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

Uncurrying for Termination and Complexity

Nao Hirokawa · Aart Middeldorp · Harald Zankl

Received: date / Accepted: date

Abstract First-order applicative rewrite systems provide a natural framework for modeling higher-order aspects. In this article we present a transformation from untyped applicative term rewrite systems to functional term rewrite systems that preserves and reflects termination. Our transformation is less restrictive than other approaches. In particular, head variables in right-hand sides of rewrite rules can be handled. To further increase the applicability of our transformation, we study the method for innermost rewriting and derivational complexity, and present a version for dependency pairs.

Keywords Term rewriting · Termination · Uncurrying

1 Introduction

In this article we are concerned with proving (innermost) termination of first-order applicative term rewrite systems. These systems provide a natural framework for modeling higher-order aspects found in functional programming languages. A prominent example of a first-order applicative system is combinatory

This research is supported by FWF (Austrian Science Fund) project P18763 and the Grant-in-Aid for Young Scientists Nos. 20800022 and 22700009 of the Japan Society for the Promotion of Science.

N. Hirokawa
School of Information Science, Japan Advanced Institute of Science and Technology, Nomi, Japan
E-mail: hirokawa@jaist.ac.jp

A. Middeldorp · H. Zankl
Institute of Computer Science, University of Innsbruck, Innsbruck, Austria
E-mail: aart.middeldorp@uibk.ac.at

H. Zankl
E-mail: harald.zankl@uibk.ac.at

logic. The signature of an applicative term rewrite system consists of constants and a single binary function symbol called application which is denoted by the infix and left-associative symbol $\star$. In term rewriting, properties such as termination and innermost termination are of particular interest since they are essential for many rewriting techniques including equational reasoning and confluence analysis (cf. [39]). Moreover, innermost termination has received a renewed interest in termination analysis of functional programs [11].

Proving termination of applicative term rewrite systems is challenging because the rewrite rules lack sufficient structure. As a consequence, simplification orders are not effective as $\star$ is the only function symbol of non-zero arity. Moreover, the dependency pair method is of little help as $\star$ is the only defined non-constant symbol. To remedy this issue two solutions have been suggested. The first line of research is based on types [1–3, 7, 21, 28, 41] and allows to study properties like termination or strong computability directly. The second approach [13, 18] aims for transformations that recover the structure of applicative rewrite rules to enable methods that do not rely on types. The benefit of the first approach is that the type information may make proving termination properties easier. However, most of those studies are within the realm of simply typed systems and hence miss polymorphism, which is used in many functional programming languages. Moreover, in contrast to untyped rewriting, no powerful automated tools are (yet) available. This is in sharp contrast to the untyped first-order setting where powerful tools exist as witnessed by the annual competition of termination tools.[1]

The main contribution of this article is a new transformation that recovers the structure in applicative rewrite rules, thereby enabling traditional methods for proving termination and innermost termination. Our transformation can deal with partial applications as well as head variables in right-hand sides of rewrite rules. The key ingredients are the η-saturation of rewrite rules (Definition 7) and the addition of sufficiently many uncurrying rules to the transformed system. These rules are also crucial for a smooth transition into the dependency pair framework. Unlike the transformation of applicative dependency pair problems presented in [13, 40], our uncurrying processor preserves minimality (cf. Section 6), which means that it can be used at any node in a modular (non-)termination proof attempt.

We remark that our results for proving innermost termination with the help of uncurrying are directly applicable for functional programming languages that adopt an eager evaluation strategy. Surprisingly, these results are also helpful in the case of lazy evaluation. Since applicative term rewrite systems modeling functional programs are left-linear and non-overlapping, termination and innermost termination coincide (see [15] for a more general result). Hence instead of establishing full termination for lazy languages we investigate innermost termination, which is equivalent in this case but typically easier to establish. In this context it is worth noting the recent work of Giesl *et al.* In [11] they present a two-stage transformation method from (functions in)

[1] `http://termcomp.uibk.ac.at`

Haskell programs to applicative dependency pair problems such that termination of the former is concluded from innermost finiteness of the latter. This approach is capable of automatically proving the termination of most functions in standard Haskell libraries.

The remainder of this article is organised as follows. After recalling preliminaries in Section 2, we present a new uncurrying transformation and prove that it preserves and reflects termination for full rewriting in Section 3. Results for innermost termination and derivational complexity are studied in Sections 4 and 5, respectively. Two extensions to the dependency pair framework are presented in Section 6. How these extensions behave for full termination is the topic of Section 6.1 while Section 6.2 is concerned with their properties concerning innermost termination. Our results are empirically evaluated in Section 7 and we conclude with a discussion of related work in Section 8.

A preliminary version of this article appeared in [18]. Several of the results on innermost rewriting and derivational complexity have been published in [44]. Theorems 6 and 11 are new contributions. So is the material in Section 6.2. Moreover, we close a non-trivial gap in the proof of [18, Theorem 33]. Some of the new contributions go beyond the scope of uncurrying, e.g., Theorem 11 gives a condition when the length of reductions is the same for innermost and full rewriting, which has an immediate impact on (automated) complexity analysis. Another fundamental new result deals with signature extensions, which do not affect termination [29,35] but surprisingly may destroy finiteness of DP problems [37]. Finally, Lemma 19 shows that innermost finiteness is not affected by signature extensions.

2 Preliminaries

In this section we fix preliminaries on rewriting, complexity, dependency pairs, and currying.

2.1 Term Rewriting

We assume familiarity with term rewriting [5] in general and termination [47] in particular. Let $\mathcal{F}$ be a signature and $\mathcal{V}$ be a set of variables disjoint from $\mathcal{F}$. By $\mathcal{T}(\mathcal{F},\mathcal{V})$ we denote the set of terms over $\mathcal{F}$ and $\mathcal{V}$. The *size* of a term t is denoted $|t|$ and the root symbol of t is denoted $\mathrm{root}(t)$. A *rewrite rule* is a pair of terms (ℓ, r), written $\ell \to r$ such that ℓ is not a variable and all variables in r are contained in ℓ. A *term rewrite system* (TRS for short) is a set of rewrite rules. A TRS $\mathcal{R}$ is said to be *duplicating* if there exist a rewrite rule $\ell \to r \in \mathcal{R}$ and a variable x that occurs more often in r than in ℓ. By $\mathcal{F}\mathsf{un}(\mathcal{R})$ we denote the set of function symbols that occur in a TRS $\mathcal{R}$.

Contexts are terms over the signature $\mathcal{F} \cup \{\Box\}$ with exactly one occurrence of the fresh constant $\Box$ (called *hole*). The expression $C[t]$ denotes the result of replacing the hole in C by the term t. A *substitution* σ is a mapping from

variables to terms and $t\sigma$ denotes the result of replacing the variables in t according to σ. Substitutions may change only finitely many variables (and are thus written as $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$). The *set of positions* of a term t is defined as $\mathcal{P}\mathsf{os}(t) = \{\epsilon\}$ if t is a variable and as $\mathcal{P}\mathsf{os}(t) = \{\epsilon\} \cup \{iq \mid q \in \mathcal{P}\mathsf{os}(t_i)\}$ if $t = f(t_1, \ldots, t_n)$. Positions are used to address occurrences of subterms. The *subterm of* t *at position* $p \in \mathcal{P}\mathsf{os}(t)$ is defined as $t|_p = t$ if $p = \epsilon$ and as $t|_p = t_i|_q$ if $p = iq$. We say a position p is *to the right of* a position q if $p = p_1 i p_2$ and $q = q_1 j q_2$ with $p_1 = q_1$ and $i > j$. For a term t and positions $p, q \in \mathcal{P}\mathsf{os}(t)$ we say $t|_p$ is to the right of $t|_q$ if p is to the right of q.

A *rewrite relation* is a binary relation on terms that is closed under contexts and substitutions. For a TRS $\mathcal{R}$ we define $\to_\mathcal{R}$ to be the smallest rewrite relation that contains $\mathcal{R}$. We call $s \to_\mathcal{R} t$ a *rewrite step* if there exist a context C, a rewrite rule $\ell \to r \in \mathcal{R}$, and a substitution σ such that $s = C[\ell\sigma]$ and $t = C[r\sigma]$. In this case we call $\ell\sigma$ a *redex* and say that $\ell\sigma$ has been *contracted*. A *root rewrite step*, denoted by $s \to^\epsilon_\mathcal{R} t$, has the shape $s = \ell\sigma \to_\mathcal{R} r\sigma = t$ for some $\ell \to r \in \mathcal{R}$. A *rewrite sequence* is a sequence of rewrite steps. The *set of normal forms* of a TRS $\mathcal{R}$ is defined as $NF(\mathcal{R}) = \{t \in \mathcal{T}(\mathcal{F}, \mathcal{V}) \mid t \text{ contains no redexes}\}$. A redex $\ell\sigma$ in a term t is called *innermost* if proper subterms of $\ell\sigma$ are normal forms, and *rightmost innermost* if in addition $\ell\sigma$ is to the right of any other innermost redex in t. A rewrite step is called *innermost* (*rightmost innermost*) if an innermost (rightmost innermost) redex is contracted, written $\xrightarrow{\mathsf{i}}$ and $\xrightarrow{\mathsf{ri}}$, respectively.

If the TRS $\mathcal{R}$ is not essential or clear from the context the subscript $\mathcal{R}$ is omitted in $\to_\mathcal{R}$ and its derivatives. As usual, $\to^+$ ($\to^*$) denotes the transitive (reflexive and transitive) closure of $\to$ and $\to^m$ its m-th iterate. A TRS is *terminating* (*innermost terminating*) if $\to^+$ ($\xrightarrow{\mathsf{i}}{}^+$) is well-founded. If $s \to^*_\mathcal{R} t$ and $t \in NF(\mathcal{R})$ then we write $s \to^!_\mathcal{R} t$.

An overlap $(\ell_1 \to r_1, p, \ell_2 \to r_2)_\mu$ of a TRS $\mathcal{R}$ consists of variants $\ell_1 \to r_1$ and $\ell_2 \to r_2$ of rules of $\mathcal{R}$ without common variables, a non-variable position $p \in \mathcal{P}\mathsf{os}(\ell_2)$, and a most general unifier μ of ℓ_1 and $\ell_2|_p$. If $p = \epsilon$ then we require that $\ell_1 \to r_1$ and $\ell_2 \to r_2$ are not variants of the same rewrite rule. A TRS without overlaps is called *non-overlapping*. An *overlay system* is a TRS whose overlaps emerge at root positions only.

Let $\mathcal{P}$ be a property of TRSs and let Φ be a transformation on TRSs with $\Phi(\mathcal{R}) = \mathcal{R}'$. We say Φ *preserves* $\mathcal{P}$ if $\mathcal{P}(\mathcal{R})$ implies $\mathcal{P}(\mathcal{R}')$ and Φ *reflects* $\mathcal{P}$ if $\mathcal{P}(\mathcal{R}')$ implies $\mathcal{P}(\mathcal{R})$. Sometimes we call Φ $\mathcal{P}$ preserving if Φ preserves $\mathcal{P}$ and $\mathcal{P}$ reflecting if Φ reflects $\mathcal{P}$, respectively.

2.2 Derivational Complexity

For complexity analysis we assume TRSs to be finite and (innermost) terminating. Hofbauer and Lautemann [20] introduced the concept of derivational complexity for terminating TRSs. The idea is to measure the maximal length of rewrite sequences (derivations) depending on the size of the starting term. Formally, the *derivation height* of a term t (with respect to a

finitely branching and well-founded relation $\to$) is defined on natural numbers as $\mathrm{dh}(t, \to) = \max\{m \in \mathbb{N} \mid t \to^m u \text{ for some } u\}$. The *derivational complexity* $\mathrm{dc}_{\mathcal{R}}(n)$ of a TRS $\mathcal{R}$ is then defined as

$$\mathrm{dc}_{\mathcal{R}}(n) = \max\{\mathrm{dh}(t, \to_{\mathcal{R}}) \mid |t| \leqslant n\}$$

Similarly we define the *innermost* derivational complexity as

$$\mathrm{idc}_{\mathcal{R}}(n) = \max\{\mathrm{dh}(t, \xrightarrow{\mathsf{i}}_{\mathcal{R}}) \mid |t| \leqslant n\}$$

Since we regard finite TRSs only, these functions are well-defined if $\mathcal{R}$ is (innermost) terminating. If $\mathrm{dc}_{\mathcal{R}}(n)$ is bounded from above by a linear, quadratic, cubic, ... function or polynomial, $\mathcal{R}$ is said to have linear, quadratic, cubic, ... or polynomial derivational complexity. A similar convention applies to $\mathrm{idc}_{\mathcal{R}}(n)$.

For functions $f, g \colon \mathbb{N} \to \mathbb{N}$ we write $f(n) \in \mathcal{O}(g(n))$ if there are constants $c, N \in \mathbb{N}$ such that $f(n) \leqslant c \cdot g(n)$ for all $n \geqslant N$.

One popular method to prove polynomial upper bounds on the derivational complexity is via triangular matrix interpretations [33], which are a special instance of monotone algebras. For a signature $\mathcal{F}$, an *$\mathcal{F}$-algebra* $\mathcal{A}$ consists of a non-empty carrier A and a set of interpretations $f_{\mathcal{A}}$ for every $f \in \mathcal{F}$. By $[\alpha]_{\mathcal{A}}(\cdot)$ we denote the usual evaluation function of $\mathcal{A}$ according to an assignment α which maps variables to values in A. An $\mathcal{F}$-algebra $\mathcal{A}$ together with a well-founded order $\succ$ on A is called a *monotone algebra* if every $f_{\mathcal{A}}$ is monotone with respect to $\succ$. Any monotone algebra $(\mathcal{A}, \succ)$ induces a well-founded order on terms: $s \succ_{\mathcal{A}} t$ if for any assignment α the condition $[\alpha]_{\mathcal{A}}(s) \succ [\alpha]_{\mathcal{A}}(t)$ holds. A TRS $\mathcal{R}$ is *compatible* with a monotone algebra $(\mathcal{A}, \succ_{\mathcal{A}})$ if $\ell \succ_{\mathcal{A}} r$ for every $\ell \to r \in \mathcal{R}$.

Matrix interpretations $(\mathcal{M}, \succ)$ (often just denoted $\mathcal{M}$) are a special form of monotone algebras. Here the carrier is $\mathbb{N}^d$ for some fixed dimension $d \in \mathbb{N} \setminus \{0\}$. The order $\succ$ is defined on $\mathbb{N}^d$ as $(u_1, \dots, u_d)^{\mathrm{T}} \succ (v_1, \dots, v_d)^{\mathrm{T}}$ if $u_1 >_{\mathbb{N}} v_1$ and $u_i \geqslant_{\mathbb{N}} v_i$ for all $2 \leqslant i \leqslant d$. If every $f \in \mathcal{F}$ of arity n is interpreted as

$$f_{\mathcal{M}}(\overrightarrow{x_1}, \dots, \overrightarrow{x_n}) = F_1\overrightarrow{x_1} + \cdots + F_n\overrightarrow{x_n} + \overrightarrow{f}$$

where $F_i \in \mathbb{N}^{d \times d}$ for all $1 \leqslant i \leqslant n$ and $\overrightarrow{f} \in \mathbb{N}^d$ then monotonicity of $\succ$ is achieved by demanding that the top left entry of every matrix F_i is non-zero. Such interpretations have been introduced in [10].

A square matrix A of dimension d is of *upper triangular* shape if $A_{(i,i)} \leqslant 1$ and $A_{(i,j)} = 0$ if $i > j$ for all $1 \leqslant i, j \leqslant d$. A matrix interpretation where for every $f \in \mathcal{F}$ all F_i ($1 \leqslant i \leqslant n$ where n is the arity of f) are upper triangular is called *triangular* (abbreviated by TMI). The next theorem is from [33].

Theorem 1 *If a TRS $\mathcal{R}$ is compatible with a TMI of dimension d then $\mathrm{dc}_{\mathcal{R}}(n) \in \mathcal{O}(n^d)$.* □

Recent generalisations of this result are reported in [31, 34, 42].

2.3 Dependency Pairs

Let $\mathcal{R}$ be a TRS over a signature $\mathcal{F}$. The signature $\mathcal{F}$ is extended with *dependency pair symbols* $f^\sharp$ for every symbol $f \in \{\mathrm{root}(\ell) \mid \ell \to r \in \mathcal{R}\}$, where $f^\sharp$ has the same arity as f. If $\ell \to r \in \mathcal{R}$ and t is a subterm of r with a defined root symbol that is not a proper subterm of ℓ then the rule $\ell^\sharp \to t^\sharp$ is a *dependency pair* of $\mathcal{R}$. Here $\ell^\sharp$ and $t^\sharp$ are the result of replacing the root symbols in ℓ and t by the corresponding dependency pair symbols. The set of dependency pairs of $\mathcal{R}$ is denoted by $\mathsf{DP}(\mathcal{R})$. A *DP problem* is a pair of TRSs $(\mathcal{P}, \mathcal{R})$ such that the root symbols of the rules in $\mathcal{P}$ do neither occur in $\mathcal{R}$ nor in proper subterms of the left- and right-hand sides of rules in $\mathcal{P}$. The problem is said to be *finite* if there is no infinite sequence

$$s_1 \to_{\mathcal{P}}^{\epsilon} t_1 \to_{\mathcal{R}}^{*} s_2 \to_{\mathcal{P}}^{\epsilon} t_2 \to_{\mathcal{R}}^{*} \cdots$$

such that all terms $t_1, t_2, \ldots$ are terminating with respect to $\mathcal{R}$. Such an infinite sequence is said to be *minimal*. The main result underlying the dependency pair approach states that termination of a TRS $\mathcal{R}$ is equivalent to finiteness of the DP problem $(\mathsf{DP}(\mathcal{R}), \mathcal{R})$.

