# MU-TERM: Verify Termination Properties Automatically (System Description)*

Raúl Gutiérrez[0000−0002−3984−2868] and Salvador Lucas[0000−0001−9923−2108]

Valencian Research Institute for Artificial Intelligence
Universitat Politècnica de València
Camino de Vera s/n, E-46022 Valencia, Spain
{rgutierrez,slucas}@dsic.upv.es

**Abstract.** We report on the new version of MU-TERM, a tool for proving termination properties of variants of rewrite systems, including conditional, context-sensitive, equational, and order-sorted rewrite systems. We follow a unified, logic-based approach to describe rewriting computations. The automatic generation of *logical models* for suitable first-order theories and formulas provides a common basis to implement the proofs.

## 1 Introduction

MU-TERM is a tool that can be used to automatically verify termination properties of variants of *Term Rewriting Systems* (TRSs): termination and innermost termination of TRSs using the DP Framework for TRSs [10] (this framework is also used to prove termination of *String Rewriting Systems*); termination and innermost termination of *context-sensitive rewriting* [16, 17] using the *Context-Sensitive DP Framework* [2, 13]; termination of *rewriting modulo associative/commutative theories* using the *A∨C-DP Framework* [4]; termination of *order-sorted rewriting* using the *Order-Sorted DP Framework* [22]; and operational termination of Conditional TRSs (CTRSs) using the *2D DP Framework* [24, 25]. In this setting, describing different kinds of rewriting computations as *proofs* of goals $s \to t$ and $s \to^* t$ with respect to an appropriate inference system is useful. Such an approach, exploiting the logic-based description of rewriting computations, involves the use of several techniques which have been recently investigated elsewhere: (i) the generation logical models and well-founded relations [19], (ii) modeling operational termination of CTRSs with conditional dependency pairs [23], (iii) the use of removal triples [24] generated by logical models [25], etc. Giving support to such techniques in termination proofs motivated the development of a new version of our tool, MU-TERM 6.0:

`http://zenon.dsic.upv.es/muterm`

We report on the new logic-based approach followed by MU-TERM 6.0, and also on the new features included since the last description of the system in 2010 [3].

* Supported by EU (FEDER), and projects RTI2018-094403-B-C32, PROMETEO/2019/098, and SP20180225. Also by INCIBE program "Ayudas para la excelencia de los equipos de investigación avanzada en ciberseguridad" (Raúl Gutiérrez).

## 2 New Features of MU-TERM

In the following, we enumerate the new features of MU-TERM 6.0 and illustrate them with some examples. Examples are intended to provide a better understanding of the techniques and often display solutions not necessarily obtained by an automatic proof with the tool, where the use of a specific proof strategy combining a sequence of *several* techniques (see Section 3) may dismiss the focused technique. Although we use some examples from other papers, all proofs of (operational) termination displayed here are new. For instance, the CTRS in Example 1 was proved operationally terminating in [25, Ex. 33], but the use of models in Example 3 below to show the absence of a link in the dependency graph is new. Examples 6 and 7 (of Order-Sorted TRSs) are discussed and proved here for the first time.

### 2.1 Logic-Based Representation of CTRSs

Given an oriented CTRS $\mathcal{R}$, with rules $\ell \to r \Leftarrow s_1 \approx t_1, \ldots, s_n \approx t_n$,[1] an inference system $\mathcal{I}(\mathcal{R})$ is obtained from the following generic inference system $\mathfrak{I}_{\mathrm{CTRS}}$

$$\text{(Rf)}\quad \frac{}{x \to^* x} \qquad \text{(C)}_{f,i}\quad \frac{x_i \to y_i}{f(x_1,\ldots,x_i,\ldots,x_k) \to f(x_1,\ldots,y_i,\ldots,x_k)} \text{ for all } f \in \mathcal{F}^{(k)} \text{ and } 1 \le i \le k$$

$$\text{(T)}\quad \frac{x \to y \quad y \to^* z}{x \to^* z} \qquad \text{(Rl)}_{\alpha}\quad \frac{s_1 \to^* t_1 \quad \cdots \quad s_n \to^* t_n}{\ell \to r} \text{ for } \alpha : \ell \to r \Leftarrow s_1 \approx t_1, \ldots, s_n \approx t_n \in \mathcal{R}$$

by *specializing* $(\mathrm{C})_{f,i}$ for each $k$-ary symbol $f$ in the signature $\mathcal{F}$ and $1 \le i \le k$, and $(\mathrm{Rl})_{\alpha}$ for all conditional rules $\alpha : \ell \to r \Leftarrow c$ in $\mathcal{R}$. Rules $\frac{B_1 \;\cdots\; B_n}{A}$ in $\mathcal{I}(\mathcal{R})$ are *schematic*: they can be used under any *instance* $\frac{\sigma(B_1) \;\cdots\; \sigma(B_n)}{\sigma(A)}$ by a substitution $\sigma$. We write $s \to_{\mathcal{R}} t$ ($s \to^*_{\mathcal{R}} t$) iff there is a proof tree for $s \to t$ ($s \to^* t$) using $\mathcal{I}(\mathcal{R})$. Operational termination of $\mathcal{R}$ is defined as the absence of infinite proof trees for goals $s \to t$ and $s \to^* t$ in $\mathcal{I}(\mathcal{R})$ [21]. In the analysis of computational properties of $\mathcal{R}$, we use the first-order *theory* $\overline{\mathcal{R}}$ obtained from $\mathcal{I}(\mathcal{R})$ by translating the inference rules $(\rho)\frac{B_1 \;\cdots\; B_n}{A}$ in $\mathcal{I}(\mathcal{R})$ into *sentences* $\overline{\rho}$ of the form $(\forall \boldsymbol{x})\ B_1 \wedge \cdots \wedge B_n \Rightarrow A$, for $\boldsymbol{x}$ the sequence of variables occurring in $A, B_1, \ldots, B_n$ [18, Sect. 4.5].

