2D Dependency Pairs for Proving Operational Termination of CTRSs*

Salvador Lucas[1,2] and José Meseguer[2]

[1] DSIC, Universitat Politècnica de València, Spain
[2] CS Dept. University of Illinois at Urbana-Champaign, IL, USA

Abstract. The notion of *operational termination* captures nonterminating computations due to subsidiary processes that are necessary to issue a *single* 'main' step but which often remain 'hidden' when the main computation sequence is observed. This highlights *two dimensions* of nontermination: one for the infinite sequencing of computation steps, and the other that concerns the proof of some single steps. For conditional term rewriting systems (CTRSs), we introduce a new *dependency pair framework* which exploits the *bidimensional* nature of conditional rewriting (rewriting steps + satisfaction of the conditions as reachability problems) to obtain a powerful and more expressive framework for proving operational termination of CTRSs.

Keywords: Conditional term rewriting, dependency pairs, program analysis, operational termination.

1 Introduction

Assume that we have an *interpreter* for a logic $\mathcal{L}$, i.e., an *inference machine* that, given a theory $\mathcal{S}$ and a goal formula φ, will try to incrementally build a proof tree for φ. Intuitively, we call $\mathcal{S}$ *terminating* if for any φ the interpreter either finds a proof in finite time, or fails in all possible attempts also in finite time. The notion of *operational termination* captures this idea, meaning that, given an initial goal, an interpreter will either succeed in finite time in producing a closed proof tree, or will fail in finite time, not being able to close or extend further any of the possible proof trees, after exhaustively searching all such proof trees [12]. In particular, operational termination captures a *'vertical' dimension* of the termination behavior which is missing in the usual definition of termination of relations as *well-founded*, i.e., "without infinite reduction sequences" (the 'horizontal' dimension).

Available tools for proving operational termination of conditional rewriting (AProVE [10] or VMTL [16]) rely on *transformations* $\mathcal{U}$ that map each operational termination problem for the CTRS $\mathcal{R}$ into a termination problem for a

* Research partially supported by NSF grant CNS 13-19109. Salvador Lucas' research was developed during a sabbatical year at the CS Dept. of the UIUC and was also partially supported by Spanish MECD grant PRX12/00214, MINECO project TIN2010-21062-C02-02, and GV grant BEST/2014/026 and project PROMETEO/2011/052.

TRS $\mathcal{U}(\mathcal{R})$. Then, available methods for proving termination of $\mathcal{U}(\mathcal{R})$ are used. However, this transformational approach has substantial limitations.

Example 1. Consider the following CTRS $\mathcal{R}$ [15, Example 8]

$$h(d) \to c(a) \tag{1}$$

$$h(d) \to c(b) \tag{2}$$

$$f(k(a), k(b), x) \to f(x, x, x) \tag{3}$$

$$g(x) \to k(y) \Leftarrow h(x) \to d, h(x) \to c(y) \tag{4}$$

As reported in [15, Example 8], $\mathcal{U}(\mathcal{R})$ is *not* terminating. However, our methods in this paper will show that $\mathcal{R}$ is operationally terminating (Example 19).

Most termination tools for proving termination of (variants of) rewriting with TRSs implement extensions or generalizations of the *Dependency Pair Framework* [7, 8]. The main idea is the following: the rules $\ell \to r$ that are able to produce infinite sequences are those whose right-hand side r contains (possibly recursive) *function calls.* The calls associated to $\ell \to r$ are represented as new rules $u \to v$, that are collected in a new TRS $\mathsf{DP}(\mathcal{R})$ of *dependency pairs* (DPs); $\mathcal{R}$ and $\mathsf{DP}(\mathcal{R})$ determine *dependency chains* whose finiteness characterize termination of $\mathcal{R}$ [1].

In this paper we generalize this approach to *deterministic* 3-*CTRSs*, which are the basis of rewriting-based languages like CafeOBJ [5] or Maude [3]. In Section 3 we show that computations starting from *minimal* operationally nonterminating terms can always follow a precise path where two sources of nontermination can be identified: infinite sequences of rewriting steps (an *horizontal* dimension), and infinitely many attempts to check the satisfaction of the conditions in the rules (a *vertical* dimension). Section 4 introduces a definition of dependency pairs that makes such a bidimensional nature of infinite computations explicit (we call them *2D DPs*). The corresponding notion of chain of dependency pairs permits a completely *independent* treatment of both dimensions of the termination problems. For 2-CTRSs (a subclass of 3-CTRSs), we characterize *termination* (i.e., the absence of infinite rewrite sequences) in terms of the "horizontal" component of our 2D DPs only. In Section 5, we adapt the *Dependency Pair Framework* [7, 8] to mechanize proofs of operational termination of deterministic 3-CTRSs using 2D DPs. The framework can also be used to prove *termination* of 2-CTRSs which are *not* operationally terminating.

Example 2. The following deterministic 2-CTRS $\mathcal{R}$:

$$g(a) \to c(b) \tag{5}$$

$$b \to f(a) \tag{6}$$

$$f(x) \to x \Leftarrow g(x) \to c(y) \tag{7}$$

is *not* operationally terminating. However, it is *terminating.* We can prove *both things* in our framework (see Examples 13 and 15), illustrating its expressiveness.

Section 6 develops the framework by introducing a number of *processors* and illustrating their use. Section 7 discusses related work and concludes.

$$\text{(Refl)} \quad \frac{}{t \to^* t}$$

$$\text{(Tran)} \quad \frac{s \to u \qquad u \to^* t}{s \to^* t}$$

$$\text{(Cong)} \quad \frac{s_i \to t_i}{f(s_1, \ldots, s_i, \ldots, s_k) \to f(s_1, \ldots, t_i, \ldots, s_k)}$$
$$\text{for all } f \in \mathcal{F} \text{ and } 1 \leq i \leq k = ar(f)$$

$$\text{(Repl)} \quad \frac{\sigma(s_1) \to^* \sigma(t_1) \;\; \ldots \;\; \sigma(s_n) \to^* \sigma(t_n)}{\sigma(\ell) \to \sigma(r)}$$
$$\text{for all rules } \ell \to r \Leftarrow s_1 \to t_1 \cdots s_n \to t_n \in \mathcal{R}$$
$$\text{and substitutions } \sigma.$$

Fig. 1. Inference rules for conditional rewriting

2 Preliminaries

Recall from [14] the usual notions and notations regarding term rewriting and CTRSs. An (*oriented*) CTRS $\mathcal{R}$ is a pair $\mathcal{R} = (\mathcal{F}, R)$ where $\mathcal{F}$ is a signature and R a set of rules $\ell \to r \Leftarrow s_1 \to t_1, \cdots, s_n \to t_n$, where the conditions $s_i \to t_i$ for $1 \leq i \leq n$ are intended to express the reachability of (instances of) t_i from (instances of) s_i. As usual, ℓ and r are called the left- and right-hand sides of the rule, and the sequence $s_1 \to t_1, \cdots, s_n \to t_n$ (often abreviated to c) is the *conditional part* of the rule. Rewrite 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. An n-CTRS contains only rewrite rules of type $m \leq n$. An oriented 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 $\mathcal{R} = (\mathcal{F}, R)$, we consider $\mathcal{F}$ as the disjoint union $\mathcal{F} = \mathcal{C} \uplus \mathcal{D}$ of symbols $c \in \mathcal{C}$ (called *constructors*) and symbols $f \in \mathcal{D}$ (called *defined functions*), where $\mathcal{D} = \{root(l) \mid (l \to r \Leftarrow c) \in R\}$ and $\mathcal{C} = \mathcal{F} - \mathcal{D}$. Terms $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ such that $root(t) \in \mathcal{D}$ are called *defined* terms. Terms in $\mathcal{T}(\mathcal{C}, \mathcal{X})$ are called *constructor* terms. $Pos_{\mathcal{D}}(t)$ is the set of positions p of subterms $t|_p$ such that $root(t|_p) \in \mathcal{D}$.

