

Mechanizing Proofs of Termination in the Context-Sensitive Dependency Pairs Framework

Raúl Gutiérrez^{1,2}

*Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Valencia, Spain*

Salvador Lucas^{1,3}

*Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
Valencia, Spain*

Abstract

The dependency pairs approach, one of the most powerful techniques for proving termination of rewriting, has been recently adapted to be used for proving termination of context-sensitive rewriting (CSR). The notion of context-sensitive dependency pair (CS-DP) is different from the standard one in that collapsing dependency pairs (i.e., rules whose right-hand side is a variable) are considered. Although the implementation and practical use of CS-DPs leads to a very powerful framework which improves the current state-of-the-art of methods for proving termination of CSR, handling collapsing pairs is not easy and often leads to impose heavy requirements over the base orderings which are used to achieve the proofs. A recent proposal removes collapsing pairs by transforming them into sets of new (standard) pairs. In this way, though, the complexity of the obtained dependency graph is heavily increased and the role of collapsing pairs for modeling context-sensitive computations gets lost. This leads to a less intuitive and accurate description of the termination behavior of the system. In this paper, we show how to get the best of the two approaches, thus obtaining a more powerful dependency pair framework which hopefully fulfills all practical and theoretical expectations.

Keywords: Context-sensitive rewriting, termination, dependency pairs.

1 Introduction

In Context-Sensitive Rewriting (CSR, [17]), a *replacement map* μ satisfying $\mu(f) \subseteq \{1, \dots, \text{ar}(f)\}$ for every function symbol f in the signature \mathcal{F} is used to discriminate the argument positions on which the rewriting steps are allowed. In this

¹ Partially supported by the EU (FEDER) and the Spanish MEC/MICINN, under grants TIN 2004-7943-C04-02, HA 2006-0007 and TIN 2007-68093-C02-02.

² Email: rgutierrez@dsic.upv.es

³ Email: slucas@dsic.upv.es

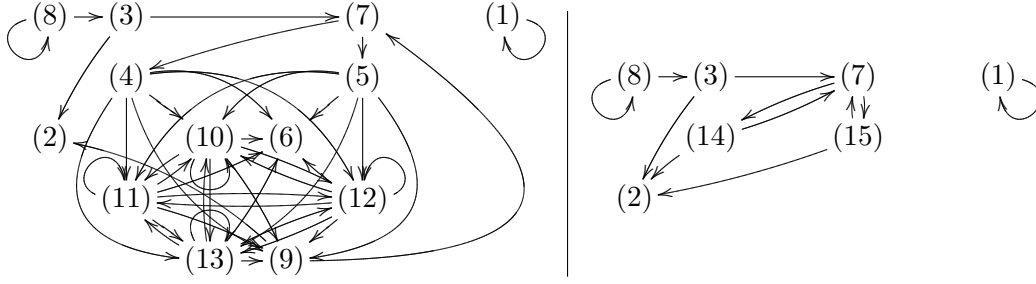


Fig. 1. Dependency graph for Example 1.1 following [1] (left) and [5] (right)

way, a terminating behavior of (context-sensitive) computations with Term Rewriting Systems (TRSs) can be obtained. CSR has shown useful to model evaluation strategies in programming languages [8,9,10,11,12,18]. In [3,4], Arts and Giesl's dependency pairs approach, a powerful technique for proving termination of rewriting, was adapted to CSR (see [5] for a more recent presentation). In [1], a transformation that replaces the *collapsing dependency pairs* (i.e., pairs whose right hand sides are variables, see [3]) by a new set of pairs that simulate their behavior was introduced. This new set of pairs is used to simplify the definition of context-sensitive dependency chain; but, on the other hand, we loose the intuition of what collapsing pairs mean in a context-sensitive rewriting chain, and some processors which are based on collapsing dependency pairs cannot be used anymore (see [5]). Furthermore, understanding the new dependency graph is harder.

Example 1.1 Consider the context-sensitive term rewriting system (CS-TRS) in [1]

$$\begin{array}{ll}
 \text{gt}(0, y) \rightarrow \text{false} & \text{p}(0) \rightarrow 0 \\
 \text{gt}(s(x), 0) \rightarrow \text{true} & \text{p}(s(x)) \rightarrow x \\
 \text{gt}(s(x), s(y)) \rightarrow \text{gt}(x, y) & \text{minus}(x, y) \rightarrow \text{if}(\text{gt}(y, 0), \text{minus}(p(x), p(y)), x) \\
 \text{if}(\text{true}, x, y) \rightarrow x & \text{div}(0, s(y)) \rightarrow 0 \\
 \text{if}(\text{false}, x, y) \rightarrow y & \text{div}(s(x), s(y)) \rightarrow s(\text{div}(\text{minus}(x, y), s(y)))
 \end{array}$$

with $\mu(\text{if}) = \{1\}$ and $\mu(f) = \{1, \dots, \text{ar}(f)\}$ for all other symbols f . Note that if no replacement restriction is considered, then the following sequence is possible and the system would be nonterminating:

$$\underline{\text{minus}(0, 0)} \rightarrow_{\mathcal{R}}^* \text{if}(\text{gt}(0, 0), \underline{\text{minus}(0, 0)}, 0) \rightarrow_{\mathcal{R}}^* \text{if}(\dots, \text{if}(\text{gt}(0, 0), \underline{\text{minus}(0, 0)}, 0), \dots) \rightarrow_{\mathcal{R}}^* \dots$$

