if $\overline{\mathcal{R}} \vdash \sigma(c), \sigma(v) \to^* \sigma(u'), \sigma(c')$ holds, then (ii.1) $\sigma(u) \gtrsim \sigma(v)$ or (ii.2) $\sigma(u) \sqsupset \sigma(v)$.*

Compatibility guarantees that, if α is followed by another (possibly the same, up to renaming) pair α' in a chain, then, by (i), a non-strict decrease is introduced in the transition $\sigma(v) \to_{\mathcal{R}}^* \sigma(u')$, and, if $P_{\mathcal{R}}$ is not a singleton, then there is a non-strict (ii.1) or a strict (ii.2) decrease in the instances of components u and v of α. In the second case (ii.2), we can *remove* α. This is the purpose of the removal pair processor. If $P_{\mathcal{R}}$ is a singleton, then (ii.1) is useless when the processor is applied to remove the only pair in $P_{\mathcal{R}}$. In this case, (ii.2) is specifically required by the processor, as shown in the following.

▶ **Definition 46** (Removal pair processor). *Let* $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H \uplus P, R)$ *be a GTRS,* $\alpha : u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P$*, and* $(\gtrsim, \sqsupset)$ *be a removal pair compatible with* $\mathcal{R}$*. Then,* $\mathsf{P}_{RP}(\mathcal{R}, \varphi) = \{((\mathcal{F}, \Pi, \mu, H \uplus (P - \{\alpha\}), R), \varphi)\}$ *if for all substitutions* σ*, whenever* $\overline{\mathcal{R}} \vdash \sigma(c)$ *holds, we have* $\sigma(u) \sqsupset \sigma(v)$*.*

▶ **Theorem 47** (Soundness and completeness of P_{RP}). P_{RP} *is sound and complete.*

As discussed in [18, 26], the following *semantic* approach is useful in practice.

▶ **Definition 48** (Semantic version of the removal pair processor). *Let* $\mathcal{R} = (\mathcal{F}, \Pi, \mu, H \uplus P, R)$ *be a GTRS and* $\alpha : u \xrightarrow{\mathsf{t}} v \Leftarrow c \in P$*. Let* $\varpi_{\gtrsim}$ *and* $\varpi_{\sqsupset}$ *be fresh predicate symbols. Let* $\mathcal{A}$ *be a model of* $\overline{\mathcal{R}}$ *such that* $\pi^{\mathcal{A}}_{\sqsupset}$ *is* terminating *on the domain of* $\mathcal{A}$*. Then,* $\mathsf{P}_{RP}(\mathcal{R}, \varphi) = \{((\mathcal{F}, \Pi, \mu, H \uplus (P - \{\alpha\}), R), \varphi)\}$ *if each of the following conditions hold:*

1. $\mathcal{A} \models (\forall x, y, z)\ ((x\ \varpi_{\sqsupset}\ y \wedge y\ \varpi_{\gtrsim}\ z) \Rightarrow x\ \varpi_{\sqsupset}\ z)$*,*
2. $\mathcal{A} \models (\forall x, y)\ (x\ \to^*\ y \Rightarrow x\ \varpi_{\gtrsim}\ y)$*,*
3. *if* P *is not a singleton, then for all* $u' \xrightarrow{\mathsf{t}} v' \Leftarrow c', u'' \xrightarrow{\mathsf{t}} v'' \Leftarrow c'' \in P$*,* $\mathcal{A} \models (\forall \vec{x})((c' \wedge v' \to^* u'' \wedge c'') \Rightarrow (u'\ \varpi_{\gtrsim}\ v' \ \vee\ u'\ \varpi_{\sqsupset}\ v'))$*,*
4. $\mathcal{A} \models (\forall \vec{x})\ (c \Rightarrow u\ \varpi_{\sqsupset}\ v)$*.*