We say that a proof tree T is *closed* whenever it is finite and contains no open goals; it is *well-formed* if it is either an open goal, or a closed proof tree, or a derivation tree of the form $\frac{T_1 \quad \cdots \quad T_n}{G}$ where, for each j, T_j is itself well-formed, and there is $i \leq n$ such that T_i is not closed, for any $j < i$ T_j is closed, and each of the $T_{i+1}, \ldots, T_n$ is an open goal [12]. An infinite proof tree is *well-formed* if it is an ascending chain of well-formed finite proof trees. Intuitively, well-formed trees are the trees that an interpreter would incrementally build when trying to solve one condition at a time from left to right. We write $s \to_{\mathcal{R}} t$ (resp. $s \to_{\mathcal{R}}^* t$)

iff there is a closed proof tree for $s \to t$ (resp. $s \to^* t$) using the inference system in Figure 1. If $s \to_{\mathcal{R}} t$, then there is $\ell \to r \Leftarrow c \in \mathcal{R}$ and $p \in \mathcal{P}os(s)$ such that $s|_p = \sigma(\ell)$ for some substitution σ (written $s \overset{p}{\to}_{\mathcal{R}} t$). If $s \to^*_{\mathcal{R}} t$, then there is $n \geq 0$ and a sequence $(s_i \overset{p_i}{\to}_{\mathcal{R}} s_{i+1})_{1 \leq i \leq n}$ where $s = s_1$ and $s_{n+1} = t$; we write $s \xrightarrow{>\Lambda}{}^* t$ if $p_i > \Lambda$ for $1 \leq i \leq n$. A CTRS $\mathcal{R}$ is *operationally terminating* if no infinite well-formed tree for a goal $s \to^*_{\mathcal{R}} t$ exists; $\mathcal{R}$ is *terminating* if there is no infinite sequence $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots$.

3 Minimal operationally nonterminating terms in CTRSs

Given a proof tree T, *root*(T) is the formula (goal) at the root of the tree, and *left*(G) is the left-hand side s of goal G, where G is $s \to t$ or $s \to^* t$ for some terms s and t.

Definition 1 (Operationally nonterminating term). *Let $\mathcal{R}$ be a CTRS. A term t such that left$(root(T)) = t$ for an infinite well-formed proof tree T is called* operationally nonterminating. *If there is no infinite well-formed proof tree T such that left$(root(T)) = t$, then we call t* operationally terminating.

Definition 2 (Minimality). *Let $\mathcal{R}$ be a CTRS. An operationally nonterminating term t is called* minimal *if every strict subterm u of t (i.e., $t \rhd u$) is* operationally terminating. *Let $\mathcal{T}_{op\text{-}\infty}$ be the set of minimal operationally nonterminating terms associated to $\mathcal{R}$.*

The following lemma shows that operationally nonterminating terms always contain a *minimal* operationally nonterminatin term.

Lemma 1. *Let $\mathcal{R} = (\mathcal{F}, R)$ be a CTRS and $s \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. If s is operationally nonterminating, then there is a subterm t of s ($s \unrhd t$) such that $t \in \mathcal{T}_{op\text{-}\infty}$.*

Proposition 1 below establishes that, for $t \in \mathcal{T}_{\text{op-}\infty}$, there is a precise way for an infinite computation to proceed. Roughly speaking, a rule $\ell \to r \Leftarrow \bigwedge_{i=1}^n s_i \to t_i$ must be used to try a *root-step* on a reduct of t. Then, there is a minimal operationally nonterminating subterm which is either (1) an instance of a non-variable subterm of the *right-hand side* r of the rule (so that the infinite computation continues through the *horizontal* dimension), or (2) an instance of a non-variable subterm of one of the *left-hand sides* s_i of a condition $s_i \to t_i$ (the infinite computation continues through the *vertical* dimension). Given a term t, $\mathcal{D}Subterm(\mathcal{R}, t) = \{t|_p \mid p \in \mathcal{P}os_{\mathcal{D}}(t)\}$ is the set of defined subterms of t with respect to rules in $\mathcal{R}$. Let *DRules*$(\mathcal{R}, t)$ be the set of (possibly conditional) rules in $\mathcal{R}$ defining *root*(t) which *depend* on other defined symbols in $\mathcal{R}$:

$$DRules(\mathcal{R}, t) = \{\ell \to r \Leftarrow c \in \mathcal{R} \mid root(\ell) = root(t), r \notin \mathcal{T}(\mathcal{C}, \mathcal{X})\}.$$

The *dependency* is captured as $r \notin \mathcal{T}(\mathcal{C}, \mathcal{X})$ in the above definition.

Example 3. For $\mathcal{R}$ in Example 1, $DRules(\mathcal{R}, h(x)) = \emptyset$ (because $c(a), c(b) \in \mathcal{T}(\mathcal{C}, \mathcal{X})$), $DRules(\mathcal{R}, g(x)) = \emptyset$ (again $k(y) \in \mathcal{T}(\mathcal{C}, \mathcal{X})$) and $DRules(\mathcal{R}, f(x, x, x)) = \{(\mathit{3})\}$.

For each $v \in \mathcal{D}Subterm(\mathcal{R}, r)$, $DRules(\mathcal{R}, v)$ contains the rules that *will (eventually) be used* in root steps $\sigma(\ell) \to \sigma(r)$ for some $\ell \to r \Leftarrow c \in DRules(\mathcal{R}, v)$ in the *immediate* continuation of the infinite computation in the *horizontal* dimension (starting from an instance $\sigma(v)$ of v). With regard to the *vertical* dimension, given a term t, the set of 'proper' conditional rules defining $root(t)$ is:

$$Rules_C(\mathcal{R}, t) = \{\ell \to r \Leftarrow \bigwedge_{i=1}^{n} s_i \to t_i \in \mathcal{R} \mid root(\ell) = root(t), n > 0\}.$$

We let $URules(\mathcal{R}, t) = DRules(\mathcal{R}, t) \cup Rules_C(\mathcal{R}, t)$.

Example 4. For $\mathcal{R}$ in Example 1, $URules(\mathcal{R}, h(x)) = DRules(\mathcal{R}, h(x)) = \emptyset$ and $URules(\mathcal{R}, f(x, x, x)) = DRules(\mathcal{R}, f(x, x, x)) = \{(3)\}$. However $URules(\mathcal{R}, g(x)) = Rules_C(\mathcal{R}, g(x)) = \{(4)\}$.

Proposition 1. *Let $\mathcal{R}$ be a deterministic 3-CTRS. Then, for all $t \in \mathcal{T}_{op\text{-}\infty}$, there exist $\alpha : \ell \to r \Leftarrow \bigwedge_{i=1}^{n} s_i \to t_i$ and a substitution σ such that $t \xrightarrow{>\Lambda}{}^{*} \sigma(\ell)$, and there is a term v such that $\ell \ntrianglerighteq v$, $\sigma(v) \in \mathcal{T}_{op\text{-}\infty}$ and either*

1. *$\alpha \in DRules(\mathcal{R}, t)$, for all $1 \leq i \leq n$, $\sigma(s_i)$ is operationally terminating and $\sigma(s_i) \to^* \sigma(t_i)$, and $v \in \mathcal{D}Subterm(\mathcal{R}, r)$ is such that $URules(\mathcal{R}, v) \neq \emptyset$, or*
2. *$\alpha \in Rules_C(\mathcal{R}, t)$, there is i, $1 \leq i \leq n$ such that $\sigma(s_j)$ is operationally terminating and $\sigma(s_j) \to^* \sigma(t_j)$ for all j, $1 \leq j < i$, and $v \in \mathcal{D}Subterm(\mathcal{R}, s_i)$ is such that $URules(\mathcal{R}, v) \neq \emptyset$.*

Remark 1. In the following we do *not* impose that the domain of the substitutions be finite. 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).

The next result formalizes a *bidimensional* view of infinite computations starting from minimal operational nonterminating terms: they can be viewed as a path over $\mathbb{N} \times \mathbb{N}$, where each bidimensional point (x_i, y_i) is labeled with a rule α_i.