If we follow the transformational definition in [1] we have the following dependency pairs (a new symbol U is introduced):

$$\begin{array}{ll}
 \text{GT}(s(x), s(y)) \rightarrow \text{GT}(x, y) & (1) & \text{M}(x, y) \rightarrow \text{IF}(\text{gt}(y, 0), \text{minus}(p(x), p(y)), x) & (7) \\
 \text{M}(x, y) \rightarrow \text{GT}(y, 0) & (2) & \text{D}(s(x), s(y)) \rightarrow \text{D}(\text{minus}(x, y), s(y)) & (8) \\
 \text{D}(s(x), s(y)) \rightarrow \text{M}(x, y) & (3) & \text{U}(\text{minus}(p(x), p(y))) \rightarrow \text{M}(p(x), p(y)) & (9) \\
 \text{IF}(\text{true}, x, y) \rightarrow \text{U}(x) & (4) & \text{U}(p(x)) \rightarrow \text{U}(x) & (10) \\
 \text{IF}(\text{false}, x, y) \rightarrow \text{U}(y) & (5) & \text{U}(p(y)) \rightarrow \text{U}(y) & (11) \\
 \text{U}(p(x)) \rightarrow \text{P}(x) & (6) & \text{U}(\text{minus}(x, y)) \rightarrow \text{U}(x) & (12) \\
 & & \text{U}(\text{minus}(x, y)) \rightarrow \text{U}(y) & (13)
 \end{array}$$

and the dependency graph has the unreadable aspect shown in Figure 1 (left). In contrast, if we consider the original definition of CS-DPs and CS-DG in [3,4], our

set of dependency pairs is the following:

$$\begin{array}{ll}
 \text{GT}(s(x), s(y)) \rightarrow \text{GT}(x, y) & (1) \qquad \text{M}(x, y) \rightarrow \text{IF}(\text{gt}(y, 0), \text{minus}(p(x), p(y)), x) & (7) \\
 \text{M}(x, y) \rightarrow \text{GT}(y, 0) & (2) \qquad \text{D}(s(x), s(y)) \rightarrow \text{D}(\text{minus}(x, y), s(y)) & (8) \\
 \text{D}(s(x), s(y)) \rightarrow \text{M}(x, y) & (3) \qquad \text{IF}(\text{true}, x, y) \rightarrow x & (14) \\
 & \qquad \text{IF}(\text{false}, x, y) \rightarrow y & (15)
 \end{array}$$

and the dependency graph is much more clear, see Figure 1 (right).

The work in [1] was motivated by the fact that mechanizing proofs of termination of CSR according to the results in [3] can be difficult due to the presence of collapsing dependency pairs. The problem is that [3] imposes hard restrictions on the orderings which are used in proofs of termination of CSR when collapsing dependency pairs are present. In this paper we address this problem in a different way. We keep collapsing CS-DPs (and their descriptive power and simplicity) while the practical problems for handling them are overcome in an elegant way.

After some preliminaries in Section 2, in Section 3 we introduce the notion of hidden term and hidden context and discuss their role in infinite μ -rewrite sequences. Then, in Section 4 we introduce a new notion of CS-DP chain which is well-suited for mechanizing proofs of termination of CSR with CS-DPs. Finally, in Section 5 we introduce a dependency pairs framework which is more appropriate for proving termination of CSR and is as powerful as the approach in [1]. Furthermore, we show that with the new definition we can also use all the existing processors from the two previous approaches. Section 6 concludes.

2 Preliminaries

We assume a basic knowledge about standard definitions and notations for term rewriting as given in, e.g., [7]. Positions p, q, \dots are represented by chains of positive natural numbers used to address subterms of t . Given positions p, q , we denote its concatenation as $p.q$. If p is a position, and Q is a set of positions, $p.Q = \{p.q \mid q \in Q\}$. We denote the empty chain by Λ . The set of positions of a term t is $\text{Pos}(t)$. The *subterm* at position p of t is denoted as $t|_p$ and $t[s]_p$ is the term t with the subterm at position p replaced by s . We write $t \succeq s$ if $s = t|_p$ for some $p \in \text{Pos}(t)$ and $t \triangleright s$ if $t \succeq s$ and $t \neq s$. The symbol labeling the root of t is denoted as $\text{root}(t)$. A *substitution* is a function $\sigma : \mathcal{X} \rightarrow \mathcal{T}(\mathcal{F}, \mathcal{X})$ such that $\sigma(x) \neq x$. A *context* is a term $C \in \mathcal{T}(\mathcal{F} \cup \{\square\}, \mathcal{X})$ with zero or more ‘holes’ \square (a fresh constant symbol). A *rewrite rule* is an ordered pair (ℓ, r) , written $\ell \rightarrow r$, with $\ell, r \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, $\ell \notin \mathcal{X}$ and $\text{Var}(r) \subseteq \text{Var}(\ell)$. The left-hand side (*lhs*) of the rule is ℓ and r is the right-hand side (*rhs*). A *TRS* is a pair $\mathcal{R} = (\mathcal{F}, R)$ where R is a set of rewrite rules. 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 $\mathcal{D} = \{\text{root}(\ell) \mid \ell \rightarrow r \in R\}$ and $\mathcal{C} = \mathcal{F} \setminus \mathcal{D}$.