The removal pair $(\gtrsim, \sqsupset)$ implicit in Definition 48 is as follows: for all terms s and t, $s \gtrsim t$ (resp. $s \sqsupset t$) iff for all valuations $\nu : \mathcal{X} \to \mathcal{A}$, $[s]^{\mathcal{A}}_{\nu}\ \varpi^{\mathcal{A}}_{\gtrsim}\ [t]^{\mathcal{A}}_{\nu}$ (resp. $[s]^{\mathcal{A}}_{\nu}\ \varpi^{\mathcal{A}}_{\sqsupset}\ [t]^{\mathcal{A}}_{\nu}$) holds, where for all terms $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, $[t]^{\mathcal{A}}_{\nu}$ denotes the (usual) interpretation of terms t using the interpretations $f^{\mathcal{A}}$ of function symbols $f \in \mathcal{F}$ by $\mathcal{A}$ and the interpretation $\nu(x)$ of variables x by ν. Thus, requirement (1) in Definition 48 guarantees that $\sqsupset \circ \gtrsim\ \subseteq\ \sqsupset$ in Definition 44 holds; (2) and (3) make $(\gtrsim, \sqsupset)$ compatible with $\mathcal{R}$ (Definition 45); and (4) permits the removal of α (Definition 46).

▶ **Example 49.** Consider the following GTRS $\mathcal{R}$ (COPS #330 [28, Ex. 5.1]):

$$x \approx y \quad \Leftarrow \quad x \to^* y \tag{48}$$

$$\mathsf{pin}(x) \to \mathsf{pout}(\mathsf{g}(x)) \tag{49}$$

$$\mathsf{pin}(x) \to \mathsf{pout}(\mathsf{f}(y)) \quad \Leftarrow \quad \mathsf{pin}(x) \approx \mathsf{pout}(\mathsf{g}(y)) \tag{50}$$

In $\mathsf{DP}_{HC}(\mathcal{R})$, we have $H_{DP}(\mathcal{R}) = \emptyset$; and $H_{DPC}(\mathcal{R})$ consists of a single clause:

$$\mathsf{PIN}(x) \xrightarrow{\mathsf{t}} y'' \quad \Leftarrow \quad \mathsf{pin}(x) \approx \mathsf{pout}(\mathsf{g}(y)), y\ \varpi_{\trianglerighteq_\mu}\ y', \mathsf{Mk}(y', y'') \tag{51}$$

With P_{RP} we can remove (51). Since $P_{\mathsf{DP}_{HC}(\mathcal{R})} = \{(51)\}$ is a singleton, requirement (3) in Definition 48 is not necessary. Requirements (1), (2), and (4) amount at considering:

$$(\forall x, y, z) \qquad x\ \varpi_{\sqsupset}\ y \wedge y\ \varpi_{\gtrsim}\ z \Rightarrow x\ \varpi_{\sqsupset}\ z \tag{52}$$

$$(\forall x, y) \qquad x \to^* y \Rightarrow x\ \varpi_{\gtrsim}\ y \tag{53}$$

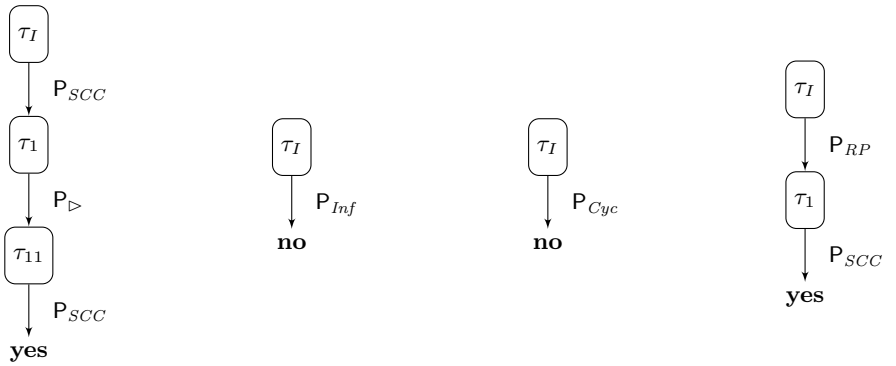
$$(\forall x, y, y', y') \qquad (\mathsf{pin}(x) \approx \mathsf{pout}(\mathsf{g}(y)) \wedge y\ \varpi_{\trianglerighteq_\mu}\ y' \wedge \mathsf{Mk}(y', y'')) \Rightarrow \mathsf{PIN}(x)\ \varpi_{\sqsupset}\ y'' \tag{54}$$