Theorem 1. *Let $\mathcal{R} = (\mathcal{F}, R)$ be a deterministic 3-CTRS and $t \in \mathcal{T}_{op\text{-}\infty}$. There is a substitution σ and an infinite sequence $\{(x_i, y_i, \alpha_i)\}_{i \in \mathbb{N}}$ of triples $(x_i, y_i, \alpha_i) \in \mathbb{N} \times \mathbb{N} \times R$ such that, for all $i \geq 0$, $x_{i+1} + y_{i+1} = x_i + y_i + 1$ and*

1. *$x_0 = y_0 = 0$, $\alpha_0 \in URules(\mathcal{R}, t)$ and $t \xrightarrow{>\Lambda}{}^{*} \sigma(\ell_0)$.*
2. *For all $i \geq 0$, and $\alpha_i : \ell_i \to r_i \Leftarrow \bigwedge_{j=1}^{n_i} s^i_j \to t^i_j \in R$, we have $\sigma(\ell_i) \in \mathcal{T}_{op\text{-}\infty}$; furthermore, there is a term v_i such that $\ell_i \ntrianglerighteq v_i$, $\sigma(v_i) \in \mathcal{T}_{op\text{-}\infty}$, $\sigma(v_i) \xrightarrow{>\Lambda}{}^{*} \sigma(\ell_{i+1})$, $\alpha_{i+1} \in URules(v_i)$, and*
   (a) *If $x_{i+1} = x_i + 1$, then $v_i \in \mathcal{D}Subterm(\mathcal{R}, r_i)$ and $\alpha_i \in DRules(\mathcal{R}, \ell_i)$.*
   (b) *If $y_{i+1} = y_i + 1$, then there is j, $1 \leq j \leq n_i$ s.t. $v_i \in \mathcal{D}Subterm(\mathcal{R}, s^i_j)$ and $\alpha_i \in Rules_C(\mathcal{R}, \ell_i)$.*

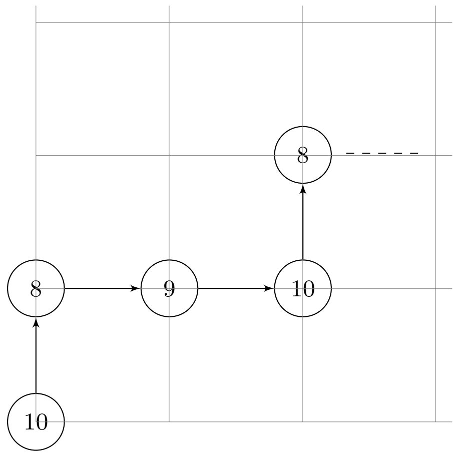

Fig. 2. Computations starting with $f(a)$ for $\mathcal{R}$ in Example 5

Example 5. Consider the following deterministic 3-CTRS $\mathcal{R}$ which is obtained from the 2-CTRS in Example 2 by a small change in rule (7) to yield (10):

$$g(a) \to c(b) \tag{8}$$

$$b \to f(a) \tag{9}$$

$$f(x) \to y \Leftarrow g(x) \to c(y) \tag{10}$$

Figure 2 shows the representation of computation starting from $f(a) \in \mathcal{T}_{\text{op-}\infty}$ according to Theorem 1, where the coordinates (x_i, y_i) have been left implicit.

Remark 2. The infinite minimal *rewrite sequence* $f(a) \to_{(10)} b \to_{(9)} f(a) \to_{(10)} b \to \cdots$ is also possible for $\mathcal{R}$ in Example 5. This is because $\sigma(g(x)) \to^* \sigma(c(y))$ for rule (10) is satisfied *without rewriting* b if $\sigma(x) = a$ and $\sigma(y) = b$. The implicit assumption in the computation model of Proposition 1 is that only reachability conditions $\sigma(s_i) \to^* \sigma(t_i)$ that are free of any infinite computation are important to decide the application of a rule. This makes real sense in practice. And, of course, it is harmless for the correctness and completeness of our approach.

According to our discussion, the following definition establishes the subsets of rules that play a special role in computations starting from minimal terms.

Definition 3. *The* dependent usable rules *for a CTRS $\mathcal{R}$ and $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ are:*

$$\mathcal{DU}(\mathcal{R}, t) = \mathit{DRules}(\mathcal{R}, t) \cup \bigcup_{(l \to r \Leftarrow c) \in \mathit{DRules}(\mathcal{R}, t)} \; \bigcup_{v \in \mathcal{D}\mathit{Subterm}(\mathcal{R}, r)} \mathcal{DU}(\mathcal{R}^{\bullet}, v)$$

where $\mathcal{R}^{\bullet} = \mathcal{R} - \mathit{DRules}(\mathcal{R}, t)$. The set of minimal usable rules *of $\mathcal{R}$ for t is:*

$$\mathcal{MU}(\mathcal{R}, t) = \mathit{URules}(\mathcal{R}, t) \cup \bigcup_{(l \to r \Leftarrow c) \in \mathit{DRules}(\mathcal{R}, t)} \; \bigcup_{v \in \mathcal{D}\mathit{Subterm}(\mathcal{R}, r)} \mathcal{MU}(\mathcal{R}^{\bullet}, v).$$

Let $\overline{\mathcal{MU}}(\mathcal{R}, t) = \emptyset$ if $\mathcal{MU}(\mathcal{R}, t)$ is a TRS and $\overline{\mathcal{MU}}(\mathcal{R}, t) = \mathcal{MU}(\mathcal{R}, t)$ otherwise.

Example 6. For $\mathcal{R}$ in Example 1, $\mathcal{DU}(\mathcal{R},h(x)) = \mathcal{MU}(\mathcal{R},h(x)) = \emptyset$; $\mathcal{DU}(\mathcal{R},g(x)) = \emptyset$ but $\mathcal{MU}(\mathcal{R},g(x)) = \overline{\mathcal{MU}}(\mathcal{R},g(x)) = \{(4)\}$, and $\mathcal{DU}(\mathcal{R},f(x,x,x)) = \mathcal{MU}(\mathcal{R},f(x,x,x)) = \{(3)\}$, but $\overline{\mathcal{MU}}(\mathcal{R},f(x,x,x)) = \emptyset$.

Example 7. For $\mathcal{R}$ in Example 5, $DRules(\mathcal{R},f(a)) = \emptyset$ (because the right-hand side y in rule (10) defining f is a variable), $\mathcal{DU}(\mathcal{R},g(x)) = \{(8),(9)\}$ and $\mathcal{MU}(\mathcal{R},g(x)) = \overline{\mathcal{MU}}(\mathcal{R},g(x)) = \mathcal{R}$.

The following result shows that an infinite computation starting from a minimal operationally nonterminating term can either start an infinite (*horizontal*) *rewrite* sequence (possibly as part of the evaluation of one of the conditions of a rule) or else climb infinitely many '*vertical*' steps over the conditions in the rules.

Corollary 1. *Let $\mathcal{R}$ be a deterministic 3-CTRS and $t \in \mathcal{T}_{op\text{-}\infty}$. Then, the sequence $\{(x_i,y_i,\alpha_i)\}_{i\geq 0}$ associated to t according to Theorem 1 satisfies one of the following conditions. Either*

1. *There is $k \geq 0$, $\ell_k \to r_k \Leftarrow c_k \in \mathcal{R}$, and an infinite* 'horizontal' *sequence $\{(x_i,y_k,\alpha_i)\}_{i\geq k}$ such that for all $i \geq k$, $x_{i+1} = x_i + 1$ and $\alpha_i \in \bigcup_{v_k \in \mathcal{D}Subterm(\mathcal{R},r_k)} \mathcal{DU}(\mathcal{R},v_k)$, or*
2. *For each $i \in \mathbb{N}$ such that $y_i > 0$ and $y_i = y_{i-1} + 1$, there is $k_i > i$ such that $y_{k_i} = y_i + 1$, and there is j_i, $1 \leq j_i \leq n_i$ such that $\alpha_{k_i-1} \in \bigcup_{v_i \in \mathcal{D}Subterm(\mathcal{R},s_{ij_i})} \overline{\mathcal{MU}}(\mathcal{R},v_i)$, with $n_{k_i-1} > 0$ conditions in the conditional part of the rule.*

In the following, we use *Dependency Pairs* to capture the nontermination behavior of computations with CTRSs.

4 2D Dependency Pairs for CTRSs

Given a signature $\mathcal{F}$ and $f \in \mathcal{F}$, we let $f^\sharp$ (often just capitalized, e.g., F) be a fresh symbol associated to f [1]. Let $\mathcal{F}^\sharp = \{f^\sharp \mid f \in \mathcal{F}\}$. For $t = f(t_1,\ldots,t_k) \in \mathcal{T}(\mathcal{F},\mathcal{X})$, we write $t^\sharp$ to denote the *marked* term $f^\sharp(t_1,\ldots,t_k)$. Our Dependency Pairs for CTRSs are organized into two blocks. The *horizontal* block contains those pairs that correspond to rules issuing root steps in infinite rewrite sequences (Proposition 1, item 1):

$$\mathsf{DP}_H(\mathcal{R}) = \{\ell^\sharp \to v^\sharp \Leftarrow c \mid \ell \to r \Leftarrow c \in R, r \trianglerighteq v, \ell \not\triangleright v, DRules(\mathcal{R},v) \neq \emptyset\}$$

Example 8. For $\mathcal{R}$ in Example 1, $\mathsf{DP}_H(\mathcal{R}) = \{F(k(a),k(b),x) \to F(x,x,x)\}$. For $\mathcal{R}$ in Example 5 (and also for $\mathcal{R}$ in Example 2), $\mathsf{DP}_H(\mathcal{R}) = \{G(a) \to B\}$.