*Example 1.* For the following CTRS $\mathcal{R}$ [27, Ex. 7.2.51]

$$\mathsf{h}(\mathsf{d}) \to \mathsf{c}(\mathsf{a}) \quad (1) \qquad \mathsf{f}(\mathsf{k}(\mathsf{a}),\mathsf{k}(\mathsf{b}),x) \to \mathsf{f}(x,x,x) \quad (3)$$

$$\mathsf{h}(\mathsf{d}) \to \mathsf{c}(\mathsf{b}) \quad (2) \qquad \mathsf{g}(x) \to \mathsf{k}(y) \Leftarrow \mathsf{h}(x) \approx \mathsf{d}, \mathsf{h}(x) \approx \mathsf{c}(y) \quad (4)$$

the theory $\overline{\mathcal{R}}$ is given in Figure 1.

### 2.2 Operational Termination of Conditional Rewrite Systems

In [24, 25] a framework for automatically proving operational termination of (oriented) CTRSs using appropriate notions of *dependency pairs* (adapting the original notion for TRSs [5]) has been introduced: the 2D DP Framework.

[1] Oriented CTRSs treat conditions $s_i \approx t_i$ in rules as *rewriting* goals $\sigma(s_i) \to^* \sigma(t_i)$ for appropriate substitutions $\sigma$ [27, Def. 7.1.3].

$x \to^* x$ | $x \to y \wedge y \to^* z \Rightarrow x \to^* z$
$x_1 \to y_1 \Rightarrow \mathsf{f}(x_1, x_2, x_3) \to \mathsf{f}(y_1, x_2, x_3)$ | $x_2 \to y_2 \Rightarrow \mathsf{f}(x_1, x_2, x_3) \to \mathsf{f}(x_1, y_2, x_3)$
$x_3 \to y_3 \Rightarrow \mathsf{f}(x_1, x_2, x_3) \to \mathsf{f}(x_1, x_2, y_3)$ | $x \to y \Rightarrow \mathsf{c}(x) \to \mathsf{c}(y)$
$x \to y \Rightarrow \mathsf{g}(x) \to \mathsf{g}(y)$ | $x \to y \Rightarrow \mathsf{h}(x) \to \mathsf{h}(y)$
$x \to y \Rightarrow \mathsf{k}(x) \to \mathsf{k}(y)$ | $\mathsf{h}(\mathsf{d}) \to \mathsf{c}(\mathsf{a})$
$\mathsf{h}(\mathsf{d}) \to \mathsf{c}(\mathsf{b})$ | $\mathsf{f}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), x) \to \mathsf{f}(x, x, x)$
$\mathsf{h}(x) \to^* \mathsf{d} \wedge \mathsf{h}(x) \to^* \mathsf{c}(y) \Rightarrow \mathsf{g}(x) \to \mathsf{k}(y)$

**Fig. 1.** First-order theory $\overline{\mathcal{R}}$ for $\mathcal{R}$ in Example 1 (all variables universally quantified)

**Dependency Pairs for CTRSs.** Given a CTRS $\mathcal{R}$, two new CTRSs $\mathsf{DP}_H(\mathcal{R})$ and $\mathsf{DP}_V(\mathcal{R})$ are introduced to capture the two *horizontal* and *vertical* dimensions of operational termination of CTRSs [23]: the usual absence of infinite rewrite sequences (termination), and the absence of infinite climbings' on a proof tree when trying to prove a goal $s \to t$ or $s \to^* t$ (called $V$-termination). $\mathsf{DP}_H(\mathcal{R})$ consists of rules $u \to v \Leftarrow c$ whose terms $u$ and $v$ capture the progress of infinite rewrite sequences involving rules $\ell \to r \Leftarrow c$ with $u$ and $v$ marked versions of $\ell$ and a subterm of $r$ respectively (only the root symbol $f$ is marked as $f^\sharp$, or just capitalized: $F$). Similarly, $\mathsf{DP}_V(\mathcal{R})$ consists of rules $u \to v \Leftarrow d$ where $v$ is a marked subterm of $s_i$ for a condition $s_i \approx t_i$ in $c$ and $d$ is $s_1 \approx t_1, \ldots, s_{i-1} \approx t_{i-1}$.[2]

*Example 2.* For $\mathcal{R}$ in Example 1, we have $\mathsf{DP}_H(\mathcal{R}) = \{\mathsf{F}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), x) \to \mathsf{F}(x, x, x)\}$ and $\mathsf{DP}_V(\mathcal{R}) = \{\mathsf{G}(x) \to \mathsf{H}(x), \mathsf{G}(x) \to \mathsf{H}(x) \Leftarrow \mathsf{h}(x) \approx \mathsf{d}\}$

As in [7, Sect. 5], we use a set of sorts $S_{DP} = \{\mathrm{S}, \mathrm{P}\}$ so that symbols $f$ are (automatically) given a rank $f : \mathrm{S} \cdots \mathrm{S} \to \mathrm{S}$ and *marked* symbols are given rank $F : \mathrm{S} \cdots \mathrm{S} \to \mathrm{P}$ [25, Sect. 4.3]. Variables of formulas in $\overline{\mathcal{R}}$ (e.g., Figure 1) are then assumed to be universally quantified on sort S.

**The 2D DP Framework for CTRSs.** The absence of infinite *chains* of 2D DPs (i.e., sequences of 2D DPs which model infinite branches in the proof trees for goals $s \to t$ and $s \to^* t$) characterizes operational termination of CTRSs [23]. This is proved using a divide-and-conquer strategy which successively decomposes operational termination problems into smaller and simpler ones. Processors $\mathsf{P}$ are used for this purpose [24]. They simplify problems by decomposing or shrinking them. In particular, the appropriate *estimation* of graphs $\mathcal{G}$ whose nodes are dependency pairs is useful to analyze the existence of such infinite chains as *cycles* in the graph. The absence of cycles implies operational termination. The presence of conditional rules and pairs introduces some particular issues which we enumerate, and discuss below.