The following structure $\mathcal{A}$ (computed by AGES) with domain $\mathbb{Z} - \mathbb{N}$ (the set of nonpositive integers); function symbols are interpreted as follows:

$$\mathsf{f}^{\mathcal{A}}(x){=}2x \qquad \mathsf{g}^{\mathcal{A}}(x){=}2x \qquad \mathsf{pin}^{\mathcal{A}}(x){=}2x - 1 \qquad \mathsf{pout}^{\mathcal{A}}(x){=}x - 1 \qquad \mathsf{PIN}^{\mathcal{A}}(x){=}2x$$

and predicate symbols as follows:

$$x\ \approx^{\mathcal{A}}\ y \Leftrightarrow y \geq x \qquad x\ \to^{\mathcal{A}}\ y \Leftrightarrow y \geq x \qquad x\ (\to^*)^{\mathcal{A}}\ y \Leftrightarrow y \geq x$$
$$x\ (\xrightarrow{\mathsf{t}})^{\mathcal{A}}\ y \Leftrightarrow \mathit{true} \qquad x\ \varpi^{\mathcal{A}}_{\trianglerighteq_\mu}\ y \Leftrightarrow y \geq x \qquad \mathsf{Mk}^{\mathcal{A}}(x, y) \Leftrightarrow y > x$$
$$x\ \varpi^{\mathcal{A}}_{\gtrsim}\ y \Leftrightarrow y \geq x \qquad x\ \varpi^{\mathcal{A}}_{\sqsupset}\ y \Leftrightarrow y > x$$



Examples 1, 38 & 43 | Examples 5 & 30 | Example 33 | Example 49

Figure 1 Proofs of (non)termination in the DP Framework for the running examples.

is a model of $\overline{\mathsf{DP}_{HC}(\mathcal{R})} \cup \{(52), (53), (54)\}$. Hence (51) can be removed from $\mathsf{DP}_{HC}(\mathcal{R})$ to obtain a GTRS $\mathcal{R}_1$, with $P_{\mathcal{R}_1} = \emptyset$, i.e., $\mathsf{P}_{RP}(\mathsf{DP}_{HC}(\mathcal{R}), \varphi) = \{\tau_1\}$, where $\tau_1 = (\mathcal{R}_1, \varphi)$. Now, $\mathsf{P}_{SCC}(\mathcal{R}_1, \varphi) = \emptyset$ (due to $P_{\mathcal{R}_1} = \emptyset$) completes the proof.

The proof trees for our running examples are shown in Figure 1.

8 Related work

The results in this paper extend and generalize the treatment of termination of *oriented* CTRSs presented in [24] which is implemented using the 2D DP Framework for oriented (2-)CTRSs [25, 26]. Furthermore, some improvements on [25, 26] are obtained:

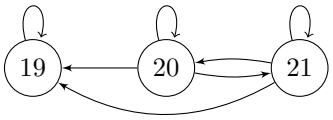
- In contrast to [24], Definition 16 provides a uniform definition of $\mathsf{DP}_{HC}(\mathcal{R})$ with pairs of two classes, $H_{DP}(\mathcal{R})$ and $H_{DPC}(\mathcal{R})$, but with a single definition of $\mathcal{R}$-chain. Also, the use of atoms $x \varpi_{\trianglerighteq_\mu} x'$ and $\mathsf{Mk}(x', x'')$ in the conditional part of pairs in $H_{DPC}(\mathcal{R})$ (also in contrast to [24, Definition 41]) often helps to prove $\overline{\mathsf{DP}_{HC}(\mathcal{R})}$-infeasibility of pairs in $H_{DPC}(\mathcal{R})$, which is helpful. For instance, (20) and (21) are actually $\overline{\mathsf{DP}_{HC}(\mathcal{R})}$-infeasible for $\mathcal{R}$ in Example 1. This would not happen with the corresponding pairs obtained from [24, Definition 41]. We included (20) and (21) here just to illustrate the use of P_{SCC}. A simpler proof would be obtained otherwise.
- Proofs of termination of CTRSs in the 2D DP Framework are restricted, in practice, to 2-CTRSs, i.e., those whose rules $\ell \to r \Leftarrow c$ satisfy that all variables in r also occur in ℓ [25, Theorem 16(1)].[2] Neither the CTRSs $\mathcal{R}$ in Examples 5 and 49 nor Example 1, viewed as a CTRS, are 2-CTRSs.
- As a consequence of [25, Theorem 16(1)], only processors which are sound on *arbitrary* chains can be used in proofs of termination in the 2D DP Framework for CTRSs. For instance, $\mathsf{P}_{\triangleright}$ cannot be used, as it is sound on minimal chains only. In contrast, our termination proof of $\mathcal{R}$ in Example 1 uses $\mathsf{P}_{\triangleright}$.
- P_{Cyc} was not considered in [25, 26] and can be used to detect infinite chains involving an arbitrary number of pairs. Also, P_{Cyc} can be implemented as a *feasibility problem*.

[2] Actually, [25, Theorem 16(1)] uses the more general requirement of rules that *preserve terminating substitutions* discussed in [24, Section 4.2]. A sufficient condition guaranteeing the property is being a 2-CTRS [24, Proposition 37].

```
mod Ex9_SchGra10 is
 sorts Boolean S .
 op true : -> Boolean .                  op 0 : -> S .       op s : S -> S .
 ops _>_ : S S -> Boolean [frozen] .     op _<=_ : S S -> Boolean [frozen] .
 op _-_ : S S -> S .                     op pair : S S -> S .
 op div : S S -> S .                     var q r x y : S .
 eq s(x) > 0 = true .          eq s(x) > s(y) = x > y .   eq 0 <= x = true .
 eq s(x) <= s(y) = x <= y .    rl x - 0 => x .
 rl 0 - y => y .               rl s(x) - s(y) => x - y .
 crl div(x,y) => pair(0,x) if y > x = true .
 crl div(x,y) => pair(s(q),r) if y <= x = true /\ div(x - y,y) => pair(q,r) .
endm
```

Figure 2 Maude enconding of the GTRS $\mathcal{R}$ in Example 1.

Approximations of the dependency graph for CTRSs in [25, Section 7.1.1] pay *no attention* to the conditional parts of pairs [25, Definitions 55 & 56]. For instance, for $\mathsf{DP}_{HC}(\mathcal{R})$ in Example 17, instead of the graph in Example 36, the techniques in [25] produce:



GTRSs also include CS-CTRS [19, Section 8.1], for which no technique for proving termination is known. Also, computations with Generalized Rewrite Theories (GRTs [4]) and Maude programs [5, 6] can often be simulated using (Equational) GTRSs as briefly discussed in [20, 21]. Vice versa, Maude can be used as a practical platform to use GTRSs.

▶ **Example 50.** The Maude module in Figure 2 provides an appropriate translation of $\mathcal{R}$ in Example 1 to Maude. Predicates are treated as *boolean* functions defined by *equations* obtained from the corresponding definite Horn clauses. Also note that the arguments of "predicate" function symbols are all *frozen*, thus disabling reductions on their arguments. This makes Maude computations with `Ex9_SchGra10` closer to $\mathcal{R}$, viewed as a GTRS. For instance, the result of the integer division of 3 by 2 is computed as follows:

```
Maude> rew div(s(s(s(0))),s(s(0))) .
rewrite in Ex9_SchGra10 : div(s(s(s(0))), s(s(0))) .
rewrites: 12 in 0ms cpu (0ms real) (363636 rewrites/second)
result S: pair(s(0), s(0))
```

Thus, our results provide a basis for the development of techniques and tools for proving termination of GRTs and Maude programs.