The *vertical* block contains pairs for shifting the infinite computation to the conditions of the rules (Proposition 1, item 2):

$$\begin{aligned}\mathsf{DP}_V(\mathcal{R}) = \{\ell^\sharp \to v^\sharp \Leftarrow \bigwedge_{j=1}^{k-1} s_j \to t_j \mid \ell \to r \Leftarrow \bigwedge_{i=1}^{n} s_i \to t_i \in R,\\ \exists k, 1 \leq k \leq n, s_k \trianglerighteq v, \ell \not\triangleright v, URules(\mathcal{R},v) \neq \emptyset\}.\end{aligned}$$

Example 9. For $\mathcal{R}$ in Example 1, $\mathsf{DP}_V(\mathcal{R}) = \emptyset$. For $\mathcal{R}$ in Example 2 and $\mathcal{R}$ in Example 5), $\mathsf{DP}_V(\mathcal{R}) = \{F(x) \to G(x)\}$.

The subterms in the conditions of the rules that originate the pairs in $\mathsf{DP}_V(\mathcal{R})$ are collected in the following set, which we use below:

$$\mathsf{V}_C(\mathcal{R}) = \{v \mid \ell \to r \Leftarrow \bigwedge_{i=1}^{n} s_i \to t_i \in R, \exists k, 1 \leq k \leq n, s_k \trianglerighteq v, \ell \ntriangleright v, \mathit{URules}(\mathcal{R}, v) \neq \emptyset\}.$$

Example 10. For $\mathcal{R}$ in Example 1, $\mathsf{V}_C(\mathcal{R}) = \emptyset$. For $\mathcal{R}$ in Example 2 and $\mathcal{R}$ in Example 5, $\mathsf{V}_C(\mathcal{R}) = \{g(x)\}$.

We also have pairs to *connect* pairs in $\mathsf{DP}_V(\mathcal{R})$ (Corollary 1, item 2):

$$\mathsf{DP}_{VH}(\mathcal{R}) = \bigcup_{w \in \mathsf{V}_C(\mathcal{R})} \{\ell^\sharp \to v^\sharp \Leftarrow c \mid \ell \to r \Leftarrow c \in \overline{\mathcal{MU}}(\mathcal{R}, w), \\ r \trianglerighteq v, \ell \ntriangleright v, \mathit{URules}(\mathcal{R}, v) \neq \emptyset\}.$$

Example 11. For $\mathcal{R}$ in Example 1, $\mathsf{DP}_{VH}(\mathcal{R}) = \emptyset$. For $\mathcal{R}$ in Example 2 and $\mathcal{R}$ in Example 5, $\mathsf{DP}_{VH}(\mathcal{R}) = \{G(a) \to B, B \to F(a)\}$.

Here is the definition of $2D$-Dependency Pairs for a CTRS.

Definition 4 ($2D$-Dependency Pairs). *The triple of* 2D-dependency pairs *(2D DPs) for the CTRS $\mathcal{R}$ is* $\mathsf{DP}_{\mathsf{2D}}(\mathcal{R}) = (\mathsf{DP}_H(\mathcal{R}), \mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}))$.

Example 12. Consider the following 3-CTRS $\mathcal{R}$ in [14, Example 7.1.5]

$$\begin{aligned}
\mathsf{less}(x, \mathsf{0}) &\to \mathsf{false} && (11)\\
\mathsf{less}(\mathsf{0}, \mathsf{s}(x)) &\to \mathsf{true} && (12)\\
\mathsf{less}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{less}(x, y) && (13)\\
\mathsf{minus}(\mathsf{0}, \mathsf{s}(y)) &\to \mathsf{0} && (14)\\
\mathsf{minus}(x, \mathsf{0}) &\to x && (15)\\
\mathsf{minus}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{minus}(x, y) && (16)\\
\mathsf{quotrem}(\mathsf{0}, \mathsf{s}(y)) &\to \mathsf{pair}(\mathsf{0}, \mathsf{0}) && (17)\\
\mathsf{quotrem}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{pair}(\mathsf{0}, \mathsf{s}(x)) \Leftarrow \mathsf{less}(x, y) \to \mathsf{true} && (18)\\
\mathsf{quotrem}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{pair}(\mathsf{s}(q), r) && (19)\\
&\Leftarrow \mathsf{less}(x, y) \to \mathsf{false}, \mathsf{quotrem}(\mathsf{minus}(x, y), \mathsf{s}(y)) \to \mathsf{pair}(q, r)
\end{aligned}$$

The set $\mathsf{DP}_H(\mathcal{R})$ consists of the rules:

$$\begin{aligned}
\mathsf{LESS}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{LESS}(x, y) && (20)\\
\mathsf{MINUS}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{MINUS}(x, y) && (21)
\end{aligned}$$

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

$$\begin{aligned}
&\mathsf{QUOTREM}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{LESS}(x, y) && (22)\\
&\mathsf{QUOTREM}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{QUOTREM}(\mathsf{minus}(x, y), \mathsf{s}(y)) \Leftarrow \mathsf{less}(x, y) \to \mathsf{false} && (23)\\
&\mathsf{QUOTREM}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{MINUS}(x, y) \Leftarrow \mathsf{less}(x, y) \to \mathsf{false} && (24)
\end{aligned}$$

Finally, $\mathsf{DP}_{VH}(\mathcal{R}) = \emptyset$.

4.1 Characterizing operational termination of CTRSs using 2D DPs

An essential property of the dependency pair method is that it provides a *characterization* of termination of a TRS $\mathcal{R}$ as the absence of infinite (minimal) *chains of dependency pairs* [1, 8]. As we prove below, this is also true for deterministic 3-CTRSs when 2D DPs are considered. First, we have to introduce a suitable notion of chain that can be used with 2D DPs.

Definition 5 (Chain of pairs - Minimal chain). *Let $\mathcal{P}, \mathcal{Q}, \mathcal{R}$ be CTRSs. A $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-chain is a finite or infinite sequence of pairs $u_i \to v_i \Leftarrow \bigwedge_{j=1}^{n_i} s_{ij} \to t_{ij} \in \mathcal{P}$, together with a substitution σ satisfying that, for all $i \geq 1$,*

1. *$\sigma(s_{ij}) \to^*_{\mathcal{R}} \sigma(t_{ij})$ for all j, $1 \leq j \leq n_i$ and*
2. *$\sigma(v_i)(\to^*_{\mathcal{R}} \circ \xrightarrow{\Lambda}{}^{=}_{\mathcal{Q}})^*\sigma(u_{i+1})$, where given a rule $\ell \to r \Leftarrow \bigwedge_{j=1}^{n} s_j \to t_j \in \mathcal{Q}$, we write $s \xrightarrow{\Lambda}{}^{=}_{\mathcal{Q}} t$ if either $s = t$ or there is a substitution θ such that $s = \theta(\ell)$, $t = \theta(r)$ and $\theta(s_i) \to^*_{\mathcal{R}} \theta(t_i)$ for all j, $1 \leq j \leq n$ (note that the satisfaction of reachability constraints involves rewritings with $\mathcal{R}$).*

As usual, we assume that different occurrences of pairs do not share any variable (renaming substitutions are used if necessary). A $(\mathcal{P}, \mathcal{Q}, \mathcal{R})$-chain is called minimal *if for all $i \geq 1$, $\sigma(v_i)$ is $\mathcal{R}$-operationally terminating.*

Remark 3. Note that, if $\mathcal{P}$ and $\mathcal{R}$ are TRSs (without conditional rules) and $\mathcal{Q} = \emptyset$, Definition 5 specializes to the standard definition of chain of pairs in the Dependency Pair Framework for TRSs [8, Definition 3].

We now provide a new characterization of operational termination of CTRSs.