1. Some pairs could be *infeasible*, i.e., unable to be used in any of the aforementioned chains. Then, we could remove them [24, Sect. 4]. Also, arcs in $\mathcal{G}$ are defined by specific (often undecidable) *sequences* $s_1 \bowtie_1 t_1, \ldots, s_n \bowtie_n t_n$ (called f-sequences

[2] A third set of dependency pairs $\mathsf{DP}_{VH}(\mathcal{R}) \subseteq \mathsf{DP}_H(\mathcal{R})$ is used in [23]. For simplicity, in the examples of this paper, $\mathsf{DP}_{VH}(\mathcal{R})$ is empty and we pay no attention to it.

[15]) where $s_i$ and $t_i$ are terms and $\bowtie_i$ are predicates $\to_i^*$ that capture the possibility of having two nodes involved in a *chain* [20, Sect. 4.5] and must be proved feasible or infeasible (for some substitution $\sigma$ which applies to terms $s_i$ and $t_i$). A typical strategy is discarding arcs whose associated sequence is *infeasible*. We discuss this in paragraph *Infeasibility in Termination Proofs* below (see Examples 3 and 4).
2. Some pairs could be '*harmless*', i.e., unable to be persistently used in any infinite chain. This can be shown if we prove a 'decrease' when such pairs are used in a chain. Again, we can remove them to obtain a simplification [25, Sect. 4.3]. We discuss this in paragraph *Use of Well-Founded Relations* below (see Example 5).

**Infeasibility in Termination Proofs.** Given a (C)TRS $\mathcal{R}$ we say that a sequence $s_1 \to^* t_1, \ldots, s_n \to^* t_n$ is $\mathcal{R}$-infeasible if there is no substitution $\sigma$ such that $\sigma(s_i) \to_{\mathcal{R}}^* \sigma(t_i)$ holds for all $1 \le i \le n$. In [20] it is proved that a sequence $s_1 \to^* t_1, \ldots, s_n \to^* t_n$ is $\mathcal{R}$-infeasible if there is a model of $\overline{\mathcal{R}} \cup \{\neg(\exists \boldsymbol{x})\ s_1 \to^* t_1, \ldots, s_n \to^* t_n\}$, where $\boldsymbol{x}$ contains the variables in $s_1, t_1, \ldots, s_n, t_n$. In termination proofs, proving infeasibility is useful at different levels. As remarked above, when the conditional part $c$ of a pair $u \to v \Leftarrow c$ is proved infeasible, we can remove it. Also, the absence of an arc between two nodes (pairs) $u \to v \Leftarrow c$ and $u' \to v' \Leftarrow c'$ in the graph $\mathcal{G}$ can be treated as the infeasibility of $v \to^* u'$ (where, as usual, we assume that $v$ and $u'$ share no variable). For instance, for $\mathcal{R}$ in Example 1, it is possible to prove that there is no arc in the 'horizontal' graph which consists of a single node $\mathsf{F}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), x) \to \mathsf{F}(x, x, x)$ (the only dependency pair in $\mathsf{DP}_H(\mathcal{R})$) by just finding a model of

$$\overline{\mathcal{R}} \cup \{\neg(\exists x, y : \mathrm{S})\ \mathsf{F}(x, x, x) \to^* \mathsf{F}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), y)\} \tag{5}$$

For this purpose, model generators AGES [14] and Mace4 [26] are used by MU-TERM.

*Example 3.* We obtain a model $\mathcal{A}$ of (5) with Mace4. The domain is $\mathcal{A} = \{0, 1\}$ (Mace4 does not support sorts; thus, both S and P are merged into a single sort). Function and predicate symbols are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = \mathsf{d}^{\mathcal{A}} = 0 \qquad \mathsf{b}^{\mathcal{A}} = \mathsf{c}^{\mathcal{A}}(x) = 1 \quad \mathsf{f}^{\mathcal{A}}(x, y, z) = \mathsf{g}^{\mathcal{A}}(x) = 0$$

$$\mathsf{h}^{\mathcal{A}}(x) = 1 - x \qquad \mathsf{k}^{\mathcal{A}}(x) = x \quad \mathsf{F}^{\mathcal{A}}(x, y, z) = \begin{cases} 1 \text{ if } x = 0 \text{ and } y = 1 \\ 0 \text{ otherwise} \end{cases}$$

$$x(\to_{\mathcal{R}})^{\mathcal{A}} y \Leftrightarrow x = y \qquad x(\to_{\mathcal{R}}^*)^{\mathcal{A}} y \Leftrightarrow x = y$$

Discarding the arc would not be possible by the usual unification-based technique in [9]. With regard to infeasibility of pairs, consider the following well-known example.