In order to prove a DP problem finite, a number of *DP processors* have been developed. DP processors are functions that take a DP problem as input and return a set of DP problems as output. In order to be employed to prove termination they need to be *sound*, that is, if all DP problems in a set returned by a DP processor are finite then the initial DP problem is finite. In addition, to ensure that a DP processor can be used to prove non-termination it must be *complete* which means that if one of the DP problems returned by the DP processor is not finite then the original DP problem is not finite.

A DP problem $(\mathcal{P}, \mathcal{R})$ is called *innermost finite* if there is no infinite sequence

$$s_1 \to_{\mathcal{P}}^{\epsilon} t_1 \overset{\mathsf{i}}{\to}_{\mathcal{R}}^{!} s_2 \to_{\mathcal{P}}^{\epsilon} t_2 \overset{\mathsf{i}}{\to}_{\mathcal{R}}^{!} \cdots$$

such that the term s_1 is in normal form with respect to $\mathcal{R}$. Soundness and completeness of DP problems for innermost termination are based on this altered notion of finiteness.

2.4 Currying

Definition 1 An *applicative* signature is a signature that consists of constants and a single binary function symbol called application, denoted by the infix and left-associative symbol $\star$. In examples we often use juxtaposition instead of $\star$. Based on an applicative signature we define *applicative terms (substitutions, contexts, TRSs)*. An applicative TRS is abbreviated by ATRS.

Every ordinary TRS can be transformed into an ATRS by currying.

Definition 2 Let $\mathcal{F}$ be a signature. The currying system $\mathcal{C}(\mathcal{F})$ consists of the rewrite rules $f_{i+1}(x_1, \ldots, x_i, y) \to f_i(x_1, \ldots, x_i) \star y$ for every n-ary function

symbol $f \in \mathcal{F}$ and every $0 \leqslant i < n$. Here $f_n = f$ and, for every $0 \leqslant i < n$, f_i is a fresh function symbol of arity i.

The currying system $\mathcal{C}(\mathcal{F})$ is confluent and terminating. Hence every term t has a unique normal form $t{\downarrow_{\mathcal{C}(\mathcal{F})}}$. For instance, $\mathsf{f}(\mathsf{a},\mathsf{b})$ is transformed into $\mathsf{f}_0\ \mathsf{a}_0\ \mathsf{b}_0$.

Definition 3 Let $\mathcal{R}$ be a TRS over the signature $\mathcal{F}$. The *curried* system $\mathcal{R}{\downarrow_{\mathcal{C}(\mathcal{F})}}$ is the ATRS consisting of the rules $\ell{\downarrow_{\mathcal{C}(\mathcal{F})}} \to r{\downarrow_{\mathcal{C}(\mathcal{F})}}$ for every $\ell \to r \in \mathcal{R}$. The signature of $\mathcal{R}{\downarrow_{\mathcal{C}(\mathcal{F})}}$ contains the application symbol $\star$ and a constant f_0 for every function symbol $f \in \mathcal{F}$.

In the following we write $\mathcal{R}{\downarrow_{\mathcal{C}}}$ for $\mathcal{R}{\downarrow_{\mathcal{C}(\mathcal{F})}}$ whenever $\mathcal{F}$ can be inferred from the context or is irrelevant. Moreover, we write f for f_0.

Example 1 The TRS $\mathcal{R}$

$$\begin{array}{cc} 0 + y \to y & 0 \times y \to 0 \\ \mathsf{s}(x) + y \to \mathsf{s}(x + y) & \mathsf{s}(x) \times y \to (x \times y) + y \end{array}$$

is transformed into the ATRS $\mathcal{R}{\downarrow_{\mathcal{C}}}$

$$\begin{array}{cc} +\ 0\ y \to y & \times\ 0\ y \to 0 \\ +\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (+\ x\ y) & \times\ (\mathsf{s}\ x)\ y \to +\ (\times\ x\ y)\ y \end{array}$$

Every rewrite sequence in $\mathcal{R}$ can be transformed into a sequence in $\mathcal{R}{\downarrow_{\mathcal{C}}}$, but the reverse does not hold. For instance, with respect to the above example, the rewrite step $+\ (\mathsf{s}\ (+\ 0))\ 0 \to \mathsf{s}\ (+\ (+\ 0)\ 0)$ in $\mathcal{R}{\downarrow_{\mathcal{C}}}$ does not correspond to a rewrite step in $\mathcal{R}$. Nevertheless, termination of $\mathcal{R}$ implies termination of $\mathcal{R}{\downarrow_{\mathcal{C}}}$.

Theorem 2 (Kennaway *et al.* [23]) *A TRS $\mathcal{R}$ is terminating if and only if $\mathcal{R}{\downarrow_{\mathcal{C}}}$ is terminating.* □

A simple self-labeling proof can be found in [30]. As an immediate consequence we get the following transformation method for proving termination of ATRSs.

Corollary 1 *An ATRS $\mathcal{R}$ is terminating if and only if there exists a terminating TRS $\mathcal{S}$ such that $\mathcal{S}{\downarrow_{\mathcal{C}}} = \mathcal{R}$ (modulo renaming).* □

In [13] this method is called *transformation $\mathcal{A}$*. As can be seen from the following example, the method does not handle partially applied terms and, more seriously, head variables. Hence the method is of limited applicability as it cannot cope with the higher-order aspects modeled by ATRSs.

Example 2 Consider the ATRS $\mathcal{R}$ (from [2])

$$\begin{array}{lll} 1\colon & \mathsf{id}\ x \to x & 4\colon \quad \mathsf{map}\ f\ \mathsf{nil} \to \mathsf{nil} \\ 2\colon & \mathsf{add}\ 0 \to \mathsf{id} & 5\colon \quad \mathsf{map}\ f\ (:\ x\ y) \to :\ (f\ x)\ (\mathsf{map}\ f\ y) \\ 3\colon & \mathsf{add}\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (\mathsf{add}\ x\ y) & \end{array}$$

Rules 1 and 4 are readily translated into functional form:

$$\mathsf{id}_1(x) \to x \qquad\qquad \mathsf{map}_2(f, \mathsf{nil}) \to \mathsf{nil}$$

However, we cannot find functional forms for rules 2 and 3 because the 'arity' of add is 1 in rule 2 and 2 in rule 3. Because of the presence of the head variable f in the subterm $f\ x$, there is no functional term t such that $t{\downarrow_{\mathcal{C}}} = : (f\ x)\ (\mathsf{map}\ f\ y)$. Hence also rule 5 cannot be transformed.

3 Full Termination

In this section we present an uncurrying transformation that can deal with ATRSs like in Example 2. This transformation preserves and reflects termination.

Throughout this section we assume that $\mathcal{R}$ is an ATRS over an applicative signature $\mathcal{F}$. In the sequel we restrict to TRSs where $\mathrm{aa}(f)$ (see next definition) is defined for every $f \in \mathcal{F}$. Note that $\mathrm{aa}(f)$ is defined if $\mathcal{R}$ is finite but may be undefined for infinite $\mathcal{R}$.

Definition 4 The *applicative arity* $\mathrm{aa}(f)$ of a constant $f \in \mathcal{F}$ is defined as the maximum n such that $f \star t_1 \star \cdots \star t_n$ is a subterm in the left- or right-hand side of a rule in $\mathcal{R}$. This notion is extended to terms as follows:

$$\mathrm{aa}(t) = \begin{cases} \mathrm{aa}(f) & \text{if } t \text{ is a constant } f \\ \mathrm{aa}(t_1) - 1 & \text{if } t = t_1 \star t_2 \end{cases}$$

Note that $\mathrm{aa}(t)$ is undefined if the head symbol of t is a variable and $\mathrm{aa}(f) = 0$ for a constant $f \in \mathcal{F}$ that does not appear in a rule in $\mathcal{R}$.

Definition 5 For an ATRS $\mathcal{R}$ the uncurrying system $\mathcal{U}(\mathcal{R})$ consists of the following rewrite rules $f_i(x_1, \ldots, x_i) \star y \to f_{i+1}(x_1, \ldots, x_i, y)$ for every constant $f \in \mathcal{F}$ and every $0 \leqslant i < \mathrm{aa}(f)$. Here $f_0 = f$ and, for every $i > 0$, f_i is a fresh function symbol of arity i. We say that $\mathcal{R}$ is *left head variable free* if no subterm of a left-hand side in $\mathcal{R}$ is of the form $t_1 \star t_2$ where t_1 is a variable. We write *ℓ-ATRS* to denote a left head variable free ATRS.

The uncurrying system $\mathcal{U}(\mathcal{R})$, or simply $\mathcal{U}$, is confluent and terminating. Hence every term t has a unique normal form $t{\downarrow_{\mathcal{U}}}$.

Definition 6 The *uncurried* system $\mathcal{R}{\downarrow_{\mathcal{U}}}$ is the TRS consisting of the rules $\ell{\downarrow_{\mathcal{U}}} \to r{\downarrow_{\mathcal{U}}}$ for every $\ell \to r \in \mathcal{R}$.

Example 3 The ATRS $\mathcal{R}$ of Example 2 is transformed into $\mathcal{R}{\downarrow_{\mathcal{U}}}$:

$$\begin{aligned} \mathsf{id}_1(x) &\to x & \mathsf{map}_2(f, \mathsf{nil}) &\to \mathsf{nil} \\ \mathsf{add}_1(0) &\to \mathsf{id} & \mathsf{map}_2(f, :_2(x, y)) &\to :_2(f \star x, \mathsf{map}_2(f, y)) \\ \mathsf{add}_2(\mathsf{s}_1(x), y) &\to \mathsf{s}_1(\mathsf{add}_2(x, y)) \end{aligned}$$

The TRS $\mathcal{R}{\downarrow_{\mathcal{U}}}$ is an obvious candidate of a TRS whose termination implies termination of the original ATRS. However, as can be seen from the following example, the rules of $\mathcal{R}{\downarrow_{\mathcal{U}}}$ are not enough to simulate an arbitrary rewrite sequence in $\mathcal{R}$.

Example 4 The ATRS $\mathcal{R}$

$$\mathsf{f}\ x \to x\ x$$

is non-terminating since $\mathsf{f}\ \mathsf{f} \to_{\mathcal{R}} \mathsf{f}\ \mathsf{f} \to_{\mathcal{R}} \cdots$ while the transformed TRS $\mathcal{R}{\downarrow_{\mathcal{U}}}$

$$\mathsf{f}_1(x) \to x \star x$$

is terminating.

The natural solution is to add $\mathcal{U}(\mathcal{R})$. In the following we write $\mathcal{U}^+(\mathcal{R})$ for $\mathcal{R}{\downarrow_{\mathcal{U}(\mathcal{R})}} \cup \mathcal{U}(\mathcal{R})$. As the following example shows, we do not yet have a sound transformation.

Example 5 The non-terminating ATRS $\mathcal{R}$

$$\mathsf{id}\ x \to x \qquad\qquad \mathsf{f}\ x \to \mathsf{id}\ \mathsf{f}\ x$$

is transformed into the terminating TRS $\mathcal{R}{\downarrow_{\mathcal{U}}}$

$$\mathsf{id}_1(x) \to x \qquad\qquad \mathsf{f}_1(x) \to \mathsf{id}_2(\mathsf{f}, x)$$

Note that $\mathrm{aa}(\mathsf{id}) = 2$ and $\mathrm{aa}(\mathsf{f}) = 1$. The TRS $\mathcal{U}^+(\mathcal{R})$ consists of the following rules

$$\begin{array}{lll} \mathsf{id}_1(x) \to x & \mathsf{id} \star x \to \mathsf{id}_1(x) & \mathsf{f} \star x \to \mathsf{f}_1(x) \\ \mathsf{f}_1(x) \to \mathsf{id}_2(\mathsf{f}, x) & \mathsf{id}_1(x) \star y \to \mathsf{id}_2(x, y) & \end{array}$$

and is easily shown to be terminating.

The ATRS $\mathcal{R}$ admits the cycle $\mathsf{f}\ x \to \mathsf{id}\ \mathsf{f}\ x \to \mathsf{f}\ x$. In $\mathcal{U}^+(\mathcal{R})$ we have $\mathsf{f}_1(x) \to \mathsf{id}_2(\mathsf{f}, x)$ but the term $\mathsf{id}_2(\mathsf{f}, x)$ does not rewrite to $\mathsf{f}_1(x)$. It would if the rule $\mathsf{id}\ x\ y \to x\ y$ were present in $\mathcal{R}$. This inspires the following definition.

Definition 7 Let $\mathcal{R}$ be an ATRS. The *η-saturated* ATRS $\mathcal{R}_\eta$ is the smallest extension of $\mathcal{R}$ such that $\ell \star x \to r \star x \in \mathcal{R}_\eta$ whenever $\ell \to r \in \mathcal{R}_\eta$ and $\mathrm{aa}(\ell) > 0$. Here x is a variable that does not appear in $\ell \to r$.

The rules added during η-saturation do not affect the termination behaviour of $\mathcal{R}$, according to the following lemma. Moreover, $\mathcal{R}_\eta$ is an ℓ-ATRS if and only if $\mathcal{R}$ is an ℓ-ATRS.

Lemma 1 *If $\mathcal{R}$ is an ℓ-ATRS then $\to_{\mathcal{R}}$ and $\to_{\mathcal{R}_\eta}$ coincide.*

Proof The inclusion $\to_{\mathcal{R}} \subseteq \to_{\mathcal{R}_\eta}$ trivially follows from the inclusion $\mathcal{R} \subseteq \mathcal{R}_\eta$. For the reverse inclusion we show that for every rewrite step $s = C[\ell\sigma] \to_{\mathcal{R}_\eta} C[r\sigma] = t$ there exist a rule $\ell' \to r' \in \mathcal{R}$ and a context C' such that $s = C'[\ell'\sigma]$ and $t = C'[r'\sigma]$. We use induction on the derivation of $\ell \to r \in \mathcal{R}_\eta$. In the base case $\ell \to r \in \mathcal{R}$ and we simply take $\ell' \to r' = \ell \to r$ and $C' = C$. In the induction step we have $\ell \to r = \ell' \star x \to r' \star x$ for some $\ell' \to r' \in \mathcal{R}_\eta$ with $\mathrm{aa}(\ell') > 0$. Define $C' = C[\Box \star x\sigma]$. Clearly $s = C'[\ell'\sigma] \to_{\mathcal{R}_\eta} C'[r'\sigma] = t$. The induction hypothesis yields a rule $\ell'' \to r'' \in \mathcal{R}$ and a context C'' such that $s = C''[\ell''\sigma]$ and $t = C''[r''\sigma]$. □

In the following we write $\mathcal{U}^+_\eta(\mathcal{R})$ for $\mathcal{R}_\eta{\downarrow_{\mathcal{U}(\mathcal{R})}} \cup \mathcal{U}(\mathcal{R})$. We can now state the first major result of this article.

Theorem 3 *An ℓ-ATRS $\mathcal{R}$ is terminating if $\mathcal{U}^+_\eta(\mathcal{R})$ is terminating.*

Before we prepare for the proof of the theorem above we show another subtlety of the uncurrying transformation. While η-saturation may change applicative arities, the applicative arities used in the definition of $\mathcal{U}^+_\eta(\mathcal{R})$ always refer to those of $\mathcal{R}$ and not $\mathcal{R}_\eta$.

Example 6 The non-terminating ATRS $\mathcal{R}$

$$\mathsf{f} \to \mathsf{g}\ \mathsf{a} \qquad\qquad \mathsf{g} \to \mathsf{f}$$

is transformed into $\mathcal{U}^+_\eta(\mathcal{R})$

$$\mathsf{f} \to \mathsf{g}_1(\mathsf{a}) \qquad \mathsf{g} \to \mathsf{f} \qquad \mathsf{g}_1(x) \to \mathsf{f} \star x \qquad \mathsf{g} \star x \to \mathsf{g}_1(x)$$

because $\mathrm{aa}(\mathsf{f}) = 0$. The resulting TRS is non-terminating. If the applicative arities of $\mathcal{R}_\eta$ were employed, uncurrying would produce the terminating TRS

$$\mathsf{f} \to \mathsf{g}_1(\mathsf{a}) \quad \mathsf{g} \to \mathsf{f} \quad \mathsf{g}_1(x) \to \mathsf{f}_1(x) \quad \mathsf{g} \star x \to \mathsf{g}_1(x) \quad \mathsf{f} \star x \to \mathsf{f}_1(x)$$

since $\mathrm{aa}(\mathsf{f}) = 1$ for $\mathcal{R}_\eta$.