In the following, we introduce some notions and notation about CSR [17]. A mapping $\mu : \mathcal{F} \rightarrow \wp(\mathbb{N})$ is a *replacement map* if $\forall f \in \mathcal{F}$, $\mu(f) \subseteq \{1, \dots, \text{ar}(f)\}$. Let $M_{\mathcal{F}}$ be the set of all replacement maps (or $M_{\mathcal{R}}$ for the replacement map of a TRS (\mathcal{F}, R)). The set of *active positions* $\text{Pos}^{\mu}(t)$ of $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ is: $\text{Pos}^{\mu}(t) = \{\Lambda\}$, if $t \in \mathcal{X}$ and $\text{Pos}^{\mu}(t) = \{\Lambda\} \cup \bigcup_{i \in \mu(\text{root}(t))} i.\text{Pos}^{\mu}(t|_i)$, if $t \notin \mathcal{X}$. The set of *replacing*

variables of t is $\mathcal{V}ar^\mu(t) = \{x \in \mathcal{V}ar(t) \mid \exists p \in \mathcal{P}os^\mu(t), t|_p = x\}$. The *replacing subterm* relation \triangleright_μ is given by $t \triangleright_\mu s$ if there is $p \in \mathcal{P}os^\mu(t)$ such that $s = t|_p$. We write $t \triangleright_\mu s$ if $t \triangleright_\mu s$ and $t \neq s$. We write $t \triangleright_\mu^\# s$ to denote that s is a *nonreplacing strict subterm* of t : $t \triangleright_\mu^\# s$ if there is a *frozen position* p , i.e. $p \in \mathcal{P}os(t) \setminus \mathcal{P}os^\mu(t)$, such that $s = t|_p$. In CSR, we (only) contract *replacing redexes*: t μ -rewrites to s , written $t \hookrightarrow_\mu s$ (or $t \hookrightarrow_{\mathcal{R},\mu} s$), iff there are $\ell \rightarrow r \in R$, $p \in \mathcal{P}os^\mu(t)$ and a substitution σ with $t|_p = \sigma(\ell)$ and $s = t[\sigma(r)]_p$. We say that a variable x is *migrating* in $\ell \rightarrow r \in R$ if $x \in \mathcal{V}ar^\mu(r) \setminus \mathcal{V}ar^\mu(\ell)$. A TRS \mathcal{R} is μ -*terminating* if \hookrightarrow_μ is terminating. A term t is μ -*terminating* if there is no infinite μ -rewrite sequence $t = t_1 \hookrightarrow_\mu t_2 \hookrightarrow_\mu \dots \hookrightarrow_\mu t_n \hookrightarrow_\mu \dots$ starting from t . A pair (\mathcal{R}, μ) where \mathcal{R} is a TRS and $\mu \in M_{\mathcal{R}}$ is often called a *CS-TRS*. We say that $f \in \mathcal{F}$ is a *hidden symbol* in $\ell \rightarrow r \in R$ if there exists a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ s.t. $r \triangleright_\mu^\# t$ and $root(t) = f$. We denote $\mathcal{H}(\mathcal{R}, \mu)$ (or just \mathcal{H}) the set of all hidden symbols in (\mathcal{R}, μ) [5].

3 Infinite μ -Rewrite Sequences

For every non- μ -terminating term t we can extract a *minimal infinite μ -rewrite sequence* of the form $t_1 \hookrightarrow_{\mathcal{R},\mu} t'_2 \triangleright_\mu t_2 \hookrightarrow_{\mathcal{R},\mu} t'_3 \triangleright_\mu t_3 \hookrightarrow_{\mathcal{R},\mu} \dots$ where every t_i , $i \geq 2$, is a *minimal non- μ -terminating term* (all its active strict subterms are μ -terminating); this is denoted by $t_i \in \mathcal{M}_{\infty,\mu}$ and t_1 (which is a subterm of t , i.e., $t \triangleright t_1$) is a *strongly minimal non- μ -terminating term* (all its strict subterms are μ -terminating); denoted $t \in \mathcal{T}_{\infty,\mu}$ [5].

As shown in Theorem 3.2 below, when a minimal infinite μ -rewrite sequence is considered, the minimal non- μ -terminating terms $u \in \mathcal{M}_{\infty,\mu}$, which are introduced by the instantiated migrating variables x of a rule $\ell \rightarrow r$ (which are frozen in r but active in ℓ , i.e., $x \in \mathcal{V}ar^\mu(r) \setminus \mathcal{V}ar^\mu(\ell)$) are instances of terms occurring in frozen positions in the right-hand sides of the rules (*hidden terms*) within a given (*hiding*) context. A term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{X}$ is a *hidden term* [1,5] if there is a rule $\ell \rightarrow r \in R$ such that t is a frozen subterm of r . In the following, $\mathcal{HT}(\mathcal{R}, \mu)$ (or just \mathcal{HT} , if \mathcal{R} and μ are clear for the context) is the set of all hidden terms in (\mathcal{R}, μ) and $\mathcal{NHT}(\mathcal{R}, \mu)$ the set of narrowable⁴ hidden terms headed by a defined symbol. A function symbol f *hides position* i if $f(r_1, \dots, r_i, \dots, r_n) \in \mathcal{HT}(\mathcal{R}, \mu)$, $i \in \mu(f)$, and r_i contains a defined symbol or a variable at an active position (i.e., $\mathcal{P}os_D^\mu(r_i) \cup \mathcal{P}os_X^\mu(r_i) \neq \emptyset$). A context $C[\square]$ is *hiding* [1] if $C[\square] = \square$ or $C[\square] = f(t_1, \dots, t_{i-1}, C'[\square], t_{i+1}, \dots, t_n)$, where f hides position i and $C'[\square]$ is a hiding context.

Example 3.1 The hidden terms in Example 1.1 are $\text{minus}(\mathfrak{p}(x), \mathfrak{p}(y))$, $\mathfrak{p}(x)$ and $\mathfrak{p}(y)$. Symbol minus hides positions 1 and 2, and \mathfrak{p} hides position 1.

These notions are used and combined to model infinite context-sensitive rewrite sequences starting from strongly minimal non- μ -terminating terms as follows.

Theorem 3.2 (Minimal sequence [14]) *Let $\mathcal{R} = (\mathcal{F}, R)$ be a TRS and $\mu \in M_{\mathcal{F}}$.*

⁴ A term s *narrows* to the term t if there is a nonvariable position $p \in \mathcal{P}os_{\mathcal{F}}(s)$ and a rule $\ell \rightarrow r$ such that $s|_p$ and ℓ unify with mgu σ , and $t = \sigma(s[r]_p)$.