*Example 4.* Consider the following CTRS $\mathcal{R}$ [8, p. 46]:

$$\mathsf{a} \to \mathsf{b} \tag{6}$$
$$\mathsf{f}(\mathsf{a}) \to \mathsf{b} \tag{7}$$
$$\mathsf{g}(x) \to \mathsf{g}(\mathsf{a}) \Leftarrow \mathsf{f}(x) \approx x \tag{8}$$

where $\mathsf{DP}_H(\mathcal{R}) = \{\mathsf{G}(x) \to \mathsf{G}(\mathsf{a}) \Leftarrow \mathsf{f}(x) \approx x, \mathsf{G}(x) \to \mathsf{A} \Leftarrow \mathsf{f}(x) \approx x\}$. Both pairs in $\mathsf{DP}_H(\mathcal{R})$ are $\mathcal{R}$-infeasible: no substitution $\sigma$ makes $\sigma(\mathsf{f}(x)) \to^* \sigma(x)$ true. We can prove it if a model $\mathcal{A}$ of $\overline{\mathcal{R}} \cup \{\neg(\exists x)\, \mathsf{f}(x) \to^* x\}$ can be found. We obtain a model with AGES. The domain is $\mathcal{A} = \mathbb{N} - \{0\}$ (since no marked symbol is involved, we can use a single interpretation domain); for function and predicate symbols:

$$\mathsf{a}^{\mathcal{A}} = 1 \quad \mathsf{b}^{\mathcal{A}} = 2 \quad \mathsf{f}^{\mathcal{A}}(x) = x + 1 \quad \mathsf{g}^{\mathcal{A}}(x) = 1 \quad x(\to_{\mathcal{R}})_{\mathrm{S}}^{\mathcal{A}} y \Leftrightarrow x(\to_{\mathcal{R}}^*)_{\mathrm{S}}^{\mathcal{A}} y \Leftrightarrow y \ge x$$

We can safely remove both pairs. Thus no infinite chain of pairs in $\mathsf{DP}_H(\mathcal{R})$ exists.

**Use of Well-Founded Relations.** The *removal triple processor* [24, Def. 70] implements the use of removal triples $(\gtrsim, \succeq, \sqsupset)$, including a well-founded relation $\sqsupset$ to remove pairs from, and hence simplify, termination problems. For instance, as shown in [25, Sect. 4.3], $\mathcal{R}$ in Example 1 is operationally terminating if we find a model $\mathcal{A}$ of

$$\mathcal{S}^{RT}_{\mathcal{R}} \cup \{(\forall x : \mathrm{S})\ \mathsf{F}(\mathsf{k}(\mathsf{a}), \mathsf{k}(\mathsf{b}), x)\ \pi_{\sqsupset}\ \mathsf{F}(x, x, x)\} \tag{9}$$

where $\pi_{\sqsupset}$ (a new predicate symbol representing $\sqsupset$) is interpreted as a *well-founded relation* $\pi^{\mathcal{A}}_{\sqsupset}$, and $\mathcal{S}^{RT}_{\mathcal{R}}$ extends $\overline{\mathcal{R}}$ with the following additional requirements to apply the processor [24, Defs. 68 and 69]:

$$(\forall x, y : \mathrm{P})\ \ x\ \pi_{\gtrsim}\ y \wedge y\ \pi_{\sqsupset}\ z \Rightarrow x\ \pi_{\sqsupset}\ z \tag{10}$$

$$(\forall x, y : \mathrm{P})\ \ x \to y \Rightarrow x\ \pi_{\gtrsim}\ y \tag{11}$$

No predicate $\pi_{\succeq}$ is necessary in this example (where a single pair is considered).

*Example 5.* We obtain a model $\mathcal{A}$ of (9) with AGES. Domains are $\mathcal{A}_{\mathrm{P}} = \{-1, 0, 1\}$ and $\mathcal{A}_{\mathrm{S}} = \{0, 1\}$. With regard to function and predicate symbols:

$$\mathsf{a}^{\mathcal{A}} = \mathsf{d}^{\mathcal{A}} = 0 \quad \mathsf{b}^{\mathcal{A}} = \mathsf{c}^{\mathcal{A}}(x) = 1 \qquad \mathsf{f}^{\mathcal{A}}(x, y, z) = \mathsf{g}^{\mathcal{A}}(x) = 0$$

$$\mathsf{h}^{\mathcal{A}}(x) = 1 \qquad \mathsf{k}^{\mathcal{A}}(x) = 1 - x \quad \mathsf{F}^{\mathcal{A}}(x, y, z) = x - y$$

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

$$x\ (\to^*)^{\mathcal{A}}_{\mathrm{S}}\ y \Leftrightarrow y \geq x \qquad x\ \pi^{\mathcal{A}}_{\gtrsim}\ y \Leftrightarrow x \geq y \qquad x\ \pi^{\mathcal{A}}_{\sqsupset}\ y \Leftrightarrow 6x \geq 1 + 6y$$

where, as in the semantic approach in [25, Sect. 4.3], $\to$ and $\to^*$ are *overloaded* for sorts P and S; thus, $(\to)^{\mathcal{A}}_{\mathrm{P}}$, $(\to^*)^{\mathcal{A}}_{\mathrm{P}}$, $(\to)^{\mathcal{A}}_{\mathrm{S}}$, and $(\to^*)^{\mathcal{A}}_{\mathrm{S}}$ are the corresponding interpretations. Note that $\pi^{\mathcal{A}}_{\sqsupset} = \{(x, y) \mid x, y \in \mathcal{A}_{\mathrm{P}}, 6x \geq 1 + 6y\} = \{(0, -1), (1, -1), (1, 0)\}$ is well-founded on $\mathcal{A}_{\mathrm{P}}$. Thus, we conclude operational termination of $\mathcal{R}$.

### 2.3 Termination of Order-Sorted Rewriting

Sorts are often used to reinforce program termination. Order-sorted dependency pairs were introduced in [22] for proving termination of order-sorted TRSs.