Before presenting the proof of Theorem 3, we revisit the running example.

Example 7 Consider again the ATRS $\mathcal{R}$ of Example 2. Proving termination of the transformed TRS $\mathcal{U}^+_\eta(\mathcal{R})$

$$\begin{array}{rcc}
\mathsf{id}_1(x) \to x & : \star x \to :_1(x) & \mathsf{id} \star x \to \mathsf{id}_1(x) \\
\mathsf{add}_1(0) \to \mathsf{id} & :_1(x) \star y \to :_2(x, y) & \mathsf{add} \star x \to \mathsf{add}_1(x) \\
\mathsf{add}_2(0, y) \to \mathsf{id}_1(y) & & \mathsf{add}_1(x) \star y \to \mathsf{add}_2(x, y) \\
\mathsf{add}_2(\mathsf{s}_1(x), y) \to \mathsf{s}_1(\mathsf{add}_2(x, y)) & & \mathsf{s} \star x \to \mathsf{s}_1(x) \\
\mathsf{map}_2(f, \mathsf{nil}) \to \mathsf{nil} & & \mathsf{map} \star x \to \mathsf{map}_1(x) \\
\mathsf{map}_2(f, :_2(x, y)) \to :_2(f \star x, \mathsf{map}_2(f, y)) & & \mathsf{map}_1(x) \star y \to \mathsf{map}_2(x, y)
\end{array}$$

is possible using LPO with a quasi-precedence.

The next lemma states an easy result that is freely used in the sequel.

Lemma 2 *Let s be an applicative term. If $s = x \star s_1 \star \cdots \star s_n$ then $s{\downarrow_{\mathcal{U}}} = x \star s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}}$ and if $s = f \star s_1 \star \cdots \star s_n$ then $s{\downarrow_{\mathcal{U}}} = f_i(s_1{\downarrow_{\mathcal{U}}}, \ldots, s_i{\downarrow_{\mathcal{U}}}) \star s_{i+1}{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}}$ for $i = \min\{\mathrm{aa}(f), n\}$.*

Proof We show the second claim by induction on n (the first one is similar). In the base case $n = 0$ and $f{\downarrow_{\mathcal{U}}} = f$ concludes this case. In the inductive step

$$s \to_{\mathcal{U}}^* s' = f_j(s_1{\downarrow_{\mathcal{U}}}, \ldots, s_j{\downarrow_{\mathcal{U}}}) \star s_{j+1}{\downarrow_{\mathcal{U}}} \star \cdots \star s_{n+1}{\downarrow_{\mathcal{U}}}$$

for $j = \min\{\mathrm{aa}(f), n\}$ follows from the induction hypothesis and the definition of $(\cdot){\downarrow_{\mathcal{U}}}$. If $i < n+1$ then $j = i$ and $s' = s{\downarrow_{\mathcal{U}}}$. In the case where $i = n+1$ then $j = n$ and the result follows from

$$f_n(s_1{\downarrow_{\mathcal{U}}}, \ldots, s_n{\downarrow_{\mathcal{U}}}) \star s_{n+1}{\downarrow_{\mathcal{U}}} \to_{\mathcal{U}} f_{n+1}(s_1{\downarrow_{\mathcal{U}}}, \ldots, s_n{\downarrow_{\mathcal{U}}}, s_{n+1}{\downarrow_{\mathcal{U}}}) = s{\downarrow_{\mathcal{U}}}$$

□

The following two lemmata state factorisation properties which are used in the proof of Theorem 3.

Lemma 3 *Let $s_1, \ldots, s_n$ be applicative terms. Then $s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} \to_{\mathcal{U}}^* (s_1 \star \cdots \star s_n){\downarrow_{\mathcal{U}}}$. If $\mathrm{aa}(s_1) \leqslant 0$ or if $\mathrm{aa}(s_1)$ is undefined then $s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} = (s_1 \star \cdots \star s_n){\downarrow_{\mathcal{U}}}$.*

Proof For the first claim observe that $s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} \,{}_{\mathcal{U}}^{*}\!\leftarrow s_1 \star \cdots \star s_n \to_{\mathcal{U}}^! (s_1 \star \cdots \star s_n){\downarrow_{\mathcal{U}}}$. The claim then follows from the confluence of $\mathcal{U}$.

For the second claim observe that if $\mathrm{aa}(s_1) \leqslant 0$ or if $\mathrm{aa}(s_1)$ is undefined then $s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} \in NF(\mathcal{U})$ and the result follows from the confluence of $\mathcal{U}$ as in the first case. □

For an applicative substitution σ, we write $\sigma{\downarrow_{\mathcal{U}}}$ for the substitution $\{x \mapsto \sigma(x){\downarrow_{\mathcal{U}}} \mid x \in \mathcal{V}\}$.

Lemma 4 *Let σ be an applicative substitution. For every applicative term t, $t{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \to_{\mathcal{U}}^* (t\sigma){\downarrow_{\mathcal{U}}}$. If t is head variable free then $t{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} = (t\sigma){\downarrow_{\mathcal{U}}}$.*

Proof We prove the former claim by induction on the term t. The proof for the head variable free case is similar. It suffices to consider the step case.

– Consider the step case with $t = x \star t_1 \star \cdots \star t_n$. Then

$$\begin{aligned} t{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} &= (x \star t_1{\downarrow_{\mathcal{U}}} \star \cdots \star t_n{\downarrow_{\mathcal{U}}})\sigma{\downarrow_{\mathcal{U}}} \\ &= \sigma(x){\downarrow_{\mathcal{U}}} \star t_1{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \star \cdots \star t_n{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \\ &\to_{\mathcal{U}}^* \sigma(x){\downarrow_{\mathcal{U}}} \star (t_1\sigma){\downarrow_{\mathcal{U}}} \star \cdots \star (t_n\sigma){\downarrow_{\mathcal{U}}} \\ &\to_{\mathcal{U}}^* (\sigma(x) \star t_1\sigma \star \cdots \star t_n\sigma){\downarrow_{\mathcal{U}}} = (t\sigma){\downarrow_{\mathcal{U}}} \end{aligned}$$

where Lemma 2 is applied in the first equality, the induction hypothesis in the first $\to_{\mathcal{U}}^*$ step, and Lemma 3 in the second $\to_{\mathcal{U}}^*$ step.

– Consider the step case with $t = f \star t_1 \star \cdots \star t_n$. Let $i = \min\{\mathrm{aa}(f), n\}$. Then

$$\begin{aligned} t{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} &= (f_i(t_1{\downarrow_{\mathcal{U}}}, \ldots, t_i{\downarrow_{\mathcal{U}}}) \star t_{i+1}{\downarrow_{\mathcal{U}}} \star \cdots \star t_n{\downarrow_{\mathcal{U}}})\sigma{\downarrow_{\mathcal{U}}} \\ &= f_i(t_1{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}}, \ldots, t_i{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}}) \star t_{i+1}{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \star \cdots \star t_n{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \\ &\to^*_{\mathcal{U}} f_i((t_1\sigma){\downarrow_{\mathcal{U}}}, \ldots, (t_i\sigma){\downarrow_{\mathcal{U}}}) \star (t_{i+1}\sigma){\downarrow_{\mathcal{U}}} \star \cdots \star (t_n\sigma){\downarrow_{\mathcal{U}}} \\ &= f_i(t_1\sigma, \ldots, t_i\sigma){\downarrow_{\mathcal{U}}} \star (t_{i+1}\sigma){\downarrow_{\mathcal{U}}} \star \cdots \star (t_n\sigma){\downarrow_{\mathcal{U}}} \\ &= (f \star t_1\sigma \star \cdots \star t_i\sigma){\downarrow_{\mathcal{U}}} \star (t_{i+1}\sigma){\downarrow_{\mathcal{U}}} \star \cdots \star (t_n\sigma){\downarrow_{\mathcal{U}}} \;=\; (t\sigma){\downarrow_{\mathcal{U}}} \end{aligned}$$

where Lemma 2 is applied in the first equality, the induction hypothesis in the first $\to^*_{\mathcal{U}}$ step and Lemma 3 in the last equality. □

Now we are ready to present the proof of Theorem 3.

Proof (of Theorem 3) We show that $s{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} t{\downarrow_{\mathcal{U}}}$ whenever $s \to_{\mathcal{R}_\eta} t$ for applicative terms s and t. This entails that any infinite $\mathcal{R}_\eta$ derivation is transformed into an infinite $\mathcal{U}^+_\eta(\mathcal{R})$ derivation. The theorem follows from this observation and Lemma 1. Let $s = C[\ell\sigma]$ and $t = C[r\sigma]$ with $\ell \to r \in \mathcal{R}_\eta$. We use induction on the size of the context C.

– If $C = \Box$ then $s{\downarrow_{\mathcal{U}}} = (\ell\sigma){\downarrow_{\mathcal{U}}} = \ell{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}}$ and $r{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \to^*_{\mathcal{U}} (r\sigma){\downarrow_{\mathcal{U}}} = t{\downarrow_{\mathcal{U}}}$ by Lemma 4. Hence $s{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} t{\downarrow_{\mathcal{U}}}$.
– Suppose $C = \Box \star s_1 \star \cdots \star s_n$ and $n > 0$. Since $\mathcal{R}_\eta$ is left head variable free, $\mathrm{aa}(\ell)$ is defined. If $\mathrm{aa}(\ell) = 0$ then

$$\begin{aligned} s{\downarrow_{\mathcal{U}}} &= (\ell\sigma \star s_1 \star \cdots \star s_n){\downarrow_{\mathcal{U}}} = (\ell\sigma){\downarrow_{\mathcal{U}}} \star s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} \\ &= \ell{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \star s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} \end{aligned}$$

and

$$\begin{aligned} r{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \star s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} &\to^*_{\mathcal{U}} (r\sigma){\downarrow_{\mathcal{U}}} \star s_1{\downarrow_{\mathcal{U}}} \star \cdots \star s_n{\downarrow_{\mathcal{U}}} \\ &\to^*_{\mathcal{U}} (r\sigma \star s_1 \star \cdots \star s_n){\downarrow_{\mathcal{U}}} = t{\downarrow_{\mathcal{U}}} \end{aligned}$$

by applications of Lemmata 4 and 3. Hence $s{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} t{\downarrow_{\mathcal{U}}}$. If $\mathrm{aa}(\ell) > 0$ then $\ell \star x \to r \star x \in \mathcal{R}_\eta$ for some fresh variable x. We have $s = C'[(\ell \star x)\tau]$ and $t = C'[(r \star x)\tau]$ for the context $C' = \Box \star s_2 \star \cdots \star s_n$ and the substitution $\tau = \sigma \cup \{x \mapsto s_1\}$. Since C' is smaller than C, we can apply the induction hypothesis which yields the desired result.
– In the remaining case $C = s_1 \star C'$. The induction hypothesis yields

$$C'[\ell\sigma]{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} C'[r\sigma]{\downarrow_{\mathcal{U}}}$$

If $\mathrm{aa}(s_1) \leqslant 0$ or if $\mathrm{aa}(s_1)$ is undefined then $s{\downarrow_{\mathcal{U}}} = s_1{\downarrow_{\mathcal{U}}} \star C'[\ell\sigma]{\downarrow_{\mathcal{U}}}$ and $t{\downarrow_{\mathcal{U}}} = s_1{\downarrow_{\mathcal{U}}} \star C'[r\sigma]{\downarrow_{\mathcal{U}}}$ by Lemma 3. If $\mathrm{aa}(s_1) > 0$ then $s_1{\downarrow_{\mathcal{U}}} = f_i(u_1, \ldots, u_i)$ for the head symbol f of s_1 and some terms $u_1, \ldots, u_i$. So

$$s{\downarrow_{\mathcal{U}}} = f_{i+1}(u_1, \ldots, u_i, C'[\ell\sigma]{\downarrow_{\mathcal{U}}})$$

and

$$t{\downarrow_{\mathcal{U}}} = f_{i+1}(u_1, \ldots, u_i, C'[r\sigma]{\downarrow_{\mathcal{U}}})$$

Hence in both cases we obtain $s{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} t{\downarrow_{\mathcal{U}}}$. □

The next example shows that the left head variable freeness condition cannot be weakened to the well-definedness of $\mathrm{aa}(\ell)$ for every left-hand side ℓ.

Example 8 Consider the non-terminating ATRS $\mathcal{R}$

$$\mathsf{f}\ (x\ \mathsf{a}) \to \mathsf{f}\ (\mathsf{g}\ \mathsf{b}) \qquad\qquad \mathsf{g}\ \mathsf{b} \to \mathsf{h}\ \mathsf{a}$$

The transformed TRS $\mathcal{U}^+_\eta(\mathcal{R})$ consists of the rules

$$\begin{array}{lll} \mathsf{f}_1(x \star \mathsf{a}) \to \mathsf{f}_1(\mathsf{g}_1(\mathsf{b})) & \mathsf{f} \star x \to \mathsf{f}_1(x) & \mathsf{h} \star x \to \mathsf{h}_1(x) \\ \mathsf{g}_1(\mathsf{b}) \to \mathsf{h}_1(\mathsf{a}) & \mathsf{g} \star x \to \mathsf{g}_1(x) & \end{array}$$

and is terminating because its rules are oriented from left to right by the lexicographic path order with precedence $\star > \mathsf{g}_1 > \mathsf{f}_1 > \mathsf{h}_1 > \mathsf{a} > \mathsf{b}$. Note that $\mathrm{aa}(\mathsf{f}\ (x\ \mathsf{a})) = 0$.

The uncurrying transformation is not always useful.

Example 9 Consider the one-rule TRS $\mathcal{R}$

$$\mathsf{C}\ x\ y\ z\ u \to x\ z\ (x\ y\ z\ u)$$

from [9]. The termination of $\mathcal{R}$ is proved by the lexicographic path order with empty precedence. The transformed TRS $\mathcal{U}^+_\eta(\mathcal{R})$ consists of

$$\begin{array}{lll} \mathsf{C}_4(x,y,z,u) \to x \star z \star (x \star y \star z \star u) & & \\ \mathsf{C} \star x \to \mathsf{C}_1(x) & & \mathsf{C}_2(x,y) \star z \to \mathsf{C}_3(x,y,z) \\ \mathsf{C}_1(x) \star y \to \mathsf{C}_2(x,y) & & \mathsf{C}_3(x,y,z) \star u \to \mathsf{C}_4(x,y,z,u) \end{array}$$

None of the tools that participated in the termination competitions between 2005 and 2010 is able to prove the termination of this TRS.

We show that the converse of Theorem 3 also holds. Hence the uncurrying transformation does not only reflect but also preserve termination. (This does not contradict the preceding example.) To show this result (Theorem 4) we transform any infinite sequence in $\mathcal{U}^+_\eta(\mathcal{R})$ into an infinite sequence of the original ℓ-ATRS $\mathcal{R}$. Since $\mathcal{U}$ is terminating, any infinite sequence in $\mathcal{U}^+_\eta(\mathcal{R})$ must have the shape

$$\to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}} \cdot \to^*_{\mathcal{U}} \cdot \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}} \cdot \to^*_{\mathcal{U}} \cdots \tag{1}$$

with infinitely many $\mathcal{R}_\eta\downarrow_{\mathcal{U}}$ steps. Below we write $\mathcal{C}'$ for the TRS that is obtained from $\mathcal{U}$ by reversing all rules: $\{r \to \ell \mid \ell \to r \in \mathcal{U}(\mathcal{R})\}$ The TRS $\mathcal{C}'$ allows to mimic a rewrite step $\to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}}$ in the original ℓ-ATRS $\mathcal{R}$ (Lemma 6) and equates any two terms that are in the relation $\to^*_{\mathcal{U}}$ (Lemma 7). Then the sequence (1) can be transformed into an infinite sequence in $\mathcal{R}$.

Remark 1 Note that $\downarrow_{\mathcal{C}}$ is not the inverse of $\downarrow_{\mathcal{U}}$ since for the TRS $\mathcal{R}$ from Example 5 we have $\mathcal{R}\downarrow_{\mathcal{U}}\downarrow_{\mathcal{C}} = \{\mathsf{id}_1\ x \to x, \mathsf{f}_1\ x \to \mathsf{id}_2\ \mathsf{f}\ x\}$ which is different from $\mathcal{R}$. But obviously $\mathcal{R}\downarrow_{\mathcal{U}}\downarrow_{\mathcal{C}'} = \mathcal{R}$ for any ATRS $\mathcal{R}$.