For all $t \in \mathcal{T}_{\infty, \mu}$, there is an infinite sequence

$$t = t_0 \xrightarrow{>\Lambda^*_{\mathcal{R}, \mu}} \sigma_1(\ell_1) \xrightarrow{\Lambda_{\mathcal{R}, \mu}} \sigma_1(r_1) \triangleright_{\mu} t_1 \xrightarrow{>\Lambda^*_{\mathcal{R}, \mu}} \sigma_2(\ell_2) \xrightarrow{\Lambda_{\mathcal{R}, \mu}} \sigma_2(r_2) \triangleright_{\mu} t_2 \xrightarrow{>\Lambda^*_{\mathcal{R}, \mu}} \dots$$

where, for all $i \geq 1$, $\ell_i \rightarrow r_i \in R$ are rewrite rules, σ_i are substitutions, and terms $t_i \in \mathcal{M}_{\infty, \mu}$ are minimal non- μ -terminating terms such that either

- (i) $t_i = \sigma_i(s_i)$ for some $s_i \notin \mathcal{X}$ such that $r_i \triangleright_{\mu} s_i$ and $\text{root}(s_i) \in \mathcal{D}$, or
- (ii) $\sigma_i(x_i) = C_i[t_i]$ for some $x_i \in \text{Var}^{\mu}(r_i) \setminus \text{Var}^{\mu}(\ell_i)$ and $C_i[t_i] = \theta_i(C'_i[t'_i])$ for some $t'_i \in \mathcal{NHT}(\mathcal{R}, \mu)$, hiding context $C'_i[\square]$, and substitution θ_i .

4 Context-Sensitive Dependency Pairs

The following definitions naturally follow from the facts which have been established in Theorem 3.2. We use the following functions [3,5]: $\text{REN}^{\mu}(t)$, which *independently* renames all *occurrences* of μ -replacing variables by using new fresh variables which are not in $\text{Var}(t)$, and $\text{NARR}_{\mathcal{R}}^{\mu}(t)$, which indicates that t is μ -narrowable (w.r.t. the intended TRS \mathcal{R}).

Definition 4.1 [Context-sensitive dependency pairs [5]] Let $\mathcal{R} = (\mathcal{F}, R) = (\mathcal{C} \uplus \mathcal{D}, R)$ be a TRS and $\mu \in M_{\mathcal{F}}$. We define $\text{DP}(\mathcal{R}, \mu) = \text{DP}_{\mathcal{F}}(\mathcal{R}, \mu) \cup \text{DP}_{\mathcal{X}}(\mathcal{R}, \mu)$ to be set of *context-sensitive dependency pairs* (CS-DPs) where:

$$\begin{aligned} \text{DP}_{\mathcal{F}}(\mathcal{R}, \mu) &= \{\ell^{\sharp} \rightarrow s^{\sharp} \mid \ell \rightarrow r \in R, r \triangleright_{\mu} s, \text{root}(s) \in \mathcal{D}, \ell \not\triangleright_{\mu} s, \text{NARR}_{\mathcal{R}}^{\mu}(\text{REN}^{\mu}(s))\} \\ \text{DP}_{\mathcal{X}}(\mathcal{R}, \mu) &= \{\ell^{\sharp} \rightarrow x \mid \ell \rightarrow r \in R, x \in \text{Var}^{\mu}(r) \setminus \text{Var}^{\mu}(\ell)\} \end{aligned}$$

We extend $\mu \in M_{\mathcal{F}}$ into $\mu^{\sharp} \in M_{\mathcal{F} \cup \mathcal{D}^{\sharp}}$ by $\mu^{\sharp}(f) = \mu(f)$ if $f \in \mathcal{F}$ and $\mu^{\sharp}(f^{\sharp}) = \mu(f)$ if $f \in \mathcal{D}$.

Now, we have to provide a suitable notion of *chain* of CS-DPs. In contrast to [1], we store the information about hidden terms and hiding contexts as a new TRS instead of introducing them as new (transformed) pairs.

Definition 4.2 [Unhiding TRS] Let $\mathcal{R} = (\mathcal{F}, R)$ be a TRS and $\mu \in M_{\mathcal{F}}$. We define $\text{unh}(\mathcal{R}, \mu)$ as the TRS formed by the rules:

- $f(x_1, \dots, x_i, \dots, x_n) \rightarrow x_i$ for every function symbol f of arity n and every $1 \leq i \leq n$ where f hides position i , and
- $t \rightarrow t^{\sharp}$ for every $t \in \mathcal{NHT}(\mathcal{R}, \mu)$.

Example 4.3 The unhiding TRS $\text{unh}(\mathcal{R}, \mu)$ for \mathcal{R} and μ in Example 1.1 is:

$$\begin{array}{llll} \text{minus}(p(x), p(y)) \rightarrow M(p(x), p(y)) & (16) & p(x) \rightarrow x & (18) \\ p(x) \rightarrow P(x) & (17) & \text{minus}(x, y) \rightarrow x & (19) & \text{minus}(x, y) \rightarrow y & (20) \end{array}$$

Definitions 4.1 and 4.2 lead to a suitable notion of *chain* which captures minimal infinite μ -rewrite sequences according to the description in Theorem 3.2.