*Example 6.* The following many-sorted TRS $\mathcal{R}$ in [29, Sect. 3.3] (in the hopefully self-explained Maude format [6]) is a terminating version of Toyama's example, which is nonterminating as a TRS (i.e., without sort information):

```
mod Toyama-MS is
 sorts S1 S2 .
 ops a b : -> S1 .     op f : S1 S1 S1 -> S1 .    op g : S2 S2 -> S2 .
 vars x : S1 .         vars y z : S2 .
 rl g(y,z) => y .      rl g(y,z) => z .           rl f(a,b,x) => f(x,x,x) .
endm
```

The 2010 version of MU-TERM could not prove it terminating.[3] According to [22], $\mathcal{R}$ has a single dependency pair:

$$\mathsf{F}(\mathsf{a}, \mathsf{b}, x) \to \mathsf{F}(x, x, x) \tag{12}$$

[3] Benchmarks available here: `http://zenon.dsic.upv.es/muterm/benchmarks/benchmarks-ostrs/benchmarks.html`.

where $\mathsf{F}$ has rank $\mathtt{S1}\ \mathtt{S1}\ \mathtt{S1} \to \mathtt{P}$ for a new sort $\mathtt{P}$ [22, Sect. 3.2] and $x$ has sort $\mathtt{S1}$. We can prove that the dependency graph consisting of this single pair has no cycle. With AGES we can compute a model of $\overline{\mathcal{R}} \cup \{\neg(\exists x, y : \mathtt{S1})\, \mathsf{F}(x,x,x) \to^* \mathsf{F}(\mathsf{a},\mathsf{b},y)\}$ which is as follows: $\mathcal{A}_{\mathtt{S1}} = \{0,1\}$, $\mathcal{A}_{\mathtt{S2}} = \{1\}$, $\mathcal{A}_{\mathtt{P}} = -\mathbb{N}$ (i.e., the set of nonpositive integers), and functions and predicates interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = 1 \qquad \mathsf{b}^{\mathcal{A}} = 0 \qquad \mathsf{f}^{\mathcal{A}}(x,y,z) = 1 \qquad \mathsf{g}^{\mathcal{A}}(x) = 1 \qquad \mathsf{F}^{\mathcal{A}}(x,y,z) = x - y - 1$$

$$x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{S1}} y \Leftrightarrow x = y = 1 \qquad x(\to^*_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{S1}} y \Leftrightarrow \mathit{true} \qquad x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{S2}} y \Leftrightarrow \mathit{true}$$

$$x(\to^*_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{S2}} y \Leftrightarrow \mathit{true} \qquad x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{P}} y \Leftrightarrow x = y \qquad x(\to^*_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{P}} y \Leftrightarrow x \geq y$$

The crucial point to obtain the proof in Example 6 is the ability to provide different interpretations to different sorts. The following example from [28] could not be handled by the 2010 version of MU-TERM because orderings were generated without paying attention to sorts (see [22, Sect. 6]).

*Example 7.* The following OS-TRS $\mathcal{R}$ [28, Ex. 11] is nonterminating as a TRS:

```
mod Example11-OL96 is
  sorts S S1 S2 S3 S4 .  subsorts S1 S2 S3 S4 < S .
  ops f g : S -> S .     op g : S3 -> S1 .          op g : S4 -> S2 .
  op h : S1 -> S2 .      op a : -> S3 .             op b : -> S4 .
  var x : S1 .           rl f(x) => f(h(x)) .       rl a => b .
endm
```

There is a single OS-DP for $\mathcal{R}$: $\mathsf{F}(x) \to \mathsf{F}(\mathsf{h}(x))$, where $\mathsf{F}$ has rank $\mathtt{S} \to \mathtt{P}$ for a new sort $\mathtt{P}$ and $x$ has sort $\mathtt{S1}$. We can prove termination of $\mathcal{R}$ by finding a removal triple $(\gtrsim, \succeq, \sqsupset)$ such that the rules of $\mathcal{R}$ are compatible with $\gtrsim$, and $\mathsf{F}(x) \sqsupset \mathsf{F}(\mathsf{h}(x))$ holds whenever $x$ ranges on terms of sort $\mathtt{S1}$. With AGES we obtain an interpretation $\mathcal{A}$ as follows: sorts are interpreted as $\mathcal{A}_{\mathtt{S}} = \{-1,0,1\}$, $\mathcal{A}_{\mathtt{S1}} = \{-1\}$, $\mathcal{A}_{\mathtt{S2}} = \{-1,0\}$, $\mathcal{A}_{\mathtt{S3}} = \{-1\}$, $\mathcal{A}_{\mathtt{S4}} = \{-1,0\}$, and $\mathcal{A}_{\mathtt{P}} = \mathbb{N} \cup \{-1\}$. Functions and predicates are interpreted as follows:

$$\mathsf{a}^{\mathcal{A}} = -1 \qquad \mathsf{b}^{\mathcal{A}} = 0 \qquad \mathsf{f}^{\mathcal{A}}(x,y,z) = x \qquad \mathsf{g}^{\mathcal{A}}_{\mathtt{S}}(x) = x$$

$$\mathsf{g}^{\mathcal{A}}_{\mathtt{S3}}(x) = -1 \qquad \mathsf{g}^{\mathcal{A}}_{\mathtt{S4}}(x) = x \qquad \mathsf{h}^{\mathcal{A}}(x,y,z) = 0 \qquad \mathsf{F}^{\mathcal{A}}(x,y,z) = -x$$

(where different overloaded versions of $\mathsf{g}$ use the input sort as a subindex) and

$$x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{S}} y \Leftrightarrow y = 0 \wedge x = -1 \qquad x(\to^*_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{S}} y \Leftrightarrow \mathit{true} \qquad x(\to_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{P}} y \Leftrightarrow x \geq y$$

$$x(\to^*_{\mathcal{R}})^{\mathcal{A}}_{\mathtt{P}} y \Leftrightarrow \mathit{true} \qquad x \gtrsim^{\mathcal{A}} y \Leftrightarrow x \geq y \qquad x \sqsupset^{\mathcal{A}} y \Leftrightarrow x > y$$

Note that the interpretation of the 'original' rewrite relation concerns sort $\mathtt{S}$ only because it is the top sort of the full sort hierarchy.