Lemma 5 *Let $\mathcal{R}$ be an ℓ-ATRS. If t, C, and σ are a term, context, and substitution over the signature of $\mathcal{U}^+_\eta(\mathcal{R})$, then $(C[t\sigma])\downarrow_{\mathcal{C}'} = C\downarrow_{\mathcal{C}'}[t\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}]$.*

Proof We show $(t\sigma)\downarrow_{\mathcal{C}'} = t\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}$ by induction on t. If $t = x$ then the result follows since $(x\sigma)\downarrow_{\mathcal{C}'} = \sigma(x)\downarrow_{\mathcal{C}'} = x\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}$. If $t = f_i(t_1, \ldots, t_i)$ then

$$\begin{aligned}(t\sigma)\downarrow_{\mathcal{C}'} &= f_i(t_1\sigma, \ldots, t_i\sigma)\downarrow_{\mathcal{C}'} = f \star (t_1\sigma)\downarrow_{\mathcal{C}'} \star \cdots \star (t_i\sigma)\downarrow_{\mathcal{C}'}\\ &= f \star t_1\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'} \star \cdots \star t_i\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'} = (f \star t_1\downarrow_{\mathcal{C}'} \star \cdots \star t_i\downarrow_{\mathcal{C}'})\sigma\downarrow_{\mathcal{C}'}\\ &= f_i(t_1, \ldots, t_i)\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}\end{aligned}$$

by the induction hypothesis. If $t = t_1 \star t_2$ then

$$\begin{aligned}(t\sigma)\downarrow_{\mathcal{C}'} &= (t_1\sigma \star t_2\sigma)\downarrow_{\mathcal{C}'} = (t_1\sigma)\downarrow_{\mathcal{C}'} \star (t_2\sigma)\downarrow_{\mathcal{C}'} = t_1\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'} \star t_2\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}\\ &= (t_1\downarrow_{\mathcal{C}'} \star t_2\downarrow_{\mathcal{C}'})\sigma\downarrow_{\mathcal{C}'} = t\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}\end{aligned}$$

by the induction hypothesis.

The proof that $(C[u])\downarrow_{\mathcal{C}'} = C\downarrow_{\mathcal{C}'}[u\downarrow_{\mathcal{C}'}]$ is by induction on C and similar. The lemma then follows from these two results. □

Lemma 6 *Let $\mathcal{R}$ be an ℓ-ATRS. If s and t are terms over the signature of $\mathcal{U}^+_\eta(\mathcal{R})$ then $s \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}} t$ implies $s\downarrow_{\mathcal{C}'} \to_{\mathcal{R}_\eta} t\downarrow_{\mathcal{C}'}$.*

Proof From $s \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}} t$ we get $s = C[\ell\sigma]$, $t = C[r\sigma]$ for some $\ell \to r \in \mathcal{R}_\eta\downarrow_{\mathcal{U}}$. By Lemma 5 we obtain $s\downarrow_{\mathcal{C}'} = C\downarrow_{\mathcal{C}'}[\ell\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}]$ and $t\downarrow_{\mathcal{C}'} = C\downarrow_{\mathcal{C}'}[r\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}]$. The result then follows in connection with the fact that $\mathcal{R}_\eta\downarrow_{\mathcal{U}}\downarrow_{\mathcal{C}'} = \mathcal{R}_\eta$. □

Lemma 7 *Let $\mathcal{R}$ be an ℓ-ATRS. If s and t are terms over the signature of $\mathcal{U}^+_\eta(\mathcal{R})$ then $s \to_{\mathcal{U}} t$ implies $s\downarrow_{\mathcal{C}'} = t\downarrow_{\mathcal{C}'}$.*

Proof From $s \to_{\mathcal{U}} t$ we get $s = C[\ell\sigma]$, $t = C[r\sigma]$ for some $\ell \to r \in \mathcal{U}$. By Lemma 5 we obtain $s\downarrow_{\mathcal{C}'} = C\downarrow_{\mathcal{C}'}[\ell\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}]$ and $t\downarrow_{\mathcal{C}'} = C\downarrow_{\mathcal{C}'}[r\downarrow_{\mathcal{C}'}\sigma\downarrow_{\mathcal{C}'}]$. The result then follows in connection with the observation that all rules in $\mathcal{U}\downarrow_{\mathcal{C}'}$ have equal left- and right-hand sides. □

Theorem 4 *If $\mathcal{R}$ is a terminating ℓ-ATRS then $\mathcal{U}^+_\eta(\mathcal{R})$ is terminating.*

Proof Assume that $\mathcal{U}^+_\eta(\mathcal{R})$ is non-terminating. Since $\mathcal{U}$ is terminating, any infinite rewrite sequence has the form

$$s_1 \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}} t_1 \to^*_{\mathcal{U}} s_2 \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}} t_2 \to^*_{\mathcal{U}} \cdots$$

Applications of Lemmata 6 and 7 transform this sequence into

$$s_1\downarrow_{\mathcal{C}'} \to_{\mathcal{R}_\eta} t_1\downarrow_{\mathcal{C}'} = s_2\downarrow_{\mathcal{C}'} \to_{\mathcal{R}_\eta} t_2\downarrow_{\mathcal{C}'} = \cdots$$

It follows that $\mathcal{R}_\eta$ is non-terminating. Since $\to_{\mathcal{R}} = \to_{\mathcal{R}_\eta}$ by Lemma 1, we conclude that $\mathcal{R}$ is non-terminating. □

Next we describe a trivial mirroring technique for TRSs. This technique can be used to eliminate some of the left head variables in an ATRS.

Definition 8 Let t be a term. The term t^M is defined as follows:

$$t^M = \begin{cases} t & \text{if } t \text{ is a variable} \\ f(t_n^M, \ldots, t_1^M) & \text{if } t = f(t_1, \ldots, t_n) \end{cases}$$

Moreover, if $\mathcal{R}$ is a TRS then $\mathcal{R}^M = \{\ell^M \to r^M \mid \ell \to r \in \mathcal{R}\}$.

We obviously have $s \to_{\mathcal{R}} t$ if and only if $s^M \to_{\mathcal{R}^M} t^M$. This gives the following result.

Theorem 5 *A TRS $\mathcal{R}$ is terminating if and only if $\mathcal{R}^M$ is terminating.* □

Example 10 Consider the one-rule ATRS $\mathcal{R}$

$$x\ (\mathsf{a}\ \mathsf{a}\ \mathsf{a}) \to \mathsf{a}\ (\mathsf{a}\ \mathsf{a})\ x$$

While $\mathcal{R}$ has a head variable in its left-hand side, the mirrored version $\mathcal{R}^M$

$$\mathsf{a}\ (\mathsf{a}\ \mathsf{a})\ x \to x\ (\mathsf{a}\ \mathsf{a}\ \mathsf{a})$$

is left head variable free. The transformed TRS $\mathcal{U}_\eta^+(\mathcal{R}^M)$

$$\mathsf{a}_2(\mathsf{a}_1(\mathsf{a}), x) \to x \star \mathsf{a}_2(\mathsf{a}, \mathsf{a}) \qquad \mathsf{a} \star x \to \mathsf{a}_1(x) \qquad \mathsf{a}_1(x) \star y \to \mathsf{a}_2(x, y)$$

is easily proved terminating with dependency pairs and a matrix interpretation of dimension one.

4 Innermost Termination

Before we prove that our transformation reflects innermost termination we show that it does not preserve innermost termination.

Example 11 Consider the ATRS $\mathcal{R}$

$$\mathsf{f}\ x \to \mathsf{f}\ x \qquad\qquad \mathsf{f} \to \mathsf{g}$$

In an innermost sequence the first rule is never applied and hence $\mathcal{R}$ is innermost terminating. The TRS $\mathcal{U}_\eta^+(\mathcal{R})$

$$\mathsf{f}_1(x) \to \mathsf{f}_1(x) \qquad \mathsf{f} \to \mathsf{g} \qquad \mathsf{f}_1(x) \to \mathsf{g} \star x \qquad \mathsf{f} \star x \to \mathsf{f}_1(x)$$

is not innermost terminating due to the rule $\mathsf{f}_1(x) \to \mathsf{f}_1(x)$.

The overlap between the rules of $\mathcal{R}$ in the above example is essential. This follows from a result of Gramlich [15] stating that innermost termination and termination coincide for locally confluent overlay systems. Hence for systems that satisfy the above conditions preservation of innermost termination can be recovered.

Theorem 6 *Let $\mathcal{R}$ be a locally confluent overlay ℓ-ATRS. If $\mathcal{R}$ is innermost terminating then $\mathcal{U}_\eta^+(\mathcal{R})$ is innermost terminating.*

Proof Suppose $\mathcal{R}$ is innermost terminating. From [15] we know that $\mathcal{R}$ is terminating. Since uncurrying preserves termination (Theorem 4) also $\mathcal{U}^+_\eta(\mathcal{R})$ is terminating. In particular, $\mathcal{U}^+_\eta(\mathcal{R})$ is innermost terminating. □

In the sequel we investigate if uncurrying reflects innermost termination. The next example shows that even in the innermost setting, η-saturation cannot be omitted. This is surprising since the η-rules are not innermost with respect to the original TRS but by uncurrying they become applicable at innermost redexes.

Example 12 The ATRS $\mathcal{R}$

$$\mathsf{h}\ x \to \mathsf{f}\ x \qquad\qquad \mathsf{f} \to \mathsf{h}$$

is not innermost terminating while $\mathcal{U}^+(\mathcal{R})$

$$\mathsf{h}_1(x) \to \mathsf{f}_1(x) \qquad \mathsf{f} \to \mathsf{h} \qquad \mathsf{h} \star x \to \mathsf{h}_1(x) \qquad \mathsf{f} \star x \to \mathsf{f}_1(x)$$

is (innermost) terminating. Note that $\mathcal{U}^+_\eta(\mathcal{R})$ is not innermost terminating because it also contains the rule $\mathsf{f}_1(x) \to \mathsf{h}_1(x)$.

The next example shows that $s \xrightarrow{\mathsf{i}}_{\mathcal{R}} t$ does not imply $s{\downarrow_{\mathcal{U}}} \xrightarrow{\mathsf{i}}^+_{\mathcal{U}^+_\eta(\mathcal{R})} t{\downarrow_{\mathcal{U}}}$. This does not contradict that uncurrying reflects innermost termination but shows that the proof of Theorem 3 cannot be adopted for the innermost case without further ado.

Example 13 Consider the ATRS $\mathcal{R}$

$$\mathsf{f} \to \mathsf{g} \qquad\qquad \mathsf{a} \to \mathsf{b} \qquad\qquad \mathsf{g}\ x \to \mathsf{h}$$

and the innermost step $s = \mathsf{f}\ \mathsf{a} \xrightarrow{\mathsf{i}}_{\mathcal{R}} \mathsf{g}\ \mathsf{a} = t$. We have $s{\downarrow_{\mathcal{U}}} = \mathsf{f} \star \mathsf{a}$ and $t{\downarrow_{\mathcal{U}}} = \mathsf{g}_1(\mathsf{a})$. In the TRS $\mathcal{U}^+_\eta(\mathcal{R})$

$$\mathsf{f} \to \mathsf{g} \qquad \mathsf{a} \to \mathsf{b} \qquad \mathsf{g}_1(x) \to \mathsf{h} \qquad \mathsf{g} \star x \to \mathsf{g}_1(x)$$

we have $s{\downarrow_{\mathcal{U}}} \xrightarrow{\mathsf{i}}_{\mathcal{U}^+_\eta(\mathcal{R})} \mathsf{g} \star \mathsf{a}$ but the step from $\mathsf{g} \star \mathsf{a}$ to $t{\downarrow_{\mathcal{U}}}$ is not innermost.

The problem in Example 13 is that uncurrying steps performed after the corresponding rewrite step need not be innermost. Moreover, not even an innermost variant of Lemma 4 holds as the next example demonstrates.

Example 14 Consider the ATRS $\mathcal{R}$ consisting of the rules

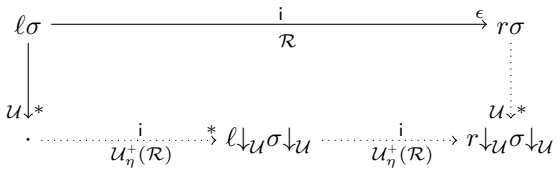
$$\mathsf{h} \to \mathsf{b} \qquad\qquad \mathsf{h}\ x \to \mathsf{c}$$

the term $t = x \star y$ and the substitution $\sigma = \{x \mapsto \mathsf{h}\}$. We have $t{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} = \mathsf{h} \star y$ and $(t\sigma){\downarrow_{\mathcal{U}}} = \mathsf{h}_1(y)$. The TRS $\mathcal{U}^+_\eta(\mathcal{R})$ consists of the rules

$$\mathsf{h} \to \mathsf{b} \qquad \mathsf{h}_1(x) \to \mathsf{c} \qquad \mathsf{h}_1(x) \to \mathsf{b} \star x \qquad \mathsf{h} \star x \to \mathsf{h}_1(x)$$

Like in the previous example, the step from $t{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}}$ to $(t\sigma){\downarrow_{\mathcal{U}}}$ is not innermost.

The above problems can be solved if we consider terms that are not completely uncurried. In our proof we follow a lazy approach and postpone uncurrying as long as possible. We show that an innermost root step in an ATRS $\mathcal{R}$ can be mimicked by an innermost sequence in $\mathcal{U}^+_\eta(\mathcal{R})$ according to the following diagram (Lemma 10):



To show this result we need that rewriting with $\to^*_{\mathcal{U}}$ preserves $\mathcal{R}$-normal forms (Lemma 8) and does not create innermost redexes (Lemma 9). By considering rightmost innermost rewriting (Lemma 11) we can then establish that uncurrying reflects innermost termination (Theorem 7).

Lemma 8 *Let $\mathcal{R}$ be an ℓ-ATRS. If s is a term over the signature of $\mathcal{R}$, $s \in NF(\mathcal{R})$, and $s \to^*_{\mathcal{U}} t$ then $t \in NF(\mathcal{R}_\eta{\downarrow_{\mathcal{U}}})$.*

Proof From Lemma 7 we obtain $s{\downarrow_{\mathcal{C}'}} = t{\downarrow_{\mathcal{C}'}}$. Note that $s{\downarrow_{\mathcal{C}'}} = s$ because s is a term over the signature of $\mathcal{R}$. If $t \notin NF(\mathcal{R}_\eta{\downarrow_{\mathcal{U}}})$ then $t \to_{\mathcal{R}_\eta{\downarrow_{\mathcal{U}}}} u$ for some term u. Lemma 6 yields $t{\downarrow_{\mathcal{C}'}} \to_{\mathcal{R}_\eta} u{\downarrow_{\mathcal{C}'}}$ and Lemma 1 yields $s \to_{\mathcal{R}} u{\downarrow_{\mathcal{C}'}}$. Hence $s \notin NF(\mathcal{R})$, contradicting the assumption. □

The following lemma states that uncurrying steps followed by taking a proper subterm can be reordered into first taking a proper subterm and then perform uncurrying steps.

Lemma 9 *Let $\mathcal{R}$ be an ℓ-ATRS. If s is a term over the signature of $\mathcal{R}$ then $s \to^*_{\mathcal{U}} \cdot \rhd u$ implies $s \rhd \cdot \to^*_{\mathcal{U}} u$.*

Proof Assume $s \to^*_{\mathcal{U}} t \rhd u$. We show that $s \rhd \cdot \to^*_{\mathcal{U}} u$ by induction on s. If s is a variable or a constant then there is nothing to show. So let $s = s_1 \star s_2$. We consider two cases.

- If the outermost $\star$ has not been uncurried then $t = t_1 \star t_2$ with $s_1 \to^*_{\mathcal{U}} t_1$ and $s_2 \to^*_{\mathcal{U}} t_2$. Without loss of generality assume that $t_1 \trianglerighteq u$. If $t_1 = u$ then $s \rhd s_1 \to^*_{\mathcal{U}} t_1$. If $t_1 \rhd u$ then the induction hypothesis yields $s_1 \rhd \cdot \to^*_{\mathcal{U}} u$ and hence also $s \rhd \cdot \to^*_{\mathcal{U}} u$.
- If the outermost $\star$ has been uncurried in the sequence from s to t then the head symbol of s_1 cannot be a variable and $\mathrm{aa}(s_1) > 0$. Hence we may write $s_1 = f \star t_1 \star \cdots \star t_i$ and $t = f_{i+1}(t'_1, \ldots, t'_i, s'_2)$ with $t_j \to^*_{\mathcal{U}} t'_j$ for all $1 \leqslant j \leqslant i$ and $s_2 \to^*_{\mathcal{U}} s'_2$. Clearly, $t'_j \trianglerighteq u$ for some $1 \leqslant j \leqslant i$ or $s'_2 \trianglerighteq t$. In all cases the result follows with the same reasoning as in the first case. □

The next lemma states that innermost root steps in an ATRS can be simulated by a (non-empty) sequence of innermost steps in $\mathcal{U}^+_\eta(\mathcal{R})$. Note that $\xrightarrow{\mathrm{i}}_{\mathcal{U}^+_\eta(\mathcal{R})}$ means innermost reduction with respect to all rules in $\mathcal{U}^+_\eta(\mathcal{R})$.