Definition 4.4 [\mathcal{S} -chain of pairs - minimal \mathcal{S} -chain] Let $\mathcal{R} = (\mathcal{F}, R)$ and $\mathcal{P} = (\mathcal{G}, P)$ be TRSs and $\mu \in M_{\mathcal{F} \cup \mathcal{G}}$. Let $(\mathcal{S}, \mu) = (\mathcal{S}_{\triangleright_{\mu}} \cup \mathcal{S}_{\sharp}, \mu)$, where $\mathcal{S}_{\triangleright_{\mu}}$ are rules of the

form $f(x_1, \dots, x_i, \dots, x_n) \rightarrow x_i \in \mathcal{S}$ for some $f \in \mathcal{F}$ and $i \in \mu(f)$; and $\mathcal{S}_\# = \mathcal{S} \setminus \mathcal{S}_{\triangleright\mu}$. A $(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ -chain is a finite or infinite sequence of pairs $u_i \rightarrow v_i \in \mathcal{P}$, together with a substitution σ satisfying that, for all $i \geq 1$,

- (i) if $v_i \notin \text{Var}(u_i) \setminus \text{Var}^\mu(u_i)$, then $\sigma(v_i) = s_i \xrightarrow{*}_{\mathcal{R}, \mu} \sigma(u_{i+1})$, and
- (ii) if $v_i \in \text{Var}(u_i) \setminus \text{Var}^\mu(u_i)$, then $\sigma(v_i) \xrightarrow{\Delta}_{\mathcal{S}_{\triangleright\mu}, \mu}^* \circ \xrightarrow{\Delta}_{\mathcal{S}_\#, \mu} s_i \xrightarrow{*}_{\mathcal{R}, \mu} \sigma(u_{i+1})$.

A $(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ -chain is called *minimal* if for all $i \geq 1$ s_i is (\mathcal{R}, μ) -terminating.

Notice that if $\mathcal{S}_{\triangleright\mu} = \{f(x_1, \dots, x_n) \rightarrow x_i \mid f \in \mathcal{D}, i \in \mu(f)\}$, then we have the notion of chain in [5] and if $\mathcal{S}_\# = \{f(x_1, \dots, x_n) \rightarrow f^\#(x_1, \dots, x_n) \mid f \in \mathcal{D}\}$, then we have the original notion of chain from [3].

Theorem 4.5 (Soundness and completeness of CS-DPs) *Let \mathcal{R} be a TRS and $\mu \in M_{\mathcal{R}}$. A CS-TRS (\mathcal{R}, μ) is terminating if and only if there is no infinite $(\text{DP}(\mathcal{R}, \mu), \mathcal{R}, \text{unh}(\mathcal{R}, \mu), \mu)$ -chain.*

5 Context-Sensitive Dependency Pairs Framework

In the DP framework [13] the focus is on the so-called *termination problems* involving two TRSs \mathcal{P} and \mathcal{R} instead of just the ‘target’ TRS \mathcal{R} . In our setting we start with the following definition (see also [1,5]).

Definition 5.1 [CS-termination problem and processor] A *CS-termination problem* τ is a tuple $\tau = (\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$, where $\mathcal{R} = (\mathcal{F}, R)$, $\mathcal{P} = (\mathcal{G}, P)$ and $\mathcal{S} = (\mathcal{F} \cup \mathcal{G}, S)$ are TRSs, and $\mu \in M_{\mathcal{F} \cup \mathcal{G}}$. A *CS-termination problem* $(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ is *finite* if there is no infinite $(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ -chain. A *CS-processor* Proc is a mapping from CS-termination problems into sets of CS-termination problems. A CS-processor Proc is *sound* if for all CS-termination problems τ , τ is finite whenever τ' is finite for all $\tau' \in \text{Proc}(\tau)$. A CS-processor Proc is *complete* if for all CS-termination problems τ , whenever τ is finite, τ' is finite for all $\tau' \in \text{Proc}(\tau)$.

In order to prove the μ -termination of a TRS \mathcal{R} , we start with the CS-termination problem $(\text{DP}(\mathcal{R}, \mu), \mathcal{R}, \text{unh}(\mathcal{R}, \mu), \mu)$ and we repeatedly apply *correct* CS-processors until no further CS-processor applies. Then, the μ -termination of \mathcal{R} is proved. The pairs in a CS-TRS (\mathcal{P}, μ) , where $\mathcal{P} = (\mathcal{G}, P)$, are partitioned as follows: $P_{\mathcal{X}} = \{u \rightarrow v \in P \mid v \in \text{Var}(u) \setminus \text{Var}^\mu(u)\}$ and $P_{\mathcal{G}} = P \setminus P_{\mathcal{X}}$. Definition 5.1 enables the use of all existing CS-processors (see [1,5]). For instance, the following processor integrates the transformation of [1] into our framework.

Theorem 5.2 (Collapsing pairs transformation) *Let $\tau = (\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ be a CS-termination problem and P_{U} be given by the following rules:*

- $u \rightarrow \text{U}(x)$ for every $u \rightarrow x \in P_{\mathcal{X}}$,
- $\text{U}(f(x_1, \dots, x_i, \dots, x_n)) \rightarrow \text{U}(x_i)$ for every $f(x_1, \dots, x_i, \dots, x_n) \rightarrow x_i \in \mathcal{S}_{\triangleright\mu}$, and
- $\text{U}(s) \rightarrow t$ for every $s \rightarrow t \in \mathcal{S}_\#$.

where U is a new fresh symbol. Let $\mathcal{P}' = (\mathcal{G} \cup \{\text{U}\}, P')$ where $P' = (P \setminus P_{\mathcal{X}}) \cup P_{\text{U}}$, and μ' extends μ by $\mu'(\text{U}) = \emptyset$. The processor $\text{Proc}_{e\text{Coll}}$ given by $\text{Proc}_{e\text{Coll}}(\tau) = \{(\mathcal{P}', \mathcal{R}, \emptyset, \mu')\}$ is sound and complete.

Now, we can apply all CS-processors from [1] and [5] which did not consider any \mathcal{S} component in termination problems. In the following sections, we describe some important CS-processors within our framework.