### 2.4 Termination of Context-Sensitive Rewriting

In *context-sensitive rewriting* (CSR [16]), a *replacement map* $\mu$ is used to restrict the arguments $\mu(f) \subseteq \{1,\ldots,k\}$ which can be rewritten for each $k$-ary symbol $f$. The restriction on arguments is top-down propagated to positions of terms $t$, which are called *active* positions of $t$. We write $s \hookrightarrow t$ if an *active* subterm of $s$ can be rewritten so that $s \to t$. In the *dependency pair approach* for proving termination of CSR [2], rules of the form $f(\ell_1,\ldots,\ell_k) \to r$ are given *dependency pairs* $f^\sharp(\ell_1,\ldots,\ell_k) \to g^\sharp(s_1,\ldots,s_m)$, for $s = g(s_1,\ldots,s_m)$ a *replacing* subterm of $r$ (i.e., a subterm $s = r|_p$ occurring at an

active position $p$ of $r$) and $g$ a defined symbol. The notation $f^\sharp$ means that $f$ is *marked* (capital letters $F$ are often used instead of $f^\sharp$). However, due to rules $\ell \to r \in \mathcal{R}$ with *migrating* variables $x \in \mathcal{V}ar^\mu(r) \setminus \mathcal{V}ar^\mu(\ell)$ (that are frozen, i.e., not active, in $\ell$ but become active in $r$, possibly 'awaking' infinite rewrite sequences), we also need *collapsing dependency pairs* $\ell^\sharp \to x$ where $x$ is a *migrating variable* of the rule.

*Example 8.* For the following TRS $\mathcal{R}$ in [30, Introd.]

$$\begin{aligned}
\mathsf{p}(\mathsf{s}(x)) &\to x & \mathsf{if}(\mathsf{true}, x, y) &\to x\\
0 + x &\to x & \mathsf{if}(\mathsf{false}, x, y) &\to y\\
\mathsf{s}(x) + y &\to \mathsf{s}(x + y) & \mathsf{zero}(0) &\to \mathsf{true}\\
0 \times y &\to 0 & \mathsf{zero}(\mathsf{s}(x)) &\to \mathsf{false}\\
\mathsf{s}(x) \times y &\to y + (x \times y) & \mathsf{fact}(x) &\to \mathsf{if}(\mathsf{zero}(x), \mathsf{s}(0), \mathsf{fact}(\mathsf{p}(x)) \times x)
\end{aligned}$$

and $\mu$ given by $\mu(\mathsf{if}) = \{1\}$ and $\mu(f) = \{1, \ldots, k\}$ for any other $k$-ary symbol $f$ [12, Ex. 1]. $\mathsf{DP}(\mathcal{R}, \mu)$ consists of pairs

$$\begin{aligned}
\mathsf{s}(x) +^\sharp y &\to x +^\sharp y & \mathsf{s}(x) \times^\sharp y &\to y +^\sharp (x \times y) & \mathsf{s}(x) \times^\sharp y &\to x \times^\sharp y\\
\mathsf{FACT}(x) &\to \mathsf{ZERO}(x) & \mathsf{FACT}(x) &\to \mathsf{IF}(\mathsf{zero}(x), \mathsf{s}(0), \mathsf{fact}(\mathsf{p}(x)) \times x)\\
\mathsf{IF}(\mathsf{true}, x, y) &\to x & \mathsf{IF}(\mathsf{false}, x, y) &\to y
\end{aligned}$$

Collapsing pairs capture a kind of *recursion* which is *hidden* below frozen parts of the terms involved in infinite context-sensitive rewrite sequences until a *migrating* variable within a rule $\ell \to r$ shows them up. The *hidden terms* of a TRS $\mathcal{R}$ are defined subterms occurring at frozen positions in the *rhs* of some rule of $\mathcal{R}$ [2]. *Hiding contexts* are contexts where hidden terms can occur *at active positions* within a context-sensitive rewrite sequence [1, 13]. There, hidden terms can *restart* a delayed recursive call after the application of a rule with migrating variables (see [12] for a detailed analysis). For $\mathcal{R}$ and $\mu$ in Example 8, the only rule with hidden terms is $\mathsf{fact}(x) \to \mathsf{if}(\mathsf{zero}(x), \mathsf{s}(0), \mathsf{fact}(\mathsf{p}(x)) \times x)$. Symbols $\mathsf{fact}$ and '$\times$' hide position 1. Symbol '$\times$' does *not* hide position 2 because the second occurrence of variable $x$ in $\mathsf{fact}(\mathsf{p}(x)) \times x$ is not frozen in the *lhs* $\ell$ of the rule. Symbol $\mathsf{p}$ hides no position. The refinements introduced in [12] have led to a more precise notion of hidden terms and contexts, enabling a better analysis of the connections between them. This has greatly improved the ability of MU-TERM to prove termination of CSR. For instance, the proof of termination of $\mathcal{R}$ and $\mu$ in Example 8, which could not be obtained with the 2010 version of MU-TERM, is now possible with MU-TERM 6.0, see the proof of `CSR_04/ExIntrod_Zan97.xml` in the 2019 Termination Competition

`http://group-mmm.org/termination/competitions/Y2019/caches/termination_33019.html`

or in our local benchmarks:

`http://zenon.dsic.upv.es/muterm/benchmarks/ijcar20/TRS_Contextsensitive/benchmarks.html`

## 3 Termination Expert

The arbitrary application of processors can generate a huge search space. Furthermore, proofs usually proceed under some *timeout*. For this reason, we need to choose a fixed strategy where fast processors that reduce the number of rules are first used, and slow processors, or processors that increase the number of rules, are used when fast processors fail. Hence, the frequency of use for the different processors depends on the

chosen strategy. With small differences depending on the particular kind of problem, we do the following:

1. If $\mathcal{R}$ is a TRS or a CS-TRS, we check whether the system is innermost equivalent [3, Sect. 2.2]. If it is true, then we transform the problem into an innermost one.
2. Then, we obtain the corresponding dependency pairs, obtaining a CTRS, OS, CS, or DP problem. Then we perform the following steps repeatedly
   (a) Decision point between processors for proving (operational) non-termination and the *strongly connected component* (SCC) processor.
   (b) Subterm criterion processor.
   (c) Removal triple processor generating models with AGES (we try different configurations, from simpler to more complex).
   (d) If $\mathcal{R}$ is a CTRS, we apply simplification and removal processors on the conditions (using AGES when a model is necessary).
   (e) Transformation processors on rules, pairs and conditions: instantiation, forward instantiation, and narrowing.