Finally, at first sight, our techniques would also apply to prove termination of *Logically Constrained TRSs* (LCTRSs [15, 16]). LCTRSs and GTRSs differ in the treatment of the conditional part of the rules: LCTRSs use fixed interpretations and GTRSs additional Horn clauses. Although rewriting with LCTRSs (which requires substitutions that *respect* the rules [15, page 347]) is, in general, more restrictive than rewriting with GTRSs, our notion of dependency pair would capture termination of LCTRSs, although the treatment of the DP problems would be different due to the semantic treatment of conditions.

9 Conclusions and future work

We have presented a Dependency Pair Framework for (dis)proving termination of GTRSs. To the best of our knowledge, this is the first time that the DP Framework has been adapted to deal with GTRSs. It extends previous proposals for proving termination of CTRSs [25]. As discussed in the previous section, even when applied to CTRSs (as particular GTRSs), we are able to obtain proofs of termination which are not possible in the 2D DP Framework. Also, the proof of termination of the CTRS $\mathcal{R}$ in Example 5 supplied with a replacement map μ (see Example 8) cannot be obtained either in the DP Framework for CSR (as it does not consider conditional rules) or in the 2D DP Framework (replacement maps are not allowed).

From a theoretical point of view, a lot of work remains to be done. For instance, the refinements in the treatement of frozen subterms in the right-hand sides of unconditional rules developed in [10, 11] should be adapted to the new conditional setting. Also, the improvements on the reduction pair processor obtained by considering powerful notions such as that of *usable rule*, which are already in use both for CS-TRSs [11, Section 6] and CTRSs [26, Section 4.4] should also be developed for GTRSs and the DP Framework introduced here. Also, extending the framework to deal with *operational termination* of GTRSs is also important, as operational termination provides the best conditions to compute with rewriting-based systems using conditional rules (see [23]). The approach for CTRSs developed in [25] (dealing with the *vertical* dimension of operational termination by means of an additional set of dependency pairs) should be thoroughly revised for GTRSs $\mathcal{R}$ as operational termination of GTRSs depends on proofs of atoms using the Horn theory component H of $\mathcal{R}$.

The *implementation* of the DP Framework for proving termination of GTRSs introduced here is also an important subject for future work. Regarding the implementation of our techniques, we are improving our tool MU-TERM [13] with the ability to deal with GTRSs. However, this actually depends on other tools, in particular infChecker [13], still unable to deal with GTRSs. Also, as discussed in [21], GTRSs can be useful to investigate computational properties of GRTs and Maude programs, which implement GRTs.

References

1 Beatriz Alarcón, Raúl Gutiérrez, and Salvador Lucas. Context-sensitive dependency pairs. *Inf. Comput.*, 208(8):922–968, 2010. doi:10.1016/j.ic.2010.03.003.

2 Thomas Arts and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theor. Comput. Sci.*, 236(1-2):133–178, 2000. doi:10.1016/S0304-3975(99)00207-8.

3 Franz Baader and Tobias Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998. doi:10.1017/CBO9781139172752.

4 Roberto Bruni and José Meseguer. Semantic foundations for generalized rewrite theories. *Theor. Comput. Sci.*, 360(1-3):386–414, 2006. doi:10.1016/j.tcs.2006.04.012.

5 Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn L. Talcott. *All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic*, volume 4350 of *Lecture Notes in Computer Science*. Springer, 2007. doi:10.1007/978-3-540-71999-1.

6 Francisco Durán, Steven Eker, Santiago Escobar, Narciso Martí-Oliet, José Meseguer, Rubén Rubio, and Carolyn L. Talcott. Programming and symbolic computation in Maude. *J. Log. Algebr. Meth. Program.*, 110, 2020. doi:10.1016/j.jlamp.2019.100497.