5.1 Context-Sensitive Dependency Graph

In the DP-approach [6,13], a *dependency graph* is associated to the TRS \mathcal{R} . The nodes of the graph are the dependency pairs in $\text{DP}(\mathcal{R})$ and there is an arc from a dependency pair $u \rightarrow v$ to a dependency pair $u' \rightarrow v'$ if there are substitutions θ and θ' such that $\theta(v) \rightarrow_{\mathcal{R}}^* \theta'(u')$. In our setting, we have the following.

Definition 5.3 [CS-graph of pairs] Let $\mathcal{R} = (\mathcal{F}, R)$, $\mathcal{P} = (\mathcal{G}, P)$ and $\mathcal{S} = (\mathcal{F} \cup \mathcal{G}, S)$ be TRSs and $\mu \in M_{\mathcal{F} \cup \mathcal{G}}$. The *CS-graph* associated to \mathcal{R} , \mathcal{P} and \mathcal{S} (denoted $\text{G}(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$) has \mathcal{P} as the set of nodes and arcs which connect them as follows:

- (i) there is an arc from $u \rightarrow v \in \mathcal{P}_{\mathcal{G}}$ to $u' \rightarrow v' \in \mathcal{P}$ if there are substitutions θ and θ' such that $\theta(v) \hookrightarrow_{\mathcal{R}, \mu}^* \theta'(u')$, and
- (ii) there is an arc from $u \rightarrow v \in \mathcal{P}_{\mathcal{X}}$ to $u' \rightarrow v' \in \mathcal{P}$ if there are substitutions θ and θ' , and a term s such that $\theta(v) \xrightarrow{\Delta}^*_{\mathcal{S}_{\triangleright \mu}, \mu} \circ \xrightarrow{\Delta}_{\mathcal{S}_{\sharp, \mu}} s \hookrightarrow_{\mathcal{R}, \mu}^* \theta'(u')$.

Example 5.4 In Figure 1 (right) we show $\text{G}(\mathcal{R}, \text{DP}(\mathcal{R}, \mu), \text{unh}(\mathcal{R}, \mu), \mu)$ for \mathcal{R} in Example 1.1.

In termination proofs, we are concerned with the so-called *strongly connected components* (SCCs) of the dependency graph, rather than with the cycles themselves (which are exponentially many) [16]. The following result formalizes the use of SCCs for dealing with CS-termination problems.

Theorem 5.5 (SCC processor) Let $\tau = (\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ be a CS-termination problem. The CS-processor Proc_{SCC} given by

$$\text{Proc}_{\text{SCC}}(\tau) = \{(\mathcal{Q}, \mathcal{R}, \mathcal{S}_{\mathcal{Q}}, \mu) \mid \mathcal{Q} \text{ contains the pairs of an SCC in } \text{G}(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)\}$$

where $\mathcal{S}_{\mathcal{Q}} = \mathcal{S}_{\triangleright \mu} \cup \{s \rightarrow t \mid s \rightarrow t \in \mathcal{S}_{\sharp, \text{NARR}}^{\mu}(\text{REN}^{\mu}(t))\}$ is sound and complete.

Example 5.6 The graph in Figure 1 (right) has three SCCs $\mathcal{P}_1 = \{(1)\}$, $\mathcal{P}_2 = \{(8)\}$, and $\mathcal{P}_3 = \{(7), (14), (15)\}$. If we apply the CS-processor Proc_{SCC} to the initial CS-termination problem $(\text{DP}(\mathcal{R}, \mu), \mathcal{R}, \text{unh}(\mathcal{R}, \mu), \mu)$ for (\mathcal{R}, μ) in Example 1.1 we obtain problems $(\mathcal{P}_1, \mathcal{R}, \text{unh}(\mathcal{R}, \mu), \mu)$, $(\mathcal{P}_2, \mathcal{R}, \text{unh}(\mathcal{R}, \mu), \mu)$, $(\mathcal{P}_3, \mathcal{R}, \text{unh}(\mathcal{R}, \mu), \mu)$.

The CS-graph is not computable. Thus, we have to use an over-approximation to it. In the following definition, besides REN^{μ} we use a new function $\text{CAP}_{\Delta}^{\mu}(t)$, which renames all subterms headed by a ‘defined’ symbol in Δ by new fresh variables.

Definition 5.7 [Estimated CS-graph of pairs] Let $\mathcal{R} = (\mathcal{F}, R)$, $\mathcal{P} = (\mathcal{G}, P)$ and $\mathcal{S} = (\mathcal{F} \cup \mathcal{G}, S)$ be TRSs and $\mu \in M_{\mathcal{F} \cup \mathcal{G}}$. The *estimated CS-graph* associated to \mathcal{R} , \mathcal{P} and \mathcal{S} (denoted $\text{EG}(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$) has \mathcal{P} as the set of nodes and arcs which connect them as follows:

- (i) there is an arc from $u \rightarrow v \in \mathcal{P}_{\mathcal{G}}$ to $u' \rightarrow v' \in \mathcal{P}$ if $\text{REN}^{\mu}(\text{CAP}_{\mathcal{D}}^{\mu}(v))$ and u' unify, and

- (ii) there is an arc from $u \rightarrow v \in \mathcal{P}_{\mathcal{X}}$ to $u' \rightarrow v' \in \mathcal{P}$ if there is $\ell \rightarrow r \in \mathcal{S}_{\#}$ such that $\text{REN}^{\mu}(\text{CAP}_{\mathcal{D}}^{\mu}(r))$ and u' unify.

Example 5.8 In Example 1.1, the estimated context-sensitive dependency graph and the context-sensitive dependency graph are the same.

5.2 Reduction Pair Processor

A μ -reduction pair (\succsim, \sqsupset) consists of a stable and μ -monotonic quasi-ordering \succsim , and a well-founded stable relation \sqsupset on terms in $\mathcal{T}(\mathcal{F}, \mathcal{X})$ which are compatible, i.e., $\succsim \circ \sqsupset \subseteq \sqsupset$ or $\sqsupset \circ \succsim \subseteq \sqsupset$ [3].