Lemma 10 *Let $\mathcal{R}$ be an ℓ-ATRS. If w is a term over the signature of $\mathcal{R}$ then $s \overset{*}{\mathcal{U}}\!\leftarrow w \overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{R}}\, t$ implies $s \overset{\mathsf{i}}{\to}\!{}^{+}_{\mathcal{U}_\eta^+(\mathcal{R})} \cdot \overset{*}{\mathcal{U}}\!\leftarrow t$.*

Proof We prove that $s \overset{\mathsf{i}}{\to}\!{}^{+}_{\mathcal{U}_\eta^+(\mathcal{R})} r{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \overset{*}{\mathcal{U}}\!\leftarrow r\sigma$ whenever $s \overset{*}{\mathcal{U}}\!\leftarrow \ell\sigma \overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{R}}\, r\sigma$ for some rewrite rule $\ell \to r$ in $\mathcal{R}$. By Lemma 4 and the confluence of $\mathcal{U}$,

$$s \overset{\mathsf{i}}{\to}\!{}^{*}_{\mathcal{U}}\, (\ell\sigma){\downarrow_{\mathcal{U}}} = \ell{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \to_{\mathcal{U}_\eta^+(\mathcal{R})} r{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \overset{*}{\mathcal{U}}\!\leftarrow r\sigma$$

It remains to show that the sequence $s \overset{\mathsf{i}}{\to}\!{}^{*}_{\mathcal{U}}\, (\ell\sigma){\downarrow_{\mathcal{U}}}$ and the step $\ell{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}} \to_{\mathcal{U}_\eta^+(\mathcal{R})} r{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}}$ are innermost with respect to $\mathcal{U}_\eta^+(\mathcal{R})$.

- For the former, let $s \overset{\mathsf{i}}{\to}\!{}^{*}_{\mathcal{U}}\, C[u] \overset{\mathsf{i}}{\to}_{\mathcal{U}} C[u'] \overset{\mathsf{i}}{\to}\!{}^{*}_{\mathcal{U}}\, (\ell\sigma){\downarrow_{\mathcal{U}}}$ with $u \overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{U}}\, u'$ and let t be a proper subterm of u. Obviously $\ell\sigma \to^*_{\mathcal{U}} C[u] \rhd t$. According to Lemma 9, $\ell\sigma \rhd v \to^*_{\mathcal{U}} t$ for some term v. Since $\ell\sigma \overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{R}}\, r\sigma$, the term v is a normal form of $\mathcal{R}$. Hence $t \in NF(\mathcal{R}_\eta{\downarrow_{\mathcal{U}}})$ by Lemma 8. Since $u \overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{U}}\, u'$, t is also a normal form of $\mathcal{U}$. Hence $t \in NF(\mathcal{U}_\eta^+(\mathcal{R}))$ as desired.
- For the latter, let t be a proper subterm of $(\ell\sigma){\downarrow_{\mathcal{U}}}$. According to Lemma 9, $\ell\sigma \rhd u \to^*_{\mathcal{U}} t$. The term u is a normal form of $\mathcal{R}$. Hence $t \in NF(\mathcal{R}_\eta{\downarrow_{\mathcal{U}}})$ by Lemma 8. Obviously, $t \in NF(\mathcal{U})$ and thus also $t \in NF(\mathcal{U}_\eta^+(\mathcal{R}))$. □

The next example shows that it is not sound to replace $\overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{R}}$ by $\overset{\mathsf{i}}{\to}_{\mathcal{R}}$ in Lemma 10.

Example 15 Consider the ATRS $\mathcal{R}$

$$\mathsf{f} \to \mathsf{g} \qquad\qquad \mathsf{f}\ x \to \mathsf{g}\ x \qquad\qquad \mathsf{a} \to \mathsf{b}$$

Then $\mathsf{f}_1(\mathsf{a}) \overset{*}{\mathcal{U}}\!\leftarrow \mathsf{f} \star \mathsf{a} \overset{\mathsf{i}}{\to}_{\mathcal{R}} \mathsf{g} \star \mathsf{a}$ but $\mathsf{f}_1(\mathsf{a}) \overset{\mathsf{i}}{\to}\!{}^{+}_{\mathcal{U}_\eta^+(\mathcal{R})} \cdot \overset{*}{\mathcal{U}}\!\leftarrow \mathsf{g} \star \mathsf{a}$ does not hold. To see the latter, consider the two reducts $\mathsf{g}_1(\mathsf{a})$ and $\mathsf{g} \star \mathsf{a}$ of $\mathsf{g} \star \mathsf{a}$ with respect to $\to^*_{\mathcal{U}}$. We have neither $\mathsf{f}_1(\mathsf{a}) \overset{\mathsf{i}}{\to}\!{}^{+}_{\mathcal{U}_\eta^+(\mathcal{R})} \mathsf{g}_1(\mathsf{a})$ nor $\mathsf{f}_1(\mathsf{a}) \overset{\mathsf{i}}{\to}\!{}^{+}_{\mathcal{U}_\eta^+(\mathcal{R})} \mathsf{g} \star \mathsf{a}$.

In order to extend Lemma 9 to non-root positions, we have to use *rightmost* innermost rewriting. This avoids the situation in the above example where parallel redexes become nested by uncurrying.

Lemma 11 *Let $\mathcal{R}$ be an ℓ-ATRS and t a term over the signature of $\mathcal{R}$. If $s \overset{*}{\mathcal{U}}\!\leftarrow t \overset{\mathsf{ri}}{\to}_{\mathcal{R}} u$ then $s \overset{\mathsf{i}}{\to}\!{}^{+}_{\mathcal{U}_\eta^+(\mathcal{R})} \cdot \overset{*}{\mathcal{U}}\!\leftarrow u$.*

Proof Let $s \overset{*}{\mathcal{U}}\!\leftarrow t = C[\ell\sigma] \overset{\mathsf{ri}}{\to}_{\mathcal{R}} C[r\sigma] = u$ with $\ell\sigma \overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{R}}\, r\sigma$. We use induction on the context C. If $C = \Box$ then $s \overset{*}{\mathcal{U}}\!\leftarrow t \overset{\mathsf{i}}{\to}\!{}^{\epsilon}_{\mathcal{R}}\, u$. Lemma 10 yields

$$s \overset{\mathsf{i}}{\to}\!{}^{+}_{\mathcal{U}_\eta^+(\mathcal{R})} \cdot \overset{*}{\mathcal{U}}\!\leftarrow u$$

For the induction step we consider two cases.

– Suppose $C = \Box \star s_1 \star \cdots \star s_n$ and $n > 0$. Since $\mathcal{R}$ is left head variable free, $\mathrm{aa}(\ell)$ is defined. If $\mathrm{aa}(\ell) = 0$ then

$$s = t' \star s'_1 \star \cdots \star s'_n \overset{*}{\underset{\mathcal{U}}{\leftarrow}} \ell\sigma \star s_1 \star \cdots \star s_n \xrightarrow{\mathsf{i}}_{\mathcal{R}} r\sigma \star s_1 \star \cdots \star s_n$$

with $t' \overset{*}{\underset{\mathcal{U}}{\leftarrow}} \ell\sigma$ and $s'_j \overset{*}{\underset{\mathcal{U}}{\leftarrow}} s_j$ for $1 \leqslant j \leqslant n$. The claim follows using Lemma 10 and the fact that innermost rewriting is closed under contexts. If $\mathrm{aa}(\ell) > 0$ then the head symbol of ℓ cannot be a variable. We have to consider two cases. In the case where the leftmost $\star$ symbol in C has not been uncurried we proceed as when $\mathrm{aa}(\ell) = 0$. If the leftmost $\star$ symbol of C has been uncurried, we reason as follows. We may write $\ell\sigma = f \star u_1 \star \cdots \star u_k$ where $k < \mathrm{aa}(f)$. We have $t = f \star u_1 \star \cdots \star u_k \star s_1 \star \cdots \star s_n$ and $u = r\sigma \star s_1 \star \cdots \star s_n$. There exists an i with $1 \leqslant i \leqslant \min\{\mathrm{aa}(f), k+n\}$ such that

$$s = f_i(u'_1, \ldots, u'_k, s'_1, \ldots, s'_{i-k}) \star s'_{i-k+1} \star \cdots \star s'_n$$

with $u'_j \overset{*}{\underset{\mathcal{U}}{\leftarrow}} u_j$ for $1 \leqslant j \leqslant k$ and $s'_j \overset{*}{\underset{\mathcal{U}}{\leftarrow}} s_j$ for $1 \leqslant j \leqslant n$. Because of rightmost innermost evaluation, the terms $u_1, \ldots, u_k, s_1, \ldots, s_n$ are normal forms of $\mathcal{R}$. According to Lemma 8 the terms $u'_1, \ldots, u'_k, s'_1, \ldots, s'_n$ are normal forms of $\mathcal{R}_\eta{\downarrow_{\mathcal{U}}}$. Since $i - k \leqslant \mathrm{aa}(\ell)$, $\mathcal{R}_\eta$ contains the rule

$$\ell \star x_1 \star \cdots \star x_{i-k} \to r \star x_1 \star \cdots \star x_{i-k}$$

where $x_1, \ldots, x_{i-k}$ are pairwise distinct variables not occurring in $\ell \to r$. Hence the substitution $\tau = \sigma \cup \{x_1 \mapsto s_1, \ldots, x_{i-k} \mapsto s_{i-k}\}$ is well-defined. We obtain

$$\begin{aligned}
s &\xrightarrow{\mathsf{i}}{}^*_{\mathcal{U}_\eta^+(\mathcal{R})} f_i(u_1{\downarrow_{\mathcal{U}}}, \ldots, u_k{\downarrow_{\mathcal{U}}}, s_1{\downarrow_{\mathcal{U}}}, \ldots, s_{i-k}{\downarrow_{\mathcal{U}}}) \star s'_{i-k+1} \star \cdots \star s'_n \\
&\xrightarrow{\mathsf{i}}_{\mathcal{U}_\eta^+(\mathcal{R})} (r \star x_1 \star \cdots \star x_{i-k}){\downarrow_{\mathcal{U}}}\tau{\downarrow_{\mathcal{U}}} \star s'_{i-k+1} \star \cdots \star s'_n \\
&\overset{*}{\underset{\mathcal{U}}{\leftarrow}} \quad (r \star x_1 \star \cdots \star x_{i-k})\tau \star s_{i-k+1} \star \cdots \star s_n \\
&= \quad r\sigma \star s_1 \star \cdots \star s_n \;=\; u
\end{aligned}$$

where we use the confluence of $\mathcal{U}$ in the first sequence.

– In the second case we have $C = s_1 \star C'$. Clearly $C'[\ell\sigma] \xrightarrow{\mathsf{ri}}_{\mathcal{R}} C'[r\sigma]$. If $\mathrm{aa}(s_1) \leqslant 0$ or if $\mathrm{aa}(s_1)$ is undefined or if $\mathrm{aa}(s_1) > 0$ and the outermost $\star$ has not been uncurried in the sequence from t to s then

$$s = s'_1 \star s' \overset{*}{\underset{\mathcal{U}}{\leftarrow}} s_1 \star C'[\ell\sigma] \xrightarrow{\mathsf{ri}}_{\mathcal{R}} s_1 \star C'[r\sigma] = u$$

with $s'_1 \overset{*}{\underset{\mathcal{U}}{\leftarrow}} s_1$ and $s' \overset{*}{\underset{\mathcal{U}}{\leftarrow}} C'[\ell\sigma]$. If $\mathrm{aa}(s_1) > 0$ and the outermost $\star$ has been uncurried in the sequence from t to s then we may write $s_1 = f \star u_1 \star \cdots \star u_k$ where $k < \mathrm{aa}(f)$. We have $s = f_{k+1}(u'_1, \ldots, u'_k, s')$ for some term s' with $s' \overset{*}{\underset{\mathcal{U}}{\leftarrow}} C'[\ell\sigma]$ and $u'_i \overset{*}{\underset{\mathcal{U}}{\leftarrow}} u_i$ for $1 \leqslant i \leqslant k$. In both cases the induction hypothesis yields

$$s' \xrightarrow{\mathsf{i}}{}^+_{\mathcal{U}_\eta^+(\mathcal{R})} \cdot \overset{*}{\underset{\mathcal{U}}{\leftarrow}} C'[r\sigma]$$

and, since innermost rewriting is closed under contexts, we obtain

$$s \xrightarrow{\mathsf{i}}{}^+_{\mathcal{U}_\eta^+(\mathcal{R})} \cdot \overset{*}{\underset{\mathcal{U}}{\leftarrow}} u$$

as desired. □

We are now ready for the result that uncurrying reflects innermost termination.

Theorem 7 *An ℓ-ATRS $\mathcal{R}$ is innermost terminating if $\mathcal{U}^+_\eta(\mathcal{R})$ is innermost terminating.*

Proof For a proof by contradiction assume an infinite sequence

$$t_1 \xrightarrow{\mathsf{ri}}_{\mathcal{R}} t_2 \xrightarrow{\mathsf{ri}}_{\mathcal{R}} t_3 \xrightarrow{\mathsf{ri}}_{\mathcal{R}} \cdots$$

Using Lemma 11 this sequence can be transformed into

$$t_1{\downarrow_{\mathcal{U}}} \xrightarrow{\mathsf{i}}{}^+_{\mathcal{U}^+_\eta(\mathcal{R})} t'_2 \xrightarrow{\mathsf{i}}{}^+_{\mathcal{U}^+_\eta(\mathcal{R})} t'_3 \xrightarrow{\mathsf{i}}{}^+_{\mathcal{U}^+_\eta(\mathcal{R})} \cdots$$

for terms $t'_2, t'_3, \ldots$ such that $t_i \to^*_{\mathcal{U}} t'_i$ for $i \geqslant 2$. The proof concludes by the fact that innermost termination is equivalent to rightmost innermost termination, a result due to Krishna Rao [26]. □

5 Derivational Complexity

Next we investigate how the uncurrying transformation affects derivational complexity for full (Section 5.1) and innermost rewriting (Section 5.2).

5.1 Full Rewriting

The next theorem explains why uncurrying can be used as a preprocessor for proving upper bounds on the derivational complexity.

Theorem 8 *If $\mathcal{R}$ is a terminating ℓ-ATRS then* $\mathrm{dc}_{\mathcal{R}}(n) \in \mathcal{O}(\mathrm{dc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n))$.

Proof Consider an arbitrary maximal rewrite sequence in $\mathcal{R}$ starting from t_0

$$t_0 \to_{\mathcal{R}} t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \cdots \to_{\mathcal{R}} t_m$$

Using the proof of Theorem 3, we can transform the sequence into

$$t_0{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} t_1{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} t_2{\downarrow_{\mathcal{U}}} \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} \cdots \to^+_{\mathcal{U}^+_\eta(\mathcal{R})} t_m{\downarrow_{\mathcal{U}}}$$

Moreover, $t_0 \to^*_{\mathcal{U}^+_\eta(\mathcal{R})} t_0{\downarrow_{\mathcal{U}}}$ holds. Therefore, $\mathrm{dh}(t_0, \to_{\mathcal{R}}) \leqslant \mathrm{dh}(t_0, \to_{\mathcal{U}^+_\eta(\mathcal{R})})$. Hence $\mathrm{dc}_{\mathcal{R}}(n) \leqslant \mathrm{dc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n)$ holds for all $n \in \mathbb{N}$, showing the result. □

Next we show that uncurrying preserves polynomial complexity. Since any duplicating TRS has at least exponential derivational complexity (cf. [19]), we only deal with non-duplicating TRSs. Furthermore we ignore pathological systems that yield constant derivational complexity (note that any non-empty ATRS admits at least derivations linear in the size of the starting term).

A TRS $\mathcal{R}$ is called *length-reducing* if $\mathcal{R}$ is non-duplicating and $|\ell| > |r|$ for all rules $\ell \to r \in \mathcal{R}$. The following lemma is an easy consequence of [19, Theorem 23]. Below, $\to_{\mathcal{R}/\mathcal{S}}$ denotes $\to^*_{\mathcal{S}} \cdot \to_{\mathcal{R}} \cdot \to^*_{\mathcal{S}}$.

Lemma 12 *Let $\mathcal{R}$ be a non-empty and non-duplicating TRS over a signature containing a function symbol of arity at least two. If a TRS $\mathcal{S}$ is length-reducing, $\mathrm{dc}_{\mathcal{R}\cup\mathcal{S}}(n) \in \mathcal{O}(\mathrm{dc}_{\mathcal{R}/\mathcal{S}}(n))$ holds whenever $\mathcal{R}\cup\mathcal{S}$ is terminating.* □

Note that the above lemma does not hold if the TRS $\mathcal{R}$ is empty.