Theorem 2 (Operational termination of CTRSs). *A deterministic* 3-*CTRS $\mathcal{R}$ is operationally terminating if and only if there is no infinite (minimal) $(\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R})$-chain and there is no infinite (minimal) $(\mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}), \mathcal{R})$-chain.*

Example 13. Consider again the $\mathcal{R}$ in Examples 2 and 5 and $\mathsf{DP}_V(\mathcal{R})$ and $\mathsf{DP}_{VH}(\mathcal{R})$ (that coincide for both CTRSs) as given in Examples 9 and 11. There is an infinite $(\mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}), \mathcal{R})$-chain:

$$B \to_{\mathsf{DP}_{VH}(\mathcal{R})} F(a) \to^*_{\mathcal{R}} F(a) \to_{\mathsf{DP}_V(\mathcal{R})} G(a) \to^*_{\mathcal{R}} G(a) \to_{\mathsf{DP}_{VH}(\mathcal{R})} B$$

witnessing that *both* CTRSs are *not* operationally terminating.

For the sake of brevity, in the following we often call H-chains to the $(\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R})$-chains. And we call V-chains to the $(\mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}), \mathcal{R})$-chains. The following result, involving chains of a simpler type (closer to the usual ones, where pairs are connected by rewritings with $\mathcal{R}$ only, see Remark 3), also characterizes operational termination of deterministic 3-CTRSs.

Theorem 3 (Operational termination of CTRSs II). *A deterministic* 3-*CTRS $\mathcal{R}$ is operationally terminating if and only if there is no infinite (minimal) $(\mathsf{DP}_H(\mathcal{R}) \cup \mathsf{DP}_V(\mathcal{R}) \cup \mathsf{DP}_{VH}(\mathcal{R}), \emptyset, \mathcal{R})$-chain.*

In the following section we further motivate the explicit and *independent* use of the H-chains and V-chains to prove termination properties of CTRSs.

4.2 Termination of 2-CTRSs

The existence of infinite H-chains witnesses *nontermination* of deterministic 3-CTRSs, i.e., the absence of infinite *rewrite sequences.*

Theorem 4 (Non-termination of CTRSs). *Let $\mathcal{R}$ be a deterministic 3-CTRS. If there is an infinite $(\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R})$-chain, then $\mathcal{R}$ is not terminating.*

The CTRS $\mathcal{R}$ in Example 5 shows that Theorem 4 provides a sufficient but *not necessary* criterion for termination of CTRSs.

Example 14. For $\mathcal{R}$ in Example 5, we have $\mathsf{DP}_H(\mathcal{R}) = \{G(a) \to B\}$ (Example 8). There is no infinite H-chain. However, $\mathcal{R}$ is *not* terminating (see Remark 2).

However, the following result holds:

Theorem 5 (Termination of 2-CTRSs). *A 2-CTRS $\mathcal{R}$ is terminating if and only if there is no infinite minimal $(\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R})$-chain.*

Example 15. For the deterministic 2-CTRS $\mathcal{R}$ in Example 2, $\mathsf{DP}_H(\mathcal{R}) = \{G(a) \to B\}$ and there is no infinite H-chain. By Theorem 5, $\mathcal{R}$ is *terminating.*

Therefore, for CTRSs with extra variables in the right-hand sides of conditional rules, the vertical and horizontal dimensions of operational termination are *not* completely independent. Theorem 5 suggests the following.

Definition 6 (V-termination of CTRSs). *A CTRS $\mathcal{R}$ is V-terminating if there is no infinite $(\mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}), \mathcal{R})$-chain.*

As a consequence of Theorems 2 and 5, we have the following.

Corollary 2. *A deterministic 2-CTRS is operationally terminating if and only if it is terminating and V-terminating.*

5 Mechanizing proofs of operational termination with 2D DPs

In the following, we speak of $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, (\mathsf{ctrs}, \gamma))$-chains, for $\gamma = \mathsf{a}$ (or $\gamma = \mathsf{m}$) if arbitrary (resp. only minimal) chains are considered. Similarly, according to Remark 3, we speak of $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, (\mathsf{trs}, \gamma))$-chains if $\mathcal{P}$ and $\mathcal{R}$ are TRSs and $\mathcal{Q} = \emptyset$.

Definition 7 (CTRS problem). *A* CTRS problem *τ is a tuple $\tau = (\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$, where $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ are CTRSs, and $e \in \{\mathsf{ctrs}, \mathsf{trs}\} \times \{\mathsf{a}, \mathsf{m}\}$ is a flag. The CTRS problem τ is finite if there is no infinite minimal $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$-chain. The CTRS problem τ is infinite if $\mathcal{R}$ is non-operationally terminating or there is an infinite minimal $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$-chain.*

Definition 8 (CTRS processor). *A* CTRS processor P *is a mapping from CTRS problems into sets of CTRS problems. Alternatively, it can also return* "no"*. A CTRS processor* P *is*

– sound *if for all CTRS problems* τ*, we have that* τ *is finite whenever* $\mathsf{P}(\tau) \neq \mathsf{no}$ *and all CTRS problems in* $\mathsf{P}(\tau)$ *are finite.*
– complete *if for all CTRS problems* τ*, we have that* τ *is infinite whenever* $\mathsf{P}(\tau) = \mathsf{no}$ *or when* $\mathsf{P}(\tau)$ *contains an infinite CTRS problem.*

A (sound) processor transforms CTRS problems into (hopefully) *simpler* ones, in such a way that the existence of an infinite chain in the original CTRS problem implies the existence of an infinite chain in the transformed one. Here, 'simpler' usually means that fewer pairs are involved. Soundness is essential for proving *operational termination*; completeness for proving *non-operational termination.*

Processors are used in a *divide and conquer* scheme to incrementally simplify the original CTRS problem as much as possible, possibly decomposing it into (a tree of) smaller pieces which are independently treated in the same way. The trivial case comes when the set of pairs $\mathcal{P}$ becomes empty. Then, no infinite chain is possible, and the CTRS problem is finite. Such *positive* answer is propagated upwards in the decision tree. In some cases, a witness of an infinite chain is obtained; then a *negative* answer "no" can be provided and propagated upwards.

Theorem 6 (2D DP framework). *Let* $\mathcal{R}$ *be a deterministic 3-CTRS. We construct* two *trees whose nodes are labeled with CTRS problems* τ *or* "yes" *or* "no"*. The roots are* $\tau_H = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R}, (\mathsf{ctrs}, \gamma))$ *and* $\tau_V = (\mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}), \mathcal{R}, (\mathsf{ctrs}, \gamma))$*, respectively (for* $\gamma \in \{\mathsf{a}, \mathsf{m}\}$*). For every node which is a CTRS problem* τ*, there is a processor* P *satisfying one of the following conditions:*

1. $\mathsf{P}(\tau) = \mathsf{no}$ *and the node has just one child that is labeled with* "no"*.*
2. $\mathsf{P}(\tau) = \emptyset$ *and the node has just one child that is labeled with* "yes"*.*
3. $\mathsf{P}(\tau) \neq \mathsf{no}$, $\mathsf{P}(\tau) \neq \emptyset$*, and the children of the node are labeled with the CTRS problems in* $\mathsf{P}(\tau)$*.*

If all leaves of both trees are labeled with "yes" *and all used processors are* sound*, then* $\mathcal{R}$ *is operationally terminating. If a leaf is labeled with* "no" *in some of the trees and all processors used on the path from the root to this leaf are* complete*, then* $\mathcal{R}$ *is operationally nonterminating.*

Remark 4. By Theorem 3, an alternative to the twofold proof starting from an H-problem and a V-problem is to start the proof of operational termination of $\mathcal{R}$ from a *single* CTRS problem $(\mathsf{DP}_H(\mathcal{R}) \cup \mathsf{DP}_V(\mathcal{R}) \cup \mathsf{DP}_{VH}(\mathcal{R}), \emptyset, \mathcal{R}, (\mathsf{ctrs}, \gamma))$.

Remark 5. In order to prove (or disprove) *termination* of a deterministic CTRS $\mathcal{R}$, we would use Theorem 6 with a *single* problem: $\tau_H = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R}, e)$. The procedure is analogous and the conclusion of a positive analysis (i.e., "yes" in all leaves of the tree) is *termination* of $\mathcal{R}$ (if it is a 2-CTRS). Similarly, a leaf labeled with "no" witnesses nontermination of $\mathcal{R}$ (if it is a 3-CTRS).

6 Processors for the 2D DP Framework

The first processor *moves* rules from $\mathcal{Q}$ to $\mathcal{P}$ in CTRS problems.