Full explanations of the processors can be found in [4, 12, 13, 19, 20, 24, 25]. The MU-TERM 6.0 logic-based approach has led to dramatic improvements, as reported here:

`http://zenon.dsic.upv.es/muterm/benchmarks/ijcar20/Comparison/benchmarks.html`

where the use of logical models is compared with the exclusive use of *polynomial interpretations* (as in MU-TERM 5.0). Polynomial interpretations are strictly less powerful in terms of solved examples (as every proof using polynomial interpretations can be obtained using the new logic-based approach). However, we *keep* them in MU-TERM 6.0 as they lead to *faster* proofs. We use polynomial interpretations as part of MU-TERM 6.0 strategy (via the *removal triple* processor).

MU-TERM 6.0 consists of more than 30000 lines of Haskell code. In the web-based interface, besides the fully automatic use of the termination expert, we can also use specific techniques like polynomial orderings, matrix interpretations, (context-sensitive) recursive path ordering, etc., which we have found useful for *teaching* purposes.

## 4 Experimental Evaluation

Since 2014, MU-TERM has proven to be the most powerful tool for proving operational termination of *conditional* rewriting and termination of *context-sensitive* rewriting, each year winning the corresponding subcategory of the annual International Competition of Termination Tools, see `http://zenon.dsic.upv.es/muterm/?page_id=82` for an historical account. In the CSR subcategory, since 2014 MU-TERM is able to prove *all* the examples proved by any other participating tool (thanks to the results in [12]).

The benchmarks web page of MU-TERM reports on specific experiments comparing the 2010 and 2020 versions. First, the 2010 version did not support CTRSs. For CS-TRSs, three new examples can be proved now (and all the examples handled by the 2010 version are also handled now). As for OS-TRSs, MU-TERM 6.0 is able to prove or disprove termination of *all* the OS-TRSs in the 2010 benchmark suite (except a non-sort-decreasing OS-TRS, not covered by the theory in [22], where sort-decreasingness [11] is required). The 2010 version could not disprove termination of OS-TRSs.

*Acknowledgments* We thank the anonymous referees for many remarks and suggestions that led to improve the paper.

## References


1. Alarcón, B., Emmes, F., Fuhs, C., Giesl, J., Gutiérrez, R., Lucas, S., Schneider-Kamp, P., Thiemann, R.: Improving context-sensitive dependency pairs. In: Cervesato, I., Veith, H., Voronkov, A. (eds.) Logic for Programming, Artificial Intelligence, and Reasoning, 15th International Conference, LPAR 2008, Doha, Qatar, November 22-27, 2008. Proceedings. Lecture Notes in Computer Science, vol. 5330, pp. 636–651. Springer (2008). https://doi.org/10.1007/978-3-540-89439-1_44
2. Alarcón, B., Gutiérrez, R., Lucas, S.: Context-sensitive dependency pairs. Inf. Comput. **208**(8), 922–968 (2010). https://doi.org/10.1016/j.ic.2010.03.003
3. Alarcón, B., Gutiérrez, R., Lucas, S., Navarro-Marset, R.: Proving termination properties with mu-term. In: Johnson, M., Pavlovic, D. (eds.) Algebraic Methodology and Software Technology - 13th International Conference, AMAST 2010, Lac-Beauport, QC, Canada, June 23-25, 2010. Revised Selected Papers. Lecture Notes in Computer Science, vol. 6486, pp. 201–208. Springer (2010). https://doi.org/10.1007/978-3-642-17796-5_12
4. Alarcón, B., Lucas, S., Meseguer, J.: A dependency pair framework for $A \vee C$-termination. In: Ölveczky, P.C. (ed.) Rewriting Logic and Its Applications - 8th International Workshop, WRLA 2010, Held as a Satellite Event of ETAPS 2010, Paphos, Cyprus, March 20-21, 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6381, pp. 35–51. Springer (2010). https://doi.org/10.1007/978-3-642-16310-4_4
5. Arts, T., Giesl, J.: Termination of term rewriting using dependency pairs. Theor. Comput. Sci. **236**(1-2), 133–178 (2000). https://doi.org/10.1016/S0304-3975(99)00207-8
6. Clavel, M., Durán, F., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer, J., Talcott, C.L. (eds.): All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic, Lecture Notes in Computer Science, vol. 4350. Springer (2007). https://doi.org/10.1007/978-3-540-71999-1
7. Endrullis, J., Waldmann, J., Zantema, H.: Matrix interpretations for proving termination of term rewriting. J. Autom. Reasoning **40**(2-3), 195–220 (2008). https://doi.org/10.1007/s10817-007-9087-9
8. Giesl, J., Arts, T.: Verification of erlang processes by dependency pairs. Appl. Algebra Eng. Commun. Comput. **12**(1/2), 39–72 (2001). https://doi.org/10.1007/s002000100063
9. Giesl, J., Thiemann, R., Schneider-Kamp, P.: Proving and disproving termination of higher-order functions. In: Gramlich, B. (ed.) Frontiers of Combining Systems, 5th International Workshop, FroCoS 2005, Vienna, Austria, September 19-21, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3717, pp. 216–231. Springer (2005). https://doi.org/10.1007/11559306_12
10. Giesl, J., Thiemann, R., Schneider-Kamp, P., Falke, S.: Mechanizing and improving dependency pairs. J. Autom. Reasoning **37**(3), 155–203 (2006). https://doi.org/10.1007/s10817-006-9057-7
11. Goguen, J.A., Meseguer, J.: Order-sorted algebra I: equational deduction for multiple inheritance, overloading, exceptions and partial operations. Theor. Comput. Sci. **105**(2), 217–273 (1992). https://doi.org/10.1016/0304-3975(92)90302-V
12. Gutiérrez, R., Lucas, S.: Function Calls at Frozen Positions in Termination of Context-Sensitive Rewriting. In: Martí-Oliet, N., Ölveczky, P., Talcott, C.L. (eds.)