In [3], when a collapsing dependency pair $u \rightarrow x$ occurs in a chain we have to *look inside* the instantiated right-hand side $\sigma(x)$ for a μ -replacing subterm that, after marking it, does rewrite to the (instantiated) left-hand side of another dependency pair. For this reason, the quasi-ordering \succsim of our reduction pair (\succsim, \sqsupset) is required to have the μ -subterm property, i.e. $\triangleright_{\mu} \subseteq \succsim$. This is similar for markings: we have to ensure that $f(x_1, \dots, x_n) \succsim f^{\#}(x_1, \dots, x_n)$ for all defined symbols f in the signature. In [5], thanks to the notion of hidden term, we can relax the last condition to be held on hidden terms only, that is we require $t \succsim t^{\#}$ for all (narrowable) hidden terms t . However, we kept the quasi-ordering \succsim compatible with \triangleright_{μ} . In [1], thanks to the notion of hiding context, we only require that \succsim is compatible with the projections $f(x_1, \dots, x_k) \rightarrow x_i$ for those symbols f and positions i such that f hides position i . However, this information is implicitly encoded as (new) pairs in the set \mathcal{P} . The strict component \sqsupset of the reduction pair (\succsim, \sqsupset) is used with these new pairs now.

In this paper, since the rules in \mathcal{S} are not considered as ordinary pairs (in the sense of [1,5]) we can relax the conditions imposed to the orderings dealing with these rules. Since rules in \mathcal{S} are applied to the root of the term, we only have to impose stability to the ordering which is compatible with these rules (no well-foundedness or μ -monotonicity required).

Therefore, we can use now μ -reduction triples $(\succsim, \sqsupset, \succeq)$ where (\succsim, \sqsupset) is a μ -reduction pair and \succeq is a stable quasi-ordering which is compatible with \succsim , i.e., $\succeq \circ \succsim \subseteq \succeq$ or $\succsim \circ \succeq \subseteq \succeq$.

Theorem 5.9 (μ -reduction triple processor) *Let $\tau = (\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ be a CS-termination problem. Let $(\succsim, \sqsupset, \succeq)$ be a μ -reduction triple such that*

- (i) $\mathcal{P} \subseteq \succsim \cup \sqsupset$, $\mathcal{R} \subseteq \succsim$, and
- (ii) whenever $\mathcal{P}_{\mathcal{X}} \neq \emptyset$ we have that $\mathcal{S} \subseteq \succsim \cup \sqsupset \cup \succeq$.

Let $\mathcal{P}_{\sqsupset} = \{u \rightarrow v \in \mathcal{P} \mid u \sqsupset v\}$. Then, the processor Proc_{RT} given by

$$\text{Proc}_{RT}(\tau) = \begin{cases} \{(\mathcal{P} \setminus \mathcal{P}_{\sqsupset}, \mathcal{R}, \mathcal{S}, \mu)\} & \text{if (i) and (ii) hold} \\ \{(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)\} & \text{otherwise} \end{cases}$$

is sound and complete.

Since rules from \mathcal{S} are only applied when collapsing pairs appear, we only need to make them compatible with some ordering if condition (ii) holds. Another advantage is that we can use μ -reduction pairs to remove rules from \mathcal{S} .

Theorem 5.10 (μ -reduction pair processor) *Let $d = (\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ be a CS-termination problem. Let (\succ, \sqsupset) be a μ -reduction pair such that*

- (i) $\mathcal{P} \subseteq \succ \cup \sqsupset$, $\mathcal{R} \subseteq \succ$, and
- (ii) whenever $\mathcal{P}_{\mathcal{X}} \neq \emptyset$ we have that $\mathcal{S} \subseteq \succ \cup \sqsupset$.

Let $\mathcal{P}_{\sqsupset} = \{u \rightarrow v \in \mathcal{P} \mid u \sqsupset v\}$ and $\mathcal{S}_{\sqsupset} = \{u \rightarrow v \in \mathcal{S} \mid u \sqsupset v\}$. Then, the processor Proc_{RP} given by

$$\text{Proc}_{RP}(d) = \begin{cases} \{(\mathcal{P} \setminus \mathcal{P}_{\sqsupset}, \mathcal{R}, \mathcal{S} \setminus \mathcal{S}_{\sqsupset}, \mu)\} & \text{if (i) and (ii) hold} \\ \{(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)\} & \text{otherwise} \end{cases}$$

is sound and complete.

Furthermore, we can increase the power of this definition by considering the *usable rules* corresponding to \mathcal{P} instead of \mathcal{R} as a whole (see [1,15]).

5.3 Collapsing Dependency Pairs Processors

With the new definition, we can apply specific processors for collapsing pairs that are very useful, but that do not apply if we do not have them in our chains (as in [1]). For instance, we can use the following result, which is often used in proofs of termination of CSR with MU-TERM [19,2]. The subTRS of $\mathcal{P}_{\mathcal{X}}$ containing the rules whose migrating variables occur on non- μ -replacing immediate subterms in the left-hand side is $\mathcal{P}_{\mathcal{X}}^1 = \{f(u_1, \dots, u_k) \rightarrow x \in \mathcal{P}_{\mathcal{X}} \mid \exists i, 1 \leq i \leq k, i \notin \mu(f), x \in \text{Var}(u_i)\}$.

Theorem 5.11 [5] *Let $\tau = (\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ be a CS-termination problem. If $\mathcal{P} = \mathcal{P}_{\mathcal{X}}^1$ then τ is finite.*

And we can also define new processors. The following processor can be applied after processor from Theorem 5.10 if we have removed all the rules in $\mathcal{S}_{\#}$.