Theorem 7 (Moving $\mathcal{Q}$-rules). *Let $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ be TRSs. Then,*

$$\mathsf{P}_{\mathsf{Q2P}}(\mathcal{P},\mathcal{Q},\mathcal{R},(\mathsf{ctrs},\gamma)) = \{(\mathcal{P}\cup\mathcal{Q},\emptyset,\mathcal{R},(\mathsf{ctrs},\mathsf{a}))\}$$

is a sound processor.

In general, $\mathsf{P}_{\mathsf{Q2P}}$ is not complete nor preserves minimality.

Example 16. Let $\mathcal{P} = \{a \to b, c \to a\}$, $\mathcal{Q} = \{b \to c\}$, and $\mathcal{R} = \{c \to c\}$. There is an infinite $(\mathcal{P},\mathcal{Q},\mathcal{R})$-chain Γ: $a \to b, c \to a, a \to b, c \to a, \ldots$ due to $b \to_{\mathcal{Q}} c$. Note that Γ is minimal because b and a are $\mathcal{R}$-terminating. However, Γ requires the use of the (only) pair in $\mathcal{Q}$ to become an infinite $(\mathcal{P}\cup\mathcal{Q},\emptyset,\mathcal{R})$-chain

$$a \to b, b \to c, c \to a, a \to b, b \to c, c \to a, \ldots$$

which is, however, *not* minimal now because c is *not* $\mathcal{R}$-terminating.

The following processor *transfers* any proof of finiteness of 2D DP problems to the DP Framework for TRSs. In this way, all existing processors for the DP Framework are now available for the 2D DP framework.

Theorem 8 (Shift to DP-Framework). *Let $\mathcal{P}$ and $\mathcal{R}$ be TRSs. Then,*

$$\mathsf{P}_{TRS}(\mathcal{P},\emptyset,\mathcal{R},(\mathsf{ctrs},\gamma)) = \{(\mathcal{P},\emptyset,\mathcal{R},(\mathsf{trs},\gamma))\}$$

is a sound and complete processor.

6.1 Graph of a CTRS problem

Given a CTRS problem $(\mathcal{P},\mathcal{Q},\mathcal{R},e)$, we provide a notion of graph that is able to represent all infinite (*minimal*) chains of pairs as given in Definition 5.

Definition 9 (Graph of a CTRS problem). *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be CTRSs. The* CTRS-graph $\mathsf{G}(\mathcal{P},\mathcal{Q},\mathcal{R},e)$ *where $e = (\mathsf{ctrs},\gamma)$ and $\gamma \in \{\mathsf{a},\mathsf{m}\}$ has $\mathcal{P}$ as the set of nodes. Given $\alpha : u \to v \Leftarrow c, \alpha' : u' \to v' \Leftarrow c' \in \mathcal{P}$, there is an arc from α to α' if α, α' is a minimal $(\mathcal{P},\mathcal{Q},\mathcal{R},e)$-chain for some substitution σ.*

In general, CTRS graphs are *not* computable due to the reachability conditions $\sigma(v)(\to^*_{\mathcal{R}} \circ \xrightarrow{\Lambda}{}^{=}_{\mathcal{Q}})^*\sigma(u')$ (for $u \to v \Leftarrow c \in \mathcal{P}$). Since the reachability problem for (conditional) rewriting is undecidable, we *approximate* it. Following [9], we approximate the CTRS-dependency graph as follows. Let $\textsc{tcap}_{\mathcal{R}}$ be:

$$\textsc{tcap}_{\mathcal{R}}(x) = y \quad \text{if } x \text{ is a variable, and}$$
$$\textsc{tcap}_{\mathcal{R}}(f(t_1,\ldots,t_k)) = \begin{cases} f([t_1],\ldots,[t_k]) & \text{if } f([t_1],\ldots,[t_k]) \text{ does not unify} \\ & \text{with } \ell \text{ for any } \ell \to r \Leftarrow c \text{ in } \mathcal{R} \\ y & \text{otherwise} \end{cases}$$

where y is a new, fresh variable that has not yet been used, and given a term s, $[s] = \textsc{tcap}_{\mathcal{R}}(s)$. We assume that ℓ shares no variable with $f([t_1],\ldots,[t_k])$ (rename if necessary). With $\textsc{tcap}_{\mathcal{R}}$ we approximate reachability problems as *unification*. According to Definitions 5 and 9, we have the following.

Definition 10 (Estimated connection). *Let $\mathcal{Q}$ and $\mathcal{R}$ be CTRSs, θ be a substitution, and $\alpha : u \to v \Leftarrow c$ to $\alpha' : u' \to v' \Leftarrow c'$ be two conditional rules. There is a $(\mathcal{Q}, \mathcal{R}, \theta)$-*connection *from α to α' if*

1. *$\textsc{tcap}_{\mathcal{R}}(\theta(v))$ and u' unify, or*
2. *$\textsc{tcap}_{\mathcal{R}}(\theta(v))$ and u'' unify with* mgu *θ' for some $\alpha'' : u'' \to v'' \Leftarrow c'' \in \mathcal{Q}$ and there is a $(\mathcal{Q} - \{\alpha''\}, \mathcal{R}, \theta')$-connection from α'' to α'.*

Definition 11 (Estimated Graph). *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be CTRSs. The* estimated *CTRS-graph $\mathsf{EG}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$ has $\mathcal{P}$ as the set of nodes. There is an arc from α to α' if there is a $(\mathcal{Q}, \mathcal{R}, \epsilon)$-connection from α to α'.*

Remark 6. If $\mathcal{Q} = \emptyset$ and $\mathcal{P}, \mathcal{R}$ are TRSs, Definitions 9 and 11 specialize to the standard ones for TRSs [8, Definition 7] (and [9, Definition 12]).

The following processor decomposes a CTRS problem $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$ with graph $\mathsf{G}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$ according to the *strongly connected components* (SCCs) of the graph, i.e., cycles in $\mathsf{G}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$ that are not contained in any other cycle.

Theorem 9 (SCC processor). *Let $\mathcal{P}$, $\mathcal{Q}$ and $\mathcal{R}$ be CTRSs. Then,*

$$\mathsf{P}_{SCC}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e) = \{(\mathcal{P}', \mathcal{Q}, \mathcal{R}, e) \mid \mathcal{P}' \subseteq \mathcal{P} \textit{ is an SCC in } \mathsf{G}(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)\}$$

is a sound and complete processor.

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

Example 17. For $\mathcal{R}$ in Example 12, $\tau_H = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R}, e)$ and $\tau_V = (\mathsf{DP}_V(\mathcal{R}), \mathsf{DP}_{VH}(\mathcal{R}), \mathcal{R}, e)$; $\mathsf{EG}(\tau_H)$ and $\mathsf{EG}(\tau_V)$ are:

$$\begin{tikzcd}
\boxed{\mathsf{EG}(\tau_H):} & 20 \arrow[loop above] & 21 \arrow[loop above] & \boxed{\mathsf{EG}(\tau_V):} & 22 & 23 \arrow[l] \arrow[r] \arrow[loop above] & 24
\end{tikzcd}$$

We have $\mathsf{P}_{SCC}(\tau_H) = \{\tau_{H1}, \tau_{H2}\}$, where $\tau_{H1} = (\{(20)\}, \emptyset, \mathcal{R}, e)$ and $\tau_{H2} = (\{(21)\}, \emptyset, \mathcal{R}, e)$. For τ_V we get $\mathsf{P}_{SCC}(\tau_V) = \{\tau_{V1}\}$, where $\tau_{V1} = (\{(23)\}, \emptyset, \mathcal{R}, e)$.

6.2 Use of orderings and argument filterings

A CTRS problem $(\mathcal{P}, \mathcal{Q}, \mathcal{R}, e)$ can be *simplified* by *removing* rules with a *decrease* with respect to a well-founded relation $\sqsupset$. In order to be more precise, in the following we say that a relation S is *compatible* with R if $S \circ R \subseteq R$ or $R \circ S \subseteq R$.