7 Melvin Fitting. *First-Order Logic and Automated Theorem Proving, Second Edition.* Graduate Texts in Computer Science. Springer, 1996. doi:10.1007/978-1-4612-2360-3.

8 Jürgen Giesl, René Thiemann, and Peter Schneider-Kamp. The dependency pair framework: Combining techniques for automated termination proofs. In Franz Baader and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, 11th International Conference, LPAR 2004, Proceedings*, volume 3452 of *Lecture Notes in Computer Science*, pages 301–331. Springer, 2004. doi:10.1007/978-3-540-32275-7_21.

9 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, and Stephan Falke. Mechanizing and improving dependency pairs. *J. Autom. Reasoning*, 37(3):155–203, 2006. doi:10.1007/s10817-006-9057-7.

10 Raúl Gutiérrez and Salvador Lucas. Proving termination in the context-sensitive dependency pair framework. In Peter Csaba Ölveczky, editor, *Rewriting Logic and Its Applications - 8th International Workshop, WRLA 2010, Held as a Satellite Event of ETAPS 2010, Revised Selected Papers*, volume 6381 of *Lecture Notes in Computer Science*, pages 18–34. Springer, 2010. doi:10.1007/978-3-642-16310-4_3.

11 Raúl Gutiérrez and Salvador Lucas. Function calls at frozen positions in termination of context-sensitive rewriting. In Narciso Martí-Oliet, Peter Csaba Ölveczky, and Carolyn L. Talcott, editors, *Logic, Rewriting, and Concurrency - Essays dedicated to José Meseguer on the Occasion of His 65th Birthday*, volume 9200 of *Lecture Notes in Computer Science*, pages 311–330. Springer, 2015. doi:10.1007/978-3-319-23165-5_15.

12 Raúl Gutiérrez and Salvador Lucas. Automatic Generation of Logical Models with AGES. In Pascal Fontaine, editor, *Automated Deduction - CADE 27 - 27th International Conference on Automated Deduction, Proceedings*, volume 11716 of *Lecture Notes in Computer Science*, pages 287–299. Springer, 2019. doi:10.1007/978-3-030-29436-6_17.

13 Raúl Gutiérrez and Salvador Lucas. Automatically Proving and Disproving Feasibility Conditions. In Nicolas Peltier and Viorica Sofronie-Stokkermans, editors, *Automated Reasoning - 10th International Joint Conference, IJCAR 2020, Proceedings, Part II*, volume 12167 of *Lecture Notes in Computer Science*, pages 416–435. Springer, 2020. doi:10.1007/978-3-030-51054-1_27.

14 Nao Hirokawa and Aart Middeldorp. Dependency pairs revisited. In Vincent van Oostrom, editor, *Rewriting Techniques and Applications, 15th International Conference, RTA 2004, Aachen, Germany, June 3-5, 2004, Proceedings*, volume 3091 of *Lecture Notes in Computer Science*, pages 249–268. Springer, 2004. doi:10.1007/978-3-540-25979-4_18.

15 Cynthia Kop and Naoki Nishida. Term rewriting with logical constraints. In Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt, editors, *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings*, volume 8152 of *Lecture Notes in Computer Science*, pages 343–358. Springer, 2013. doi:10.1007/978-3-642-40885-4_24.

16 Cynthia Kop and Naoki Nishida. Constrained term rewriting tool. In Martin Davis, Ansgar Fehnker, Annabelle McIver, and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning - 20th International Conference, LPAR-20 2015, Suva, Fiji, November 24-28, 2015, Proceedings*, volume 9450 of *Lecture Notes in Computer Science*, pages 549–557. Springer, 2015. doi:10.1007/978-3-662-48899-7_38.

17 Salvador Lucas. Context-sensitive Rewriting. *ACM Comput. Surv.*, 53(4):78:1–78:36, 2020. doi:10.1145/3397677.

18 Salvador Lucas. Using Well-Founded Relations for Proving Operational Termination. *J. Autom. Reasoning*, 64(2):167–195, 2020. doi:10.1007/s10817-019-09514-2.