Theorem 5.12 (Marking rules processor) *Let $\tau = (\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)$ be a CS-termination problem. Then, the processor $\text{Proc}_{\#}$ given by*

$$\text{Proc}_{\#}(\tau) = \begin{cases} \{(\mathcal{P} \setminus \mathcal{P}_{\mathcal{X}}, \mathcal{R}, \emptyset, \mu)\} & \text{if } \mathcal{S}_{\#} = \emptyset \text{ hold} \\ \{(\mathcal{P}, \mathcal{R}, \mathcal{S}, \mu)\} & \text{otherwise} \end{cases}$$

is sound and complete.

6 Conclusions

When proofs of termination of CSR are mechanized following the context-sensitive dependency pairs approach [3], handling collapsing pairs is difficult. In [1] this problem is solved by a transformation which disregards collapsing pairs (so we loose their descriptive power), adds a new fresh symbol U which has nothing to do with the original CS-TRS, and makes the dependency graph harder to understand.

We have shown a different way to mechanize the context-sensitive dependency pairs approach. The idea is adding a new TRS, the *unhiding TRS*, which avoids the

extra requirements in [3]. Thanks to the flexibility of our framework, we can use all existing processors in the literature, improve the existing ones by taking advance of having collapsing pairs, and define new processors.

References

- [1] B. Alarcón, F. Emmes, C. Fuhs, J. Giesl, R. Gutiérrez, S. Lucas, P. Schneider-Kamp and R. Thiemann, *Improving Context-Sensitive Dependency Pairs*, in: I. Cervesato, H. Veith and A. Voronkov, editors, *Proc. of 15th International Conference on Logic for Programming, Artificial Intelligence and Reasoning, LPAR'08*, Lecture Notes in Computer Science **5330** (2008), pp. 636–651.
- [2] B. Alarcón, R. Gutiérrez, J. Iborra and S. Lucas, *Proving Termination of Context-Sensitive Rewriting with MU-TERM*, *Electronic Notes in Theoretical Computer Science* **188** (2007), pp. 105–115.
- [3] B. Alarcón, R. Gutiérrez and S. Lucas, *Context-Sensitive Dependency Pairs*, in: S. Arun-Kumar and N. Garg, editors, *Proc. of 26th Conference on Foundations of Software Technology and Theoretical Computer Science, FST&TCS'06*, Lecture Notes in Computer Science **4337** (2006), pp. 297–308.
- [4] B. Alarcón, R. Gutiérrez and S. Lucas, *Improving the Context-Sensitive Dependency Graph*, *Electronic Notes in Theoretical Computer Science* **188** (2007), pp. 91–103.
- [5] B. Alarcón, R. Gutiérrez and S. Lucas, *Context-sensitive dependency pairs*, Technical report, Universidad Politécnica de Valencia (2008), available as Technical Report DSIC-II/10/08.
- [6] T. Arts and J. Giesl, *Termination of Term Rewriting Using Dependency Pairs*, *Theoretical Computer Science* **236** (2000), pp. 133–178.
- [7] F. Baader and T. Nipkow, *Term Rewriting and All That*, Cambridge University Press, 1998.
- [8] R. Bruni and J. Meseguer, *Semantic Foundations for Generalized Rewrite Theories*, *Theoretical Computer Science* **360** (2006), pp. 386–414.
- [9] F. Durán, S. Lucas, C. Marché, J. Meseguer and X. Urbain, *Proving Operational Termination of Membership Equational Programs*, *Higher Order Symbolic Computation* **21** (2008), pp. 59–88.
- [10] J. Endrullis and D. Hendriks, *From Outermost to Context-Sensitive Rewriting*, in: R. Treinen, editor, *Proc. of 20th International Conference on Rewriting Techniques and Applications, RTA'09*, Lecture Notes in Computer Science **5595** (2009), pp. 305–319.
- [11] M. L. Fernández, *Relaxing Monotonicity for Innermost Termination*, *Information Processing Letters* **93** (2005), pp. 117–123.
- [12] J. Giesl and A. Middeldorp, *Transformation Techniques for Context-Sensitive Rewrite Systems*, *Journal of Functional Programming* **14** (2004), pp. 379–427.
- [13] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. *Mechanizing and Improving Dependency Pairs*. *Journal of Automatic Reasoning* **37(3)** (2006), pp. 155–203.
- [14] R. Gutiérrez, *Context-Sensitive Dependency Pairs Framework*, Master's thesis, Departamento de Sistemas Informáticos y Computación, Universitat Politècnica de València, València, Spain (2008).
- [15] R. Gutiérrez, S. Lucas and X. Urbain, *Usable Rules for Context-Sensitive Rewrite Systems*, in: A. Voronkov, editor, *Proc. of 19th International Conference on Rewriting Techniques and Applications, RTA'08*, Lecture Notes in Computer Science **5117** (2008), pp. 126–141.
- [16] N. Hirokawa and A. Middeldorp, *Automating the Dependency Pair Method*, *Information and Computation* **199** (2005), pp. 172–199.
- [17] S. Lucas, *Context-Sensitive Computations in Functional and Functional Logic Programs*, *Journal of Functional and Logic Programming* **1998** (1998), pp. 1–61.
- [18] S. Lucas, *Termination of On-Demand Rewriting and Termination of OBJ Programs*, in: R. De Nicola and H. Søndergaard, editors, *Proc. of 3rd International Conference on Principles and Practice of Declarative Programming, PPDP'01* (2001), ACM Press, pp. 82–93.
- [19] S. Lucas, *MU-TERM: A Tool for Proving Termination of Context-Sensitive Rewriting*, in: V. v. Oostrom, editor, *Proc. of 15th International Conference on Rewriting Techniques and Applications, RTA'04*, Lecture Notes in Computer Science **3091** (2004), pp. 200–209, available at <http://zenon.dsic.upv.es/muterm/>.