Definition 12 (Removal triple). *A* removal triple $(\gtrsim, \succeq, \sqsupset)$ *consists of relations $\gtrsim, \succeq, \sqsupset$ on terms such that $\sqsupset$ is well-founded; for all $R \in \{\gtrsim, \succeq\}$, R is compatible with $\sqsupset$; and $\gtrsim \circ \succeq \subseteq \gtrsim$ or $\gtrsim \circ \succeq \subseteq \succeq$.*

An *argument filtering* π for a signature $\mathcal{F}$ is a mapping that assigns to each k-ary function symbol $f \in \mathcal{F}$ an argument position $i \in \{1, \ldots, k\}$ or a (possibly empty) list $[i_1, \ldots, i_m]$ of argument positions $1 \leq i_1 < \cdots < i_m \leq k$ [11]. The *trivial* argument filtering $\pi_\top(f) = [1, \ldots, k]$ (for each k-ary symbol $f \in \mathcal{F}$) does nothing. The signature $\mathcal{F}_\pi$ of symbols with filtered arguments consists of all function symbols $f \in \mathcal{F}$ such that $\pi(f) = [i_1, \ldots, i_m]$; the arity of f in $\mathcal{F}_\pi$ is m but we do not change the name. An argument filtering π induces a mapping from $\mathcal{T}(\mathcal{F}, \mathcal{X})$ to $\mathcal{T}(\mathcal{F}_\pi, \mathcal{X})$, also denoted by π, which removes subterms:

$$\pi(t) = \begin{cases} t & \text{if } t \text{ is a variable} \\ \pi(t_i) & \text{if } t = f(t_1, \ldots, t_k) \text{ and } \pi(f) = i \\ f(\pi(t_{i_1}), \ldots, \pi(t_{i_m})) & \text{if } t = f(t_1, \ldots, t_k) \text{ and } \pi(f) = [i_1, \ldots, i_m] \end{cases}$$

And if R is a relation on terms, we let $\pi(R) = \{(\pi(s), \pi(t)) \mid (s, t) \in R\}$. Argument filterings provide a simple way to *remove* parts of the syntactic structure of a rule. In this way, we obtain simpler rules that are easier to compare. In the following, given (possibly empty) set of rules $\mathcal{R}, \mathcal{S}$ and a rule $\alpha : \ell \to r \Leftarrow c$, we define the (possible) *replacement* of α in $\mathcal{R}$ by the rules $\mathcal{S}$ as follows:

$$\mathcal{R}[\mathcal{S}]_\alpha = \begin{cases} (\mathcal{R} - \{\alpha\}) \cup \mathcal{S} & \text{if } \alpha \in \mathcal{R} \\ \mathcal{R} & \text{otherwise} \end{cases}$$

Theorem 10 (Removal triple processor). *Let $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ be CTRSs, π be an argument filtering and $(\gtrsim, \succeq, \sqsupset)$ be a removal triple such that $\pi(\to^*_{\mathcal{R}}) \subseteq \gtrsim$ and for all $\ell \to r \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$ and substitutions σ, if for all $s \to t \in c$, $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ holds, then $\pi(\sigma(\ell)) \bowtie \pi(\sigma(r))$ holds for some $\bowtie \in \{\gtrsim, \succeq, \sqsupset\}$. Let $\alpha : u \to v \Leftarrow c \in \mathcal{P} \cup \mathcal{Q}$ be such that, for all substitutions σ, if for all $s \to t \in c$, $\sigma(s) \to^*_{\mathcal{R}} \sigma(t)$ holds, then $\pi(\sigma(u)) \sqsupset \pi(\sigma(v))$ holds. Then,*

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

is a sound and complete processor.

Example 18. For τ_{H1}, τ_{H2} and τ_{V1} in Example 17, we apply P_{RT} to those problems with $\pi_\top$ (which we do not make explicit here, as it does nothing) and using the same removal triple $(\geq, \geq, >)$ induced by the polynomial interpretation

$$\begin{array}{rrrr} [\mathsf{false}] = 0 & [\mathsf{true}] = 0 & [\mathsf{0}] = 0 & [\mathsf{s}](x) = x + 1 \\ [\mathsf{less}](x) = 0 & [\mathsf{minus}](x, y) = x & [\mathsf{pair}](x, y) = 0 & [\mathsf{quotrem}](x, y) = 0 \\ [\mathsf{LESS}](x, y) = x & [\mathsf{MINUS}](x, y) = x & [\mathsf{QUOTREM}](x, y) = x & \end{array}$$

over the naturals $\mathbb{N}$ by $s \geq t$ if $[s] \geq [t]$ and $s > t$ if $[s] > [t]$. We have:

$$\begin{array}{rcl} [\mathsf{less}(x, \mathsf{0})] = & 0 & \geq 0 = [\mathsf{false}] \\ [\mathsf{less}(\mathsf{0}, \mathsf{s}(x))] = & 0 & \geq 0 = [\mathsf{true}] \\ [\mathsf{less}(\mathsf{s}(x), \mathsf{s}(y))] = & 0 & \geq 0 = [\mathsf{less}(x, y)] \\ [\mathsf{minus}(\mathsf{0}, \mathsf{s}(y))] = & 0 & \geq 0 = [\mathsf{0}] \\ [\mathsf{minus}(x, \mathsf{0})] = & x & \geq x = [x] \\ [\mathsf{minus}(\mathsf{s}(x), \mathsf{s}(y))] = & x + 1 & \geq x = [\mathsf{minus}(x, y)] \\ [\mathsf{quotrem}(\mathsf{0}, \mathsf{s}(y))] = & 0 & \geq 0 = [\mathsf{pair}(\mathsf{0}, \mathsf{0})] \\ [\mathsf{quotrem}(\mathsf{s}(x), \mathsf{s}(y))] = & 0 & \geq 0 = [\mathsf{pair}(\mathsf{0}, \mathsf{s}(x))] \\ [\mathsf{quotrem}(\mathsf{s}(x), \mathsf{s}(y))] = & 0 & \geq 0 = [\mathsf{pair}(\mathsf{s}(q), r)] \end{array}$$

$$\begin{aligned}[\mathsf{LESS}(\mathsf{s}(x),\mathsf{s}(y))] &= x+1 > x = [\mathsf{LESS}(x,y)]\\ [\mathsf{MINUS}(\mathsf{s}(x),\mathsf{s}(y))] &= x+1 > x = [\mathsf{MINUS}(x,y)]\\ [\mathsf{QUOTREM}(\mathsf{s}(x),\mathsf{s}(y))] &= x+1 > x = [\mathsf{QUOTREM}(\mathsf{minus}(x,y),\mathsf{s}(y))]\end{aligned}$$

Since $\geq$ is monotonic, stable, reflexive and transitive, the first nine inequalities prove $\to_{\mathcal{R}}^{*} \subseteq \geq$ (we do not really need to pay attention to the conditional part of the rules). Similarly, since $\sqsupset$ is stable, the last three strict inequalities prove $\sigma(u) \sqsupset \sigma(v)$ for all $u \to v \Leftarrow c \in \mathcal{P}$ (in the corresponding CTRS problem) and substitution σ, again without paying attention to the conditional part of the rules. This proves τ_{H1}, τ_{H2} and τ_{V1} finite, and $\mathcal{R}$ operationally terminating.

Theorem 11 (Unsatisfiable rules). *Let $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ be CTRSs, π be an argument filtering, $\gtrsim$ and $\sqsupset$ be relations on terms such that $\gtrsim$ is compatible with $\sqsupset$, $\pi(\to_{\mathcal{R}}^{*}) \subseteq \gtrsim$, and $\sqsupset$ is well-founded. Let $\alpha : \ell \to r \Leftarrow c \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{R}$ and $s_i \to t_i \in c$ be such that for all substitutions σ, $\pi(\sigma(t_i)) \sqsupset \pi(\sigma(s_i))$ holds. Then,*

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

is a sound and (if $\alpha \notin \mathcal{R}$ or $e = (\rho, \mathsf{a})$) complete processor.

Example 19. For $\mathcal{R}$ in Example 1, $\mathsf{DP}_H(\mathcal{R}) = \{F(k(a),k(b),x) \to F(x,x,x)\}$, and $\mathsf{DP}_V(\mathcal{R}) = \mathsf{DP}_{VH}(\mathcal{R}) = \emptyset$. For $\tau_H = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R}, (\mathsf{ctrs},\mathsf{m}))$, we use

$$[a] = [b] = [c](x) = [g](x) = [h](x) = [k](x) = [f](x,y,z) = 0 \quad \text{and} \quad [d] = 1$$

to generate $\geq$ and easily show (as in Example 18) that $\to_{\mathcal{R}}^{*} \subseteq \geq$. Since $[h(x)] = 0$ and $[d] = 1$, we have $[d] > [h(x)]$. With P_{UR}, we remove (4) from $\mathcal{R}$ to obtain $\tau_{H1} = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R} - \{(4)\}, (\mathsf{ctrs},\mathsf{m}))$ that satisfies the conditions for a shift with P_{TRS} to a DP problem $\tau_{\mathsf{trs}} = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R} - \{(4)\}, (\mathsf{trs},\mathsf{m}))$ that can then be solved by using any processor for TRSs. For instance, the forward instantiation processor [8, Definition 28] can be used to prove finiteness of τ_{trs}.