19 Salvador Lucas. Applications and extensions of context-sensitive rewriting. *Journal of Logical and Algebraic Methods in Programming*, 121:100680, 2021. doi:10.1016/j.jlamp.2021.100680.

20 Salvador Lucas. Confluence of Conditional Rewriting Modulo. In Aniello Murano and Alexandra Silva, editors, *32nd EACSL Annual Conference on Computer Science Logic (CSL 2024)*, volume 288 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 37:1–37:21, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CSL.2024.37.

21 Salvador Lucas. Local confluence of conditional and generalized term rewriting systems. *Journal of Logical and Algebraic Methods in Programming*, 136:paper 100926, pages 1–23, 2024. doi:10.1016/j.jlamp.2023.100926.

22 Salvador Lucas, Claude Marché, and José Meseguer. Operational termination of conditional term rewriting systems. *Inf. Process. Lett.*, 95(4):446–453, 2005. doi:10.1016/j.ipl.2005.05.002.

23 Salvador Lucas and José Meseguer. Normal forms and normal theories in conditional rewriting. *J. Log. Algebr. Meth. Program.*, 85(1):67–97, 2016. doi:10.1016/j.jlamp.2015.06.001.

24 Salvador Lucas and José Meseguer. Dependency pairs for proving termination properties of conditional term rewriting systems. *J. Log. Algebr. Meth. Program.*, 86(1):236–268, 2017. doi:10.1016/j.jlamp.2016.03.003.

25 Salvador Lucas, José Meseguer, and Raúl Gutiérrez. The 2D Dependency Pair Framework for conditional rewrite systems. Part I: Definition and basic processors. *J. Comput. Syst. Sci.*, 96:74–106, 2018. doi:10.1016/j.jcss.2018.04.002.

26 Salvador Lucas, José Meseguer, and Raúl Gutiérrez. The 2D Dependency Pair Framework for Conditional Rewrite Systems. Part II: Advanced Processors and Implementation Techniques. *J. Autom. Reasoning*, 64(8):1611–1662, 2020. doi:10.1007/s10817-020-09542-3.

27 Elliott Mendelson. *Introduction to mathematical logic (4. ed.)*. Chapman and Hall, 1997.

28 Enno Ohlebusch. Termination of logic programs: Transformational methods revisited. *Appl. Algebra Eng. Commun. Comput.*, 12(1/2):73–116, 2001. doi:10.1007/s002000100064.

29 Enno Ohlebusch. *Advanced Topics in Term Rewriting*. Springer, 2002. doi:10.1007/978-1-4757-3661-8.

30 Dag Prawitz. *Natural deduction. A proof theoretical study*. Stockholm Studies in Philosophy. Almqvist & Wiksell, 1965.

31 John Alan Robinson. A Machine-Oriented Logic Based on the Resolution Principle. *J. ACM*, 12(1):23–41, 1965. doi:10.1145/321250.321253.

32 Felix Schernhammer and Bernhard Gramlich. Characterizing and proving operational termination of deterministic conditional term rewriting systems. *J. Log. Algebraic Methods Program.*, 79(7):659–688, 2010. doi:10.1016/j.jlap.2009.08.001.

33 Terese. *Term rewriting systems*, volume 55 of *Cambridge tracts in theoretical computer science*. Cambridge University Press, 2003.

34 Yoshihito Toyama and Michio Oyamaguchi. Church-rosser property and unique normal form property of non-duplicating term rewriting systems. In Nachum Dershowitz and Naomi Lindenstrauss, editors, *Conditional and Typed Rewriting Systems, 4th International Workshop, CTRS-94, Jerusalem, Israel, July 13-15, 1994, Proceedings*, volume 968 of *Lecture Notes in Computer Science*, pages 316–331. Springer, 1994. doi:10.1007/3-540-60381-6_19.