Theorem 9 *Let $\mathcal{R}$ be a non-empty, non-duplicating, and terminating ℓ-ATRS. If $\mathrm{dc}_{\mathcal{R}}(n)$ is in $\mathcal{O}(n^k)$ then $\mathrm{dc}_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}}(n)$ and $\mathrm{dc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n)$ are in $\mathcal{O}(n^k)$.*

Proof Suppose that $\mathrm{dc}_{\mathcal{R}}(n)$ is in $\mathcal{O}(n^k)$. First consider a maximal rewrite sequence of $\to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}}$ starting from a term t_0:

$$t_0 \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}} t_1 \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}} \cdots \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}} t_m$$

By Lemmata 7 and 1 we obtain the sequence

$$t_0\downarrow_{\mathcal{C}'} \to_{\mathcal{R}} t_1\downarrow_{\mathcal{C}'} \to_{\mathcal{R}} \cdots \to_{\mathcal{R}} t_m\downarrow_{\mathcal{C}'}$$

Thus, $\mathrm{dh}(t_0, \to_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}}) \leqslant \mathrm{dh}(t_0\downarrow_{\mathcal{C}'}, \to_{\mathcal{R}})$. Because $|t_0\downarrow_{\mathcal{C}'}| \leqslant 2|t_0|$ holds, we obtain $\mathrm{dc}_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}}(n) \leqslant \mathrm{dc}_{\mathcal{R}}(2n)$. From the assumption the right-hand side is in $\mathcal{O}(n^k)$, Therefore, $\mathrm{dc}_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}}(n)$ is in $\mathcal{O}(n^k)$. Because $\mathcal{U}$ is length-reducing, $\mathrm{dc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n)$ is also in $\mathcal{O}(n^k)$, by Lemma 12. □

In practice it is recommendable to investigate $\mathrm{dc}_{\mathcal{R}_\eta\downarrow_{\mathcal{U}}/\mathcal{U}}(n)$ instead of $\mathrm{dc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n)$, see [45]. The next example shows that uncurrying might be useful to enable criteria for polynomial complexity.

Example 16 Consider the ATRS $\mathcal{R}$ consisting of the rules

$$\mathsf{add}\ x\ 0 \to x \qquad\qquad \mathsf{add}\ x\ (\mathsf{s}\ y) \to \mathsf{s}\ (\mathsf{add}\ x\ y)$$

The system $\mathcal{U}^+_\eta(\mathcal{R})$ consists of the rules

$$\mathsf{add}_2(x,0) \to x \qquad\qquad \mathsf{add}_2(x,\mathsf{s}_1(y)) \to \mathsf{s}_1(\mathsf{add}_2(x,y))$$
$$\mathsf{add}_1(x) \star y \to \mathsf{add}_2(x,y) \qquad \mathsf{add} \star x \to \mathsf{add}_1(x) \qquad \mathsf{s} \star x \to \mathsf{s}_1(x)$$

It is easy to see that the following TMI $\mathcal{M}$ of dimension 2 below orients all rules in $\mathcal{U}^+_\eta(\mathcal{R})$ strictly, inducing a quadratic bound on the derivational complexity of $\mathcal{U}^+_\eta(\mathcal{R})$ (according to Theorem 1):

$$\mathsf{add}_{1\,\mathcal{M}}(x) = \mathsf{s}_{1\,\mathcal{M}}(x) = \begin{pmatrix}1&0\\0&1\end{pmatrix} x + \begin{pmatrix}0\\1\end{pmatrix}$$
$$\mathsf{add}_{2\,\mathcal{M}}(x,y) = \star_{\mathcal{M}}(x,y) = \begin{pmatrix}1&1\\0&1\end{pmatrix} x + \begin{pmatrix}1&1\\0&1\end{pmatrix} y$$
$$\mathsf{s}_{\mathcal{M}} = 0_{\mathcal{M}} = \mathsf{add}_{\mathcal{M}} = \begin{pmatrix}0\\1\end{pmatrix}$$

Theorem 8 then establishes a quadratic bound on the derivational complexity of $\mathcal{R}$. In contrast to $\mathcal{U}^+_\eta(\mathcal{R})$, the ATRS $\mathcal{R}$ itself does not admit such an interpretation of dimension 2. To see this, we encoded the required condition as a satisfaction problem in non-linear arithmetic over the integers. MiniSmt [46] can prove this problem unsatisfiable by simplifying it into a trivially unsatisfiable constraint. Details can be inferred from the website mentioned in Footnote 4 on page 32.

5.2 Innermost Rewriting

Next we consider innermost derivational complexity. Let $\mathcal{R}$ be an innermost terminating TRS. From a result by Krishna Rao [26, Section 5.1] which has been generalised by van Oostrom [36, Theorems 2 and 3] we infer that

$$\mathrm{dh}(t, \xrightarrow{\mathsf{i}}_{\mathcal{R}}) = \mathrm{dh}(t, \xrightarrow{\mathsf{ri}}_{\mathcal{R}})$$

holds for all terms t.

Theorem 10 *Let $\mathcal{R}$ be an innermost terminating ℓ-ATRS. Then* $\mathrm{idc}_{\mathcal{R}}(n) \in \mathcal{O}(\mathrm{idc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n))$.

Proof Consider a maximal rightmost innermost rewrite sequence

$$t_0 \xrightarrow{\mathsf{ri}}_{\mathcal{R}} t_1 \xrightarrow{\mathsf{ri}}_{\mathcal{R}} t_2 \xrightarrow{\mathsf{ri}}_{\mathcal{R}} \cdots \xrightarrow{\mathsf{ri}}_{\mathcal{R}} t_m$$

Using Lemma 11 the sequence can be transformed into

$$t_0 \xrightarrow{\mathsf{i}+}_{\mathcal{U}^+_\eta(\mathcal{R})} t'_1 \xrightarrow{\mathsf{i}+}_{\mathcal{U}^+_\eta(\mathcal{R})} t'_2 \xrightarrow{\mathsf{i}+}_{\mathcal{U}^+_\eta(\mathcal{R})} \cdots \xrightarrow{\mathsf{i}+}_{\mathcal{U}^+_\eta(\mathcal{R})} t'_m$$

for terms $t'_1, t'_2, \ldots, t'_m$ such that $t_i \to^*_{\mathcal{U}} t'_i$ for all $1 \leqslant i \leqslant m$. Thus,

$$\mathrm{dh}(t_0, \xrightarrow{\mathsf{i}}_{\mathcal{R}}) = \mathrm{dh}(t_0, \xrightarrow{\mathsf{ri}}_{\mathcal{R}}) \leqslant \mathrm{dh}(t_0, \xrightarrow{\mathsf{i}}_{\mathcal{U}^+_\eta(\mathcal{R})}).$$

Hence, we conclude $\mathrm{idc}_{\mathcal{R}}(n) \in \mathcal{O}(\mathrm{idc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n))$. □

As Example 11 on page 15 showed, uncurrying does not preserve innermost termination. Similarly, it does not preserve innermost polynomial complexity even if the original ATRS has linear derivational complexity.

Example 17 Consider the non-duplicating ATRS $\mathcal{R}$

$$\mathsf{f} \to \mathsf{s} \qquad\qquad \mathsf{f}\ (\mathsf{s}\ x) \to \mathsf{s}\ (\mathsf{s}\ (\mathsf{f}\ x))$$

Since the right rule is never used in innermost rewriting, $\mathrm{idc}_{\mathcal{R}}(n) \in \mathcal{O}(n)$ is shown by easy induction on n. We show that the innermost derivational complexity of $\mathcal{U}^+_\eta(\mathcal{R})$ is at least exponential. The TRS $\mathcal{U}^+_\eta(\mathcal{R})$ consists of the five rules:

$$\begin{array}{ccc} \mathsf{f} \to \mathsf{s} & \mathsf{f}_1(\mathsf{s}_1(x)) \to \mathsf{s}_1(\mathsf{s}_1(\mathsf{f}_1(x))) & \mathsf{f} \star x \to \mathsf{f}_1(x) \\ \mathsf{f}_1(x) \to \mathsf{s}_1(x) & & \mathsf{s} \star x \to \mathsf{s}_1(x) \end{array}$$

One can verify

$$\mathrm{dh}(\overbrace{\mathsf{f}_1(\cdots(\mathsf{f}_1}^{n}(\mathsf{s}_1(x)))), \xrightarrow{\mathsf{i}}_{\mathcal{U}^+_\eta(\mathcal{R})}) \geqslant 2^n$$

for all $n \geqslant 1$. Hence $\mathrm{idc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n+3) \geqslant 2^n$ for all $n \geqslant 0$.

Similar to Theorem 6, the result can be recovered for locally confluent overlay systems. In the sequel a substitution σ is called normalised (for a TRS $\mathcal{R}$) if $x\sigma \in NF(\mathcal{R})$ for all $x \in \mathcal{V}$. The next lemmata state useful properties that prepare for the proof. The first of these states a trivial diamond-like property of innermost rewriting. It is used in the proofs of the next lemmata to rearrange innermost rewrite sequences such that the number of steps is preserved.

Lemma 13 *Let $\mathcal{R}$ be a TRS. If $s \xrightarrow{\mathsf{i}} t$ and $s \xrightarrow{\mathsf{i}} u$ by rewriting innermost redexes at parallel positions then $t \xrightarrow{\mathsf{i}} v$ and $u \xrightarrow{\mathsf{i}} v$ for some term v.* □

Lemma 14 *If $t\sigma \xrightarrow{\mathsf{i}}{}^{n} u \in NF(\mathcal{R})$ then $t\sigma \xrightarrow{\mathsf{i}}{}^{n_1} t\tau \xrightarrow{\mathsf{i}}{}^{n_2} u$ for some normalised substitution τ and $n_1, n_2 \in \mathbb{N}$ with $n_1 + n_2 = n$.*

Proof We use induction on n. Since the base case is trivial, we consider the inductive step. Suppose $t\sigma \xrightarrow{\mathsf{i}}{}^{n} u \in NF(\mathcal{R})$. Without loss of generality we assume $\mathcal{D}\mathsf{om}(\sigma) \subseteq \mathcal{V}\mathsf{ar}(t)$. We proceed by a case distinction. If σ is normalised then the claim follows with $n_1 = 0$, $n_2 = n$, and $\tau = \sigma$. In the other case, by Lemma 13 we can reorder the sequence such that the first rewrite step takes place in the substitution part. Therefore, there exists a substitution σ' with $x\sigma \xrightarrow{\mathsf{i}} x\sigma'$ for some $x \in \mathcal{D}\mathsf{om}(\sigma)$ and $y\sigma = y\sigma'$ for all $y \in \mathcal{D}\mathsf{om}(\sigma)\backslash\{x\}$. Writing k for the number of occurrences of x in t, we have $t\sigma \xrightarrow{\mathsf{i}}{}^{k} t\sigma' \xrightarrow{\mathsf{i}}{}^{n-k} u$. Since $k \geqslant 1$, the induction hypothesis applied to $t\sigma' \xrightarrow{\mathsf{i}}{}^{n-k} u$ yields $t\sigma' \xrightarrow{\mathsf{i}}{}^{m_1} t\tau \xrightarrow{\mathsf{i}}{}^{m_2} u$ with $m_1 + m_2 = n - k$ and normalised τ. Combining this with $t\sigma \xrightarrow{\mathsf{i}}{}^{k} t\sigma'$, $t\sigma \xrightarrow{\mathsf{i}}{}^{k+m_1} t\tau \xrightarrow{\mathsf{i}}{}^{m_2} u$ is obtained. By taking $n_1 = k + m_1$ and $n_2 = m_2$, we obtain $n_1 + n_2 = k + m_1 + m_2 = k + (n - k) = n$ which proves the claim. □

Lemma 15 *Let $\mathcal{R}$ be a non-duplicating overlay system. If $t \to^{m} u \in NF(\mathcal{R})$ then $t \xrightarrow{\mathsf{i}}{}^{n} u$ for some $n \geqslant m$.*

Proof We use induction on m. Since the base case is trivial, we consider the inductive step. Suppose

$$C[\ell\sigma] \to C[r\sigma] \to^{m} u \in NF(\mathcal{R})$$

with $\ell \to r \in \mathcal{R}$. The induction hypothesis yields $C[r\sigma] \xrightarrow{\mathsf{i}}{}^{n} u$ for some $n \geqslant m$. By Lemma 13 this sequence can be written as $C[r\sigma] \xrightarrow{\mathsf{i}}{}^{n_1} C[u'] \xrightarrow{\mathsf{i}}{}^{n_2} u$ with $n = n_1 + n_2$ and $u' \in NF(\mathcal{R})$. From Lemma 14 we obtain a normalised τ and $m_1, m_2 \in \mathbb{N}$ with $m_1 + m_2 = n_1$ such that $r\sigma \xrightarrow{\mathsf{i}}{}^{m_1} r\tau \xrightarrow{\mathsf{i}}{}^{m_2} u'$. Because innermost rewriting is closed under contexts, $C[r\sigma] \xrightarrow{\mathsf{i}}{}^{m_1} C[r\tau] \xrightarrow{\mathsf{i}}{}^{m_2+n_2} u$. Since $\mathcal{R}$ is non-duplicating, $C[\ell\sigma] \xrightarrow{\mathsf{i}}{}^{m_0} C[\ell\tau]$ for some $m_0 \geqslant m_1$. Because $\mathcal{R}$ is an overlay system and τ is normalised $\ell\tau \xrightarrow{\mathsf{i}} r\tau$. Hence

$$C[\ell\sigma] \xrightarrow{\mathsf{i}}{}^{m_0} C[\ell\tau] \xrightarrow{\mathsf{i}} C[r\tau] \xrightarrow{\mathsf{i}}{}^{m_2+n_2} u$$

Here $m_0 + 1 + m_2 + n_2 \geqslant m_1 + 1 + m_2 + n_2 = n_1 + 1 + n_2 = n + 1 \geqslant m + 1$. □

By the above lemma the next theorem is obtained.

Theorem 11 *If $\mathcal{R}$ is a non-duplicating and terminating overlay system then* $\mathrm{idc}_{\mathcal{R}}(n) = \mathrm{dc}_{\mathcal{R}}(n)$.

Proof Since $\mathcal{R}$ is terminating, Lemma 15 yields $\mathrm{dh}(t, \xrightarrow{\mathsf{i}}_{\mathcal{R}}) \geqslant \mathrm{dh}(t, \to_{\mathcal{R}})$. Combining this with the obvious $\mathrm{dh}(t, \xrightarrow{\mathsf{i}}_{\mathcal{R}}) \leqslant \mathrm{dh}(t, \to_{\mathcal{R}})$ concludes the proof. □

Using Gramlich's [15] result on the equivalence of termination and innermost termination for locally confluent overlay systems we obtain the following corollary from Theorems 9 and 11.

Corollary 2 *Let $\mathcal{R}$ be a non-duplicating, innermost terminating, and locally confluent overlay ℓ-ATRS. If* $\mathrm{idc}_{\mathcal{R}}(n)$ *is in* $\mathcal{O}(n^k)$ *then* $\mathrm{idc}_{\mathcal{U}^+_\eta(\mathcal{R})}(n)$ *is in* $\mathcal{O}(n^k)$. □

6 Uncurrying with Dependency Pairs

In this section we incorporate the uncurrying transformation into the dependency pair framework [4, 12, 14, 17, 40].

In the sequel we present two DP processors that uncurry applicative DP problems, which are DP problems over signatures containing constants and two application symbols: $\star$ and $\star^\sharp$. Properties for full termination of these processors are studied in Section 6.1 while innermost termination is considered in Section 6.2.

First we define a suitable set of uncurrying rules for DP problems. Let $(\mathcal{P}, \mathcal{R})$ be an applicative DP problem. Here the applicative arities of function symbols are computed with respect to $\star$ and $\mathcal{P} \cup \mathcal{R}$. By $\mathcal{U}(\mathcal{P}, \mathcal{R})$ we denote the uncurrying rules derived from $\mathcal{U}(\mathcal{P}) \cup \mathcal{U}(\mathcal{R})$. In this section we assume that $\mathcal{F}$ is $\mathcal{F}\mathsf{un}(\mathcal{P} \cup \mathcal{R})$, write $\mathcal{U}(\mathcal{F})$ for $\mathcal{U}(\mathcal{P}, \mathcal{R})$ and let $\mathcal{U}^+_\eta(\mathcal{R}, \mathcal{F})$ denote $\mathcal{R}_\eta{\downarrow_{\mathcal{U}(\mathcal{F})}} \cup \mathcal{U}(\mathcal{F})$. If no confusion can arise, $\mathcal{F}$ is dropped in $\mathcal{U}(\mathcal{F})$ and $\mathcal{U}^+_\eta(\mathcal{R}, \mathcal{F})$. An applicative DP problem $(\mathcal{P}, \mathcal{R})$ is said to be ℓ-applicative if $\mathcal{P} \cup \mathcal{R}$ is left head variable free.

Definition 9 Let $(\mathcal{P}, \mathcal{R})$ be a DP problem. The DP processor $\mathcal{U}_1$ is defined as

$$(\mathcal{P}, \mathcal{R}) \mapsto \begin{cases} \{(\mathcal{P}{\downarrow_{\mathcal{U}(\mathcal{F})}}, \mathcal{U}^+_\eta(\mathcal{R}, \mathcal{F}))\} & \text{if } (\mathcal{P}, \mathcal{R}) \text{ is } \ell\text{-applicative} \\ \{(\mathcal{P}, \mathcal{R})\} & \text{otherwise} \end{cases}$$

where $\mathcal{F} = \mathcal{F}\mathsf{un}(\mathcal{P} \cup \mathcal{R})$.