Theorem 12 (Unsatisfiable rules II). *Let $\mathcal{P}$, $\mathcal{Q}$, and $\mathcal{R}$ be CTRSs, π be an argument filtering, $\gtrsim$ and $\sqsupset$ be relations on terms such that $\gtrsim$ is compatible with $\sqsupset$, $\sqsupset$ is well-founded, and $\pi(\to_{\mathcal{R}}) \subseteq \sqsupset$. Let $\alpha : \ell \to r \Leftarrow c \in \mathcal{P} \cup \mathcal{Q} \cup \mathcal{R}$ and $s_i \to t_i \in c$ be such that $\pi(s_i)$ and $\pi(t_i)$ do not unify and for all substitutions σ, $\pi(\sigma(t_i)) \gtrsim \pi(\sigma(s_i))$ holds. Then,*

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

is a sound and (if $\alpha \notin \mathcal{R}$ or $e = (\rho, \mathsf{a})$) complete processor.

Example 20. Consider the following CTRS $\mathcal{R}$ [6, page 46]:

$$a \to b \qquad f(a) \to b \qquad g(x) \to g(a) \Leftarrow f(x) \to x$$

$\mathsf{DP}_H(\mathcal{R})$ consists of a single rule: $G(x) \to G(a) \Leftarrow f(x) \to x$ and $\mathsf{DP}_V(\mathcal{R}) = \mathsf{DP}_{VH}(\mathcal{R}) = \emptyset$. We use the relations $\geq$ and $>$ generated by

$$[a] = 1 \qquad [b] = 0 \qquad [f](x) = [g](x) = [G(x)] = x$$

(over $\mathbb{N}$), and P_{UR} to remove the rule in $\mathsf{DP}_H(\mathcal{R})$ from $\tau_H = (\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R}, e)$, thus proving operational termination of $\mathcal{R}$. Note that $>$ is monotonic, stable, transitive, and well-founded, and we have $\to_{\mathcal{R}} \subseteq >$ (and also $\to^+_{\mathcal{R}} \subseteq >$); the crucial point is that no substitution σ satisfies $\sigma(f(x)) \to^*_{\mathcal{R}} \sigma(x)$ for the conditional rules: since $f(x)$ and x do not unify, we should have $\sigma(f(x)) \to^+_{\mathcal{R}} \sigma(x)$ and hence $\sigma(f(x)) > \sigma(x)$. But $[\sigma(f(x))] = \sigma(x) \not> \sigma(x) = [\sigma(x)]$. Thus, we do not need to ensure that $[\sigma(g(x))] > [\sigma(g(a))]$ holds! However, $[x] = x \geq x = [f(x)]$.

7 Related work and conclusions

To the best of our knowledge, this is the first correct and complete characterization of operational termination of deterministic 3-CTRS which is based on the notion of dependency pair. The notion of minimal operationally nonterminating term and the properties explored here (Section 3) are also new in the literature. Our *bidimensional* approach simplifies the analysis of operational termination and is also useful to prove other properties like nontermination of 3-CTRSs and termination of 2-CTRSs. The analysis of termination of 2-CTRSs can also be accomplished as termination of the *underlying TRS* (i.e., the TRS $\mathcal{R}_u$ which is obtained by just dropping the conditional part of the rules). However, in contrast to our Theorem 5, the analysis of termination of 2-CTRSs $\mathcal{R}$ as termination of the underlying TRS $\mathcal{R}_u$ provides a *sufficient* condition only; it may fail in those cases where taking into account the conditions of the rules is essential to prove termination. For instance the one rule 2-CTRS $a \to a \Leftarrow a \to b$ is terminating but $\mathcal{R}_u$ is *not*. We prove $(\mathsf{DP}_H(\mathcal{R}), \emptyset, \mathcal{R}, e) = (\{A \to A \Leftarrow a \to b\}, \emptyset, \mathcal{R}, e)$ finite (and hence $\mathcal{R}$ terminating) using P_{UR} with the removal triple $(\geq, \geq, >)$ generated by the interpretation $[a] = 0$ and $[b] = 1$ to remove $A \to A \Leftarrow a \to b$.

The recent *Conditional Dependency Pairs* (CDPs) by Nakamura et al. [13] apply to a subclass of *1-CTRSs* where the conditions c in 1-rules ($\ell \to r \Leftarrow c$) are *terms* instead of sequences $s_1 \to t_1, \ldots, s_n \to t_n$. An instance $\sigma(c)$ of c is satisfied if and only if $\sigma(c) \to^* \mathsf{true}$. We generate a (usually strict) *subset* of the pairs considered in [13, Definition 3.1]: $\mathsf{DP}_H(\mathcal{R}) \cup \mathsf{DP}_V(\mathcal{R}) \cup \mathsf{DP}_{VH}(\mathcal{R}) \subseteq CDP(\mathcal{R})$. Their chains [13, Definition 3.2] are also different to ours (Definition 5).

As remarked in the introduction, existing tools for proving termination of conditional TRSs currently use transformation techniques. We are not aware of any implementation of *direct* methods. The transformation which is typically used for this purpose is $\mathcal{U}$ in [14, Definition 7.2.48]. This transformation is not complete, however. For instance, $\mathcal{U}(\mathcal{R})$ is not terminating for $\mathcal{R}$ in Examples 1 and 20, but we proved them operationally terminating in Examples 19 and 20. Furthermore, when $\mathcal{U}(\mathcal{R})$ is terminating, tools may fail to find a proof. This is often due to the loss of information introduced by transformations, and also to the presence of new symbols and rules that prevent the search process from finding a proof. The techniques presented in this paper have been incorporated in the latest version of the tool MU-TERM [2]. The first benchmarks of existing examples in the literature are very positive and show that the 2D DP framework permits simple and fast proofs like the ones in the examples of this paper. This makes these techniques available to tools like MTT [4], which 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. Direct termination methods for these wider logics will require extending the techniques presented here to the case of order-sorted conditional rewrite theories with types and subtypes, and where rewriting is context-sensitive and can take place modulo axioms B. This is envisaged as an interesting subject for future work.

Acknoledgements. We thank Raúl Gutiérrez for implementing the 2D DP Framework in MU-TERM. We also thank Francisco Durán for his helpful comments.

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. 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.
4. F. Durán, S. Lucas, and J. Meseguer. MTT: The Maude Termination Tool (System Description). *Proc. of IJCAR'08.* LNAI 5195:313–319, Springer-Verlag, 2008.
5. K. Futatsugi and R. Diaconescu. *CafeOBJ Report.* World Scientific, AMAST Series, 1998.
6. J. Giesl and T. Arts. Verification of Erlang Processes by Dependency Pairs. *Applicable Algebra in Engineering, Communication and Computing* 12:39-72, 2001.
7. 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, Springer-Verlag, 2004.
8. J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Mechanizing and Improving Dependency Pairs. *Journal of Automatic Reasoning*, 37(3):155–203, 2006.
9. J. Giesl, R. Thiemann, and P. Schneider-Kamp. Proving and Disproving Termination of Higher-Order Functions. In *Proc. of FroCoS'05*, LNAI 3717:216-231, Springer-Verlag, 2005.
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, Springer-Verlag, 2006.
11. K. Kusakari, M. Nakamura, and Y. Toyama. Argument Filtering Transformation. In *Proc. of PPDP'99*, LNCS 1702:47–61, Springer-Verlag, 1999.
12. S. Lucas, C. Marché, and J. Meseguer. Operational termination of conditional term rewriting systems. *Information Processing Letters*, 95:446–453, 2005.
13. M. Nakamura, K. Ogata, and K. Futatsugi. On Proving Operational Termination Incrementally with Modular Conditional Dependency Pairs. International Journal of Computer Science, 40:2, 2013.
14. E. Ohlebusch. *Advanced Topics in Term Rewriting.* Springer-Verlag, Apr. 2002.
15. 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.
16. F. Schernhammer and B. Gramlich. VMTL - A Modular Termination Laboratory. In *Proc. of RTA'09*, LNCS 5595:285–294, Springer-Verlag, 2009.