Logic, Rewriting, and Concurrency - Essays dedicated to José Meseguer on the Occasion of His 65th Birthday. LNCS, vol. 9200, pp. 311–330. Springer (2015). https://doi.org/10.1007/978-3-319-23165-5_15
13. Gutiérrez, R., Lucas, S.: Proving termination in the context-sensitive dependency pair framework. In: Ölveczky, P.C. (ed.) Rewriting Logic and Its Applications - 8th International Workshop, WRLA 2010, Held as a Satellite Event of ETAPS 2010, Paphos, Cyprus, March 20-21, 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6381, pp. 18–34. Springer (2010). https://doi.org/10.1007/978-3-642-16310-4_3
14. Gutiérrez, R., Lucas, S.: Automatic generation of logical models with AGES. In: Fontaine, P. (ed.) Automated Deduction - CADE 27 - 27th International Conference on Automated Deduction, Natal, Brazil, August 27-30, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11716, pp. 287–299. Springer (2019). https://doi.org/10.1007/978-3-030-29436-6_17
15. Gutiérrez, R., Lucas, S.: Automatically Proving and Disproving Feasibility Conditions). In: Peltier, N., Sofronie-Stokkermans, V. (eds.) Proc. of the 7th International Joint Conference on Automated Reasoning, IJCAR 2020. LNCS, vol. to appear. Springer (2020)
16. Lucas, S.: Context-sensitive computations in functional and functional logic programs. J. Funct. Log. Program. **1998**(1) (1998), `http://danae.uni-muenster.de/lehre/kuchen/JFLP/articles/1998/A98-01/A98-01.html`
17. Lucas, S.: Context-sensitive rewriting strategies. Inf. Comput. **178**(1), 294–343 (2002). https://doi.org/10.1006/inco.2002.3176
18. Lucas, S.: Proving semantic properties as first-order satisfiability. Artif. Intell. **277** (2019). https://doi.org/10.1016/j.artint.2019.103174
19. Lucas, S., Gutiérrez, R.: Automatic synthesis of logical models for order-sorted first-order theories. J. Autom. Reasoning **60**(4), 465–501 (2018). https://doi.org/10.1007/s10817-017-9419-3
20. Lucas, S., Gutiérrez, R.: Use of logical models for proving infeasibility in term rewriting. Inf. Process. Lett. **136**, 90–95 (2018). https://doi.org/10.1016/j.ipl.2018.04.002
21. Lucas, S., Marché, C., Meseguer, J.: Operational termination of conditional term rewriting systems. Inf. Process. Lett. **95**(4), 446–453 (2005). https://doi.org/10.1016/j.ipl.2005.05.002
22. Lucas, S., Meseguer, J.: Order-sorted dependency pairs. In: Antoy, S., Albert, E. (eds.) Proceedings of the 10th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming, July 15-17, 2008, Valencia, Spain. pp. 108–119. ACM (2008). https://doi.org/10.1145/1389449.1389463
23. Lucas, S., Meseguer, J.: Dependency pairs for proving termination properties of conditional term rewriting systems. J. Log. Algebraic Methods Program. **86**(1), 236–268 (2017). https://doi.org/10.1016/j.jlamp.2016.03.003
24. Lucas, S., Meseguer, J., Gutiérrez, R.: The 2D Dependency Pair Framework for conditional rewrite systems. Part I: Definition and basic processors. J. Comput. Syst. Sci. **96**, 74–106 (2018). https://doi.org/10.1016/j.jcss.2018.04.002
25. Lucas, S., Meseguer, J., Gutiérrez, R.: The 2D Dependency Pair Framework for Conditional Rewrite Systems—Part II: Advanced Processors and Implementation Techniques. Journal of Automated Reasoning **in press** (2020). https://doi.org/10.1007/s10817-020-09542-3
26. McCune, W.: Prover9 & Mace4. Tech. rep. (2005–2010), `http://www.cs.unm.edu/~mccune/prover9/`

27. Ohlebusch, E.: Advanced topics in term rewriting. Springer (2002), http://www.springer.com/computer/swe/book/978-0-387-95250-5
28. Ölveczky, P.C., Lysne, O.: Order-sorted termination: The unsorted way. In: Hanus, M., Rodríguez-Artalejo, M. (eds.) Algebraic and Logic Programming, 5th International Conference, ALP'96, Aachen, Germany, September 25-27, 1996, Proceedings. Lecture Notes in Computer Science, vol. 1139, pp. 92–106. Springer (1996). https://doi.org/10.1007/3-540-61735-3_6
29. Zantema, H.: Termination of term rewriting: Interpretation and type elimination. J. Symb. Comput. **17**(1), 23–50 (1994). https://doi.org/10.1006/jsco.1994.1003
30. Zantema, H.: Termination of context-sensitive rewriting. In: Comon, H. (ed.) Rewriting Techniques and Applications, 8th International Conference, RTA-97, Sitges, Spain, June 2-5, 1997, Proceedings. Lecture Notes in Computer Science, vol. 1232, pp. 172–186. Springer (1997). https://doi.org/10.1007/3-540-62950-5_69