Example 18 Consider the ℓ-applicative (note that it is left head variable free) DP problem $(\{x^\sharp\,(\mathsf{a}\;\mathsf{a}) \to (\mathsf{a}\;\mathsf{a}\;\mathsf{a})^\sharp\,x\}, \varnothing)$. Processor $\mathcal{U}_1$ transforms it into the problem $(\{x \star^\sharp (\mathsf{a}_1(\mathsf{a})) \to \mathsf{a}_2(\mathsf{a}, \mathsf{a}) \star^\sharp x\}, \{\mathsf{a} \star x \to \mathsf{a}_1(x), \mathsf{a}_1(x) \star y \to \mathsf{a}_2(x, y)\})$ because the applicative arity of a is two. The latter DP problem is easily shown finite by a matrix interpretation of dimension one counting the symbols $\star$ and a_1.

A drawback of $\mathcal{U}_1$ is that dependency pair symbols are excluded from the uncurrying process. Typically, all pairs in $\mathcal{P}$ have the same root symbol $\star^\sharp$. The next example shows that uncurrying root symbols of $\mathcal{P}$ can be beneficial.

Example 19 Consider the ATRS consisting of the single rule $\mathsf{a}\ x\ \mathsf{a} \to \mathsf{a}\ (\mathsf{a}\ \mathsf{a})\ x$. After processing the only SCC in the dependency graph with $\mathcal{U}_1$, the rule $\mathsf{a}_1(x)\ \star^\sharp\ \mathsf{a} \to \mathsf{a}_1(\mathsf{a}_1(\mathsf{a}))\ \star^\sharp\ x$ must be oriented. This cannot be done with a matrix interpretation of dimension one nor with a reduction pair based on any other simplification order. If we transform the rule into $\mathsf{a}_2^\sharp(x, \mathsf{a}) \to \mathsf{a}_2^\sharp(\mathsf{a}_1(\mathsf{a}), x)$ this becomes trivial.

To this end we introduce a simple variant of *freezing* [43].

Definition 10 A *simple freeze* is a partial mapping $\circledast$ that assigns to a function symbol of arity $n > 0$ an argument position $i \in \{1, \dots, n\}$. Every simple freeze $\circledast$ induces the following partial mapping on non-variable terms $t = f(t_1, \dots, t_n)$, also denoted by $\circledast$:

- if $\circledast(f)$ is undefined or $n = 0$ then $\circledast(t) = t$,
- if $\circledast(f) = i$ and $t_i = g(u_1, \dots, u_m)$ then

$$\circledast(t) = \circledast_{fg}(t_1, \dots, t_{i-1}, u_1, \dots, u_m, t_{i+1}, \dots, t_n)$$

  where $\circledast_{fg}$ is a fresh function symbol of arity $m + n - 1$,
- if $\circledast(f) = i$ and t_i is a variable then $\circledast(t)$ is undefined.

We denote $\{\circledast(\ell) \to \circledast(r) \mid \ell \to r \in \mathcal{R}\}$ by $\circledast(\mathcal{R})$.

Now uncurrying for dependency pair symbols is formulated with the simple freeze $\circledast(\star^\sharp) = 1$, transforming $f_n(t_1, \dots, t_n)\ \star^\sharp\ t_{n+1}$ to $\circledast_{\star^\sharp f_n}(t_1, \dots, t_n, t_{n+1})$. Writing $f_{n+1}^\sharp$ for $\circledast_{\star^\sharp f_n}$, we obtain the frozen term $f_{n+1}^\sharp(t_1, \dots, t_n, t_{n+1})$. In Example 19 we have

$$\circledast(\{\mathsf{a}_1(x)\ \star^\sharp\ \mathsf{a} \to \mathsf{a}_1(\mathsf{a}_1(\mathsf{a}))\ \star^\sharp\ x\}) = \{\mathsf{a}_2^\sharp(x, \mathsf{a}) \to \mathsf{a}_2^\sharp(\mathsf{a}_1(\mathsf{a}), x)\}$$

The next definition introduces a condition that remedies that freezing is not sound in general (cf. Example 20).

Definition 11 A term t is *strongly root stable* with respect to a TRS $\mathcal{R}$ if $t\sigma \to_\mathcal{R}^* \cdot \to_\mathcal{R}^\epsilon u$ does not hold for any substitution σ and term u. Let $\circledast$ be a simple freeze. A DP problem $(\mathcal{P}, \mathcal{R})$ is *$\circledast$-stable* if $\circledast(\mathcal{P})$ is well-defined and t_i is strongly root stable for $\mathcal{R}$ whenever $s \to f(t_1, \dots, t_n) \in \mathcal{P}$ and $\circledast(f) = i$.

Definition 12 Let $(\mathcal{P}, \mathcal{R})$ be a DP problem and $\circledast$ a simple freeze. The DP processor $\circledast$ is defined as

$$(\mathcal{P}, \mathcal{R}) \mapsto \begin{cases} \{(\circledast(\mathcal{P}), \mathcal{R})\} & \text{if } (\mathcal{P}, \mathcal{R}) \text{ is } \circledast\text{-stable} \\ \{(\mathcal{P}, \mathcal{R})\} & \text{otherwise} \end{cases}$$

Furthermore, the DP processor $\mathcal{U}_2$ is defined as

$$(\mathcal{P}, \mathcal{R}) \mapsto \{\circledast((\mathcal{P}', \mathcal{R}')) \mid (\mathcal{P}', \mathcal{R}') \in \mathcal{U}_1((\mathcal{P}, \mathcal{R}))\}$$

where $\circledast(\star^\sharp) = 1$.

The DP processor $\circledast$ exploits the fact that a root step in $\mathcal{P}$ gives rise to a root step in $\circledast(\mathcal{P})$ and vice versa. This follows from the root stability of the left argument of left-hand sides rooted by $\star^\sharp$, which is a consequence of the $\circledast$-stability of $(\mathcal{P},\mathcal{R})$. Moreover, $t \to^*_{\mathcal{R}} u$ if and only if $\circledast(t) \to^*_{\circledast(\mathcal{R})} \circledast(u)$ because $\star^\sharp$ does not occur in the rules of $\mathcal{R}$.

6.1 Full Termination

Recently it has been observed by Sternagel and Thiemann [37] that the signature influences whether a DP problem is finite or not. In particular, restricting the signature of a non-finite DP problem $(\mathcal{P},\mathcal{R})$ to the function symbols that occur in $\mathcal{P} \cup \mathcal{R}$ may make it finite. This is in sharp contrast to (innermost) termination of TRSs [29] (and *innermost non-finiteness* of DP problems, cf. Lemma 19).

We first show that for ℓ-applicative DP problems this cannot happen.[2]

Lemma 16 *Let $(\mathcal{P},\mathcal{R})$ be an ℓ-applicative DP problem over the signature $\mathcal{G}$. If $(\mathcal{P},\mathcal{R})$ is finite over $\mathcal{F}$ and $\star \in \mathcal{F}$ then $(\mathcal{P},\mathcal{R})$ is finite over $\mathcal{G}$.*

Proof Let $\mathcal{V}' = \mathcal{V} \uplus \{x_f \mid f \in \mathcal{G} \setminus \mathcal{F}\}$. We define a mapping I from $\mathcal{T}(\mathcal{G},\mathcal{V})$ to $\mathcal{T}(\mathcal{F},\mathcal{V}')$ as follows:

$$I(t) = \begin{cases} t & \text{if } t \in \mathcal{V} \\ f(I(t_1),\dots,I(t_n)) & \text{if } t = f(t_1,\dots,t_n) \text{ and } f \in \mathcal{F} \\ x_f \star I(t_1) \star \cdots \star I(t_n) & \text{if } t = f(t_1,\dots,t_n) \text{ and } f \in \mathcal{G} \setminus \mathcal{F} \end{cases}$$

Note that $I(u) = u$ for $u \in \mathcal{T}(\mathcal{F},\mathcal{V})$. Obviously $I(C[t\sigma]) = I(C)[I(t)I(\sigma)]$ and hence we immediately obtain that $s \to^\epsilon_{\mathcal{P}} t$ implies $I(s) \to^\epsilon_{\mathcal{P}} I(t)$ and that $s \to_{\mathcal{R}} t$ implies $I(s) \to_{\mathcal{R}} I(t)$ because $\ell = I(\ell)$ and $r = I(r)$ for all $\ell \to r \in \mathcal{P} \cup \mathcal{R}$. To show that any $I(t)$ is terminating with respect to $\mathcal{R}$ whenever $t \in \mathcal{T}(\mathcal{G},\mathcal{V})$ is terminating with respect to $\mathcal{R}$, we show that if $I(t) \to_{\mathcal{R}} u$ for some term $u \in \mathcal{T}(\mathcal{F},\mathcal{V}')$ then there exists a term $s \in \mathcal{T}(\mathcal{G},\mathcal{V})$ with $t \to_{\mathcal{R}} s$ and $I(s) = u$. Now let $I(t) = C[\ell\sigma] \to_{\mathcal{R}} C[r\sigma] = u$ for some $\ell \to r \in \mathcal{R}$. Clearly $t = C'[v\sigma']$ for some C', v, and σ' with $C = I(C')$, $\ell = I(v)$, and $\sigma = I(\sigma')$. By left head variable freeness of $\mathcal{R}$ we obtain $v = \ell \in \mathcal{T}(\mathcal{F},\mathcal{V})$. Hence by injectivity of I we conclude $t = C'[\ell\sigma'] \to_{\mathcal{R}} C'[r\sigma'] = s$. These observations guarantee that any presupposed minimal sequence using terms from $\mathcal{T}(\mathcal{G},\mathcal{V})$ is transformed by the mapping I into a minimal sequence using terms from $\mathcal{T}(\mathcal{F},\mathcal{V}')$. It follows that $(\mathcal{P},\mathcal{R})$ is finite over $\mathcal{G}$. □

The next result prepares for the soundness proof of $\mathcal{U}_1$.

Lemma 17 *Let $(\mathcal{P},\mathcal{R})$ be an ℓ-applicative DP problem. If $\ell\sigma \to^\epsilon r\sigma$ with $\ell \to r \in \mathcal{P}$ then $(\ell\sigma)\downarrow_{\mathcal{U}(\mathcal{F})} \to^\epsilon_{\mathcal{P}\downarrow_{\mathcal{U}(\mathcal{F})}} r\downarrow_{\mathcal{U}(\mathcal{F})}\sigma\downarrow_{\mathcal{U}(\mathcal{F})}$.*

[2] An alternative proof has been independently obtained by Sternagel and Thiemann [38].

Proof By Lemma 4 and the assumption that ℓ is head variable free. □

Theorem 12 *The DP processor $\mathcal{U}_1$ is sound and complete.*

Proof Let $(\mathcal{P},\mathcal{R})$ be an ℓ-applicative DP problem. If $\star \notin \mathcal{F}$ then $\mathcal{U}_1((\mathcal{P},\mathcal{R})) = \{(\mathcal{P},\mathcal{R})\}$ and there is nothing to show. If $\star \in \mathcal{F}$ then according to the preceding lemma we may assume that the signature of the DP problem $(\mathcal{P},\mathcal{R})$ is $\mathcal{F}$.

We first show soundness. Suppose the DP problem $(\mathcal{P}{\downarrow_{\mathcal{U}}},\mathcal{U}^+_\eta(\mathcal{R}))$ is finite. We have to show that $(\mathcal{P},\mathcal{R})$ is finite. Suppose to the contrary that $(\mathcal{P},\mathcal{R})$ is not finite. So there exists a minimal rewrite sequence

$$s_1 \to^\epsilon_{\mathcal{P}} t_1 \to^*_{\mathcal{R}} s_2 \to^\epsilon_{\mathcal{P}} t_2 \to^*_{\mathcal{R}} \cdots \tag{2}$$

By Lemmata 17 and 4 together with the claim in the proof of Theorem 3, this sequence can be transformed into

$$s_1{\downarrow_{\mathcal{U}}} \to^\epsilon_{\mathcal{P}{\downarrow_{\mathcal{U}}}} u_1 \to^*_{\mathcal{U}} t_1{\downarrow_{\mathcal{U}}} \to^*_{\mathcal{U}^+_\eta(\mathcal{R})} s_2{\downarrow_{\mathcal{U}}} \to^\epsilon_{\mathcal{P}{\downarrow_{\mathcal{U}}}} u_2 \to^*_{\mathcal{U}} t_2{\downarrow_{\mathcal{U}}} \to^*_{\mathcal{U}^+_\eta(\mathcal{R})} \cdots$$

It remains to show that all terms u_1, u_2, … are terminating with respect to $\mathcal{U}^+_\eta(\mathcal{R})$. Fix i. We have $u_i{\downarrow_{\mathcal{C}'}} = t_i{\downarrow_{\mathcal{U}}}{\downarrow_{\mathcal{C}'}} = t_i$. Due to the minimality of (2), t_i is terminating with respect to $\mathcal{R}$ and, according to Lemma 1, also with respect to $\mathcal{R}_\eta$. Hence, due to the proof of Theorem 4, u_i is terminating with respect to $\mathcal{U}^+_\eta(\mathcal{R})$.

Next we show completeness of the DP processor $\mathcal{U}_1$. Suppose that the DP problem $(\mathcal{P}{\downarrow_{\mathcal{U}}},\mathcal{U}^+_\eta(\mathcal{R}))$ is not finite. So there exists a minimal rewrite sequence

$$s_1 \to^\epsilon_{\mathcal{P}{\downarrow_{\mathcal{U}}}} t_1 \to^*_{\mathcal{U}^+_\eta(\mathcal{R})} s_2 \to^\epsilon_{\mathcal{P}{\downarrow_{\mathcal{U}}}} t_2 \to^*_{\mathcal{U}^+_\eta(\mathcal{R})} \cdots$$

Using Lemmata 6 and 7 this sequence can be transformed into

$$s_1{\downarrow_{\mathcal{C}'}} \to^\epsilon_{\mathcal{P}} t_1{\downarrow_{\mathcal{C}'}} \to^*_{\mathcal{R}_\eta} s_2{\downarrow_{\mathcal{C}'}} \to^\epsilon_{\mathcal{P}} t_2{\downarrow_{\mathcal{C}'}} \to^*_{\mathcal{R}_\eta} \cdots$$

In order to conclude that the DP problem $(\mathcal{P},\mathcal{R})$ is not finite, it remains to show that the terms $t_1{\downarrow_{\mathcal{C}'}}$, $t_2{\downarrow_{\mathcal{C}'}}$, … are terminating with respect to $\mathcal{R}_\eta$. This follows from the assumption that the terms t_1, t_2, … are terminating with respect to $\mathcal{U}^+_\eta(\mathcal{R})$ in connection with the proof of Theorem 3. An application of Lemma 1 concludes the proof. □

Theorem 13 *The DP processor $\circledast$ is sound and complete.*

Proof Let $(\mathcal{P},\mathcal{R})$ be a $\circledast$-stable DP problem. We show that every minimal rewrite sequence

$$s_1 \to^\epsilon_{\mathcal{P}} t_1 \to^*_{\mathcal{R}} s_2 \to^\epsilon_{\mathcal{P}} t_2 \to^*_{\mathcal{R}} \cdots$$

can be transformed into the minimal sequence

$$\circledast(s_1) \to^\epsilon_{\circledast(\mathcal{P})} \circledast(t_1) \to^*_{\mathcal{R}} \circledast(s_2) \to^\epsilon_{\circledast(\mathcal{P})} \circledast(t_2) \to^*_{\mathcal{R}} \cdots$$

and vice versa. This follows from the following three observations.

$s_i \to^{\epsilon}_{\mathcal{P}} t_i$ if and only if $\circledast(s_i) \to^{\epsilon}_{\circledast(\mathcal{P})} \circledast(t_i)$

We have $s_i \to^{\epsilon}_{\mathcal{P}} t_i$ if and only if $s_i = \ell\sigma$ and $t_i = r\sigma$ with $\ell \to r \in \mathcal{P}$. Since $\circledast(\mathcal{P})$ is well-defined and $\circledast$ is injective on terms, the latter is equivalent to

$$\circledast(s_i) = \circledast(\ell\sigma) = \circledast(\ell)\sigma \to^{\epsilon}_{\circledast(\mathcal{P})} \circledast(r)\sigma = \circledast(r\sigma) = \circledast(t_i)$$

$t_i \to^*_{\mathcal{R}} s_{i+1}$ if and only if $\circledast(t_i) \to^*_{\mathcal{R}} \circledast(s_{i+1})$

Since t_i and s_{i+1} have the same root symbol we can write $t_i = f(u_1, \ldots, u_n)$ and $s_{i+1} = f(u'_1, \ldots, u'_n)$. If $\circledast(f)$ is undefined or $n = 0$ then $\circledast(s_i) = s_i \to^*_{\mathcal{R}} t_i = \circledast(t_i)$. Suppose $\circledast(f) = k$. Since t_i is an instance of a right-hand side of a pair in $\mathcal{P}$ and $\circledast(\mathcal{P})$ is well-defined, u_k cannot be a variable. Write $u_k = g(v_1, \ldots, v_m)$. According to $\circledast$-stability, u_k is root stable and thus $u'_k = g(v'_1, \ldots, v'_m)$. Hence

$$\begin{aligned} t_i &= f(u_1, \ldots, u_{k-1}, g(v_1, \ldots, v_m), u_{k+1}, \ldots, u_n) \\ s_{i+1} &= f(u'_1, \ldots, u'_{k-1}, g(v'_1, \ldots, v'_m), u'_{k+1}, \ldots, u'_n) \end{aligned}$$

and

$$\begin{aligned} \circledast(t_i) &= \circledast_{fg}(u_1, \ldots, u_{k-1}, v_1, \ldots, v_m, u_{k+1}, \ldots, u_n) \\ \circledast(s_{i+1}) &= \circledast_{fg}(u'_1, \ldots, u'_{k-1}, v'_1, \ldots, v'_m, u'_{k+1}, \ldots, u'_n) \end{aligned}$$

Consequently, $t_i \to^*_{\mathcal{R}} s_{i+1}$ if and only if $u_j \to^*_{\mathcal{R}} u'_j$ for $1 \leqslant j \leqslant n$ with $j \neq k$ and $v_j \to^*_{\mathcal{R}} v'_j$ for $1 \leqslant j \leqslant m$ if and only if $\circledast(t_i) \to^*_{\mathcal{R}} \circledast(s_{i+1})$.

t_i terminates with respect to $\mathcal{R}$ if and only if $\circledast(t_i)$ terminates with respect to $\mathcal{R}$

This follows immediately from the observation above that all reductions in t_i take place in the arguments u_j or v_j. □

Corollary 3 *The DP processor $\mathcal{U}_2$ is sound and complete.*

Proof Immediate from Theorems 12 and 13 together with the fact that the composition of sound and complete DP processors yields a sound and complete DP processor. □

The next example shows that $\circledast$-stability is essential for soundness.

Example 20 Consider the non-terminating ATRS $\mathcal{R}$ consisting of the two rules

$$\mathsf{f\ a} \to \mathsf{g\ a} \qquad\qquad \mathsf{g} \to \mathsf{f}$$

which induces the infinite DP problem $(\mathcal{P}, \mathcal{R})$ with $\mathcal{P}$ consisting of the rules

$$\mathsf{f}^\sharp\ \mathsf{a} \to \mathsf{g}^\sharp\ \mathsf{a} \qquad\qquad \mathsf{f}^\sharp\ \mathsf{a} \to \mathsf{g}^\sharp$$

Since $\mathcal{P}{\downarrow_{\mathcal{U}}} = \mathcal{P}$ and $\mathcal{U}_1$ is sound, the DP problem $(\mathcal{P}, \mathcal{U}^+_\eta(\mathcal{R}))$ is also infinite. The set $\circledast(\mathcal{P}{\downarrow_{\mathcal{U}}})$ consists of the rules

$$\mathsf{f}^\sharp_1(\mathsf{a}) \to \mathsf{g}^\sharp_1(\mathsf{a}) \qquad\qquad \mathsf{f}^\sharp_1(\mathsf{a}) \to \mathsf{g}^\sharp$$

Clearly, the DP problem $(\circledast(\mathcal{P}), \mathcal{U}^+_\eta(\mathcal{R}))$ is finite. Note that $(\mathcal{P}, \mathcal{U}^+_\eta(\mathcal{R}))$ is not $\circledast$-stable as $\mathsf{g} \to^{\epsilon}_{\mathcal{U}^+_\eta(\mathcal{R})} \mathsf{f}$.

Since ⎈-stability is undecidable in general, for automation we need to approximate strong root stability. We present a simple criterion which is based on the term approximation TCAP from [13], where it was used to give a better approximation of dependency graphs.

Definition 13 ([13]) Let $\mathcal{R}$ be a TRS and t a term. The term $\mathsf{TCAP}_{\mathcal{R}}(t)$ is inductively defined as follows. If t is a variable, $\mathsf{TCAP}_{\mathcal{R}}(t)$ is a fresh variable. If $t = f(t_1, \ldots, t_n)$ then we let $u = f(\mathsf{TCAP}_{\mathcal{R}}(t_1), \ldots, \mathsf{TCAP}_{\mathcal{R}}(t_n))$ and define $\mathsf{TCAP}_{\mathcal{R}}(t)$ to be u if u does not unify with the left-hand side of a rule in $\mathcal{R}$, and a fresh variable otherwise.

Lemma 18 *A term t is strongly root stable for a TRS $\mathcal{R}$ if $\mathsf{TCAP}_{\mathcal{R}}(t) \notin \mathcal{V}$.*

Proof The only possibility for $\mathsf{TCAP}_{\mathcal{R}}(t) \notin \mathcal{V}$ is when $t = f(t_1, \ldots, t_n)$ and $u = f(\mathsf{TCAP}_{\mathcal{R}}(t_1), \ldots, \mathsf{TCAP}_{\mathcal{R}}(t_n))$ does not unify with a left-hand side of a rule in $\mathcal{R}$. Assume to the contrary that t is not strongly root stable. Then there are a substitution σ and a left-hand side ℓ of a rule in $\mathcal{R}$ such that $t\sigma \to_{\mathcal{R}}^* \ell\tau$. Write $\ell = f(l_1, \ldots, l_n)$. We have $t\sigma = f(t_1\sigma, \ldots, t_n\sigma)$ with $t_i\sigma \to_{\mathcal{R}}^* l_i\tau$ for $1 \leqslant i \leqslant n$. Hence $\mathsf{TCAP}_{\mathcal{R}}(t_i)\delta_i = l_i\tau$ for some substitution δ_i ([13, proof of Theorem 13]). Since the terms $\mathsf{TCAP}_{\mathcal{R}}(t_1), \ldots, \mathsf{TCAP}_{\mathcal{R}}(t_n)$ are linear and do not share variables, it follows that u unifies with ℓ, contradicting the assumption. □

Example 21 Consider the DP problem $(\mathcal{P}{\downarrow_{\mathcal{U}}}, \mathcal{U}^+_\eta(\mathcal{R}))$ of Example 19 with $\mathcal{P}{\downarrow_{\mathcal{U}}}$ consisting of the rule

$$\mathsf{a}_1(x) \star^\sharp \mathsf{a} \to \mathsf{a}_1(\mathsf{a}_1(\mathsf{a})) \star^\sharp x$$

and $\mathcal{U}^+_\eta(\mathcal{R})$ consisting of the rules

$$\mathsf{a}_2(x, \mathsf{a}) \to \mathsf{a}_2(\mathsf{a}_1(\mathsf{a}), x) \qquad \mathsf{a} \star x \to \mathsf{a}_1(x) \qquad \mathsf{a}_1(x) \star y \to \mathsf{a}_2(x, y)$$

Since $\mathsf{TCAP}_{\mathcal{U}^+_\eta(\mathcal{R})}(\mathsf{a}_1(\mathsf{a}_1(\mathsf{a}))) = \mathsf{a}_1(\mathsf{a}_1(\mathsf{a}))$ is not a variable, $\mathsf{a}_1(\mathsf{a}_1(\mathsf{a}))$ is strongly root stable. Hence $(\mathcal{P}{\downarrow_{\mathcal{U}}}, \mathcal{U}^+_\eta(\mathcal{R}))$ is ⎈-stable.

6.2 Innermost Termination

We start this section with a motivating example (which is related to Example 18).

Example 22 Consider the ATRS $\mathcal{R}$ consisting of the rule $x\ (\mathsf{a}\ \mathsf{a}) \to (\mathsf{a}\ \mathsf{a}\ \mathsf{a})\ x$. The only SCC in the dependency graph is $\mathcal{P} := \{x\ ^\sharp\ (\mathsf{a}\ \mathsf{a}) \to (\mathsf{a}\ \mathsf{a}\ \mathsf{a})\ ^\sharp\ x\}$. Since $\mathcal{P} \cup \mathcal{R}$ is not left head variable free the processor $\mathcal{U}_1$ cannot be applied. For proving innermost termination the usable rules processor [13] transforms $(\mathcal{P}, \mathcal{R})$ into $(\mathcal{P}, \varnothing)$ since the rule in $\mathcal{R}$ is not usable. Because $\mathcal{U}_1$ is sound for innermost termination (cf. Theorem 14), Example 18 finishes the innermost termination proof of $\mathcal{R}$.

In the following we deal with applicative DP problems $(\mathcal{P},\mathcal{R})$ for innermost termination. The following is the counterpart of Lemma 16 for innermost termination. Note that in contrast to Lemma 16, the result holds for arbitrary DP problems.

Lemma 19 *Let $(\mathcal{P},\mathcal{R})$ be a DP problem over the signature $\mathcal{G}$. If $(\mathcal{P},\mathcal{R})$ is innermost finite over $\mathcal{F}$ then $(\mathcal{P},\mathcal{R})$ is innermost finite over $\mathcal{G}$.*

Proof Let $\mathcal{V}' = \mathcal{V} \uplus \{x_t \mid t \in \mathcal{T}(\mathcal{G},\mathcal{V})\}$. Similar to the proof of Lemma 16 we define a mapping I from $\mathcal{T}(\mathcal{G},\mathcal{V})$ to $\mathcal{T}(\mathcal{F},\mathcal{V}')$ as follows:

$$I(t) = \begin{cases} t & \text{if } t \in \mathcal{V} \\ f(I(t_1),\ldots,I(t_n)) & \text{if } t = f(t_1,\ldots,t_n) \text{ and } f \in \mathcal{F} \\ x_t & \text{if } t = f(t_1,\ldots,t_n) \text{ and } f \in \mathcal{G} \setminus \mathcal{F} \end{cases}$$

Note that $I(t) = t$ for $t \in \mathcal{T}(\mathcal{F},\mathcal{V})$. First we show that $s \to^{\epsilon}_{\mathcal{P}} t$ implies $I(s) \to^{\epsilon}_{\mathcal{P}} I(t)$. From $s \to^{\epsilon}_{\mathcal{P}} t$ we get $s = \ell\sigma$ and $t = r\sigma$ for some $\ell \to r \in \mathcal{P}$. Since $\ell, r \in \mathcal{T}(\mathcal{F},\mathcal{V})$ we get $\ell = I(\ell)$ and $r = I(r)$ and consequently $I(s) = \ell I(\sigma) \to^{\epsilon}_{\mathcal{P}} rI(\sigma) = I(t)$. Since $t \to_{\mathcal{R}} s$ need not imply $I(t) \to_{\mathcal{R}} I(s)$ in general, we restrict ourselves to terms satisfying a special property. Let T be the set of all terms in $\mathcal{T}(\mathcal{G},\mathcal{V})$ whose $\mathcal{G}\setminus\mathcal{F}$-rooted subterms are normal forms. Note that if $C[\ell\sigma] \in T$ for some $\ell \to r \in \mathcal{R}$ then $I(C) = C$ since C cannot contain a symbol from $\mathcal{G}\setminus\mathcal{F}$. Hence $t \to_{\mathcal{R}} s$ with $t \in T$ implies $I(t) = C[\ell I(\sigma)] \to_{\mathcal{R}} C[rI(\sigma)] = I(s)$ and $I(s) \in T$ (since $\to_{\mathcal{R}}$ cannot introduce function symbols from $\mathcal{G} \setminus \mathcal{F}$). Together with the fact that $I(s) \in \mathit{NF}(\mathcal{R})$ whenever $s \in \mathit{NF}(\mathcal{R})$ this ensures that $I(t) \to^{\mathsf{i}!}_{\mathcal{R}} I(s)$ whenever $t \to^{\mathsf{i}!}_{\mathcal{R}} s$ and $t \in T$. Hence any presupposed sequence

$$s_1 \to^{\epsilon}_{\mathcal{P}} t_1 \to^{\mathsf{i}!}_{\mathcal{R}} s_2 \to^{\epsilon}_{\mathcal{P}} t_2 \to^{\mathsf{i}!}_{\mathcal{R}} \cdots$$

with $s_1 \in \mathit{NF}(\mathcal{R})$ (and hence $s_i, t_i \in T$ for all $i \geqslant 1$) using terms from $\mathcal{T}(\mathcal{G},\mathcal{V})$ is transformed into a sequence

$$I(s_1) \to^{\epsilon}_{\mathcal{P}} I(t_1) \to^{\mathsf{i}!}_{\mathcal{R}} I(s_2) \to^{\epsilon}_{\mathcal{P}} I(t_2) \to^{\mathsf{i}!}_{\mathcal{R}} \cdots$$

with $I(s_1) \in \mathit{NF}(\mathcal{R})$ using terms from $\mathcal{T}(\mathcal{F},\mathcal{V}')$. It follows that $(\mathcal{P},\mathcal{R})$ is innermost finite over $\mathcal{G}$. □

Theorem 14 *The DP processor $\mathcal{U}_1$ is sound for innermost termination.*

Proof Let $(\mathcal{P},\mathcal{R})$ be an ℓ-applicative DP problem. By the preceding lemma we may assume without loss of generality that the signature of $(\mathcal{P},\mathcal{R})$ is $\mathcal{F}$. Assume $(\mathcal{P},\mathcal{R})$ is not innermost finite. According to Krishna Rao [26] there exists an infinite sequence

$$s_1 \to^{\epsilon}_{\mathcal{P}} t_1 \to^{\mathsf{ri}!}_{\mathcal{R}} s_2 \to^{\epsilon}_{\mathcal{P}} t_2 \to^{\mathsf{ri}!}_{\mathcal{R}} \cdots$$

with $s_1 \in \mathit{NF}(\mathcal{R})$. We show that there is a sequence

$$s_1{\downarrow_{\mathcal{U}}} \to^{\epsilon}_{\mathcal{P}\downarrow_{\mathcal{U}}} t'_1 \to^{\mathsf{i}!}_{\mathcal{U}^+_\eta(\mathcal{R})} s_2{\downarrow_{\mathcal{U}}} \to^{\epsilon}_{\mathcal{P}\downarrow_{\mathcal{U}}} t'_2 \to^{\mathsf{i}!}_{\mathcal{U}^+_\eta(\mathcal{R})} \cdots$$

with terms t'_1, t'_2, ... such that $t_i \to^*_{\mathcal{U}} t'_i$ for $i \geqslant 1$. Fix i and let $\ell \to r$ be the rule from $\mathcal{P}$ that is used in $s_i \to^\epsilon_{\mathcal{P}} t_i$. So $t_i = r\sigma$ for some substitution σ. Lemma 17 yields $s_i{\downarrow_{\mathcal{U}}} \to^\epsilon_{\mathcal{P}{\downarrow_{\mathcal{U}}}} t'_i$ for the term $t'_i = r{\downarrow_{\mathcal{U}}}\sigma{\downarrow_{\mathcal{U}}}$. Clearly $t_i \to^*_{\mathcal{U}} t'_i$. Repeated application of Lemma 11 yields

$$t'_i \stackrel{\mathsf{i}}{\to}^*_{\mathcal{U}^+_\eta(\mathcal{R})} s'_{i+1} \stackrel{*}{\mathcal{U}}\!\leftarrow s_{i+1}$$

for some term s'_{i+1} that is a normal form of $\mathcal{R}_\eta{\downarrow_{\mathcal{U}}}$ due to Lemma 8 and the fact that s_{i+1} is a normal form of $\mathcal{R}$. It follows that $s'_{i+1} \stackrel{\mathsf{i}}{\to}^*_{\mathcal{U}^+_\eta(\mathcal{R})} s_{i+1}{\downarrow_{\mathcal{U}}}$ by repeated applications of Lemma 8 and innermost normalising s'_{i+1} with respect to $\mathcal{U}$. (Note that $\mathcal{U}$ is terminating and confluent.) Since $s_{i+1}{\downarrow_{\mathcal{U}}}$ is a normal form of $\mathcal{U}^+_\eta(\mathcal{R})$, we obtain $t'_i \stackrel{\mathsf{i}}{\to}^!_{\mathcal{U}^+_\eta(\mathcal{R})} s_{i+1}{\downarrow_{\mathcal{U}}}$. Since $s_1{\downarrow_{\mathcal{U}}} \in \mathit{NF}(\mathcal{U}^+_\eta(\mathcal{R}))$ whenever $s_1 \in \mathit{NF}(\mathcal{R})$ (Lemma 8) we conclude that the DP problem $(\mathcal{P}{\downarrow_{\mathcal{U}}}, \mathcal{U}^+_\eta(\mathcal{R}))$ is not innermost finite, as desired. □

Theorem 15 *The DP processor $\circledast$ is sound and complete for innermost termination.*

Proof Let $(\mathcal{P}, \mathcal{R})$ be $\circledast$-stable. Every infinite sequence

$$s_1 \to^\epsilon_{\mathcal{P}} t_1 \stackrel{\mathsf{i}}{\to}^!_{\mathcal{R}} s_2 \to^\epsilon_{\mathcal{P}} t_2 \stackrel{\mathsf{i}}{\to}^!_{\mathcal{R}} \cdots$$

can be transformed into the sequence

$$\circledast(s_1) \to^\epsilon_{\circledast(\mathcal{P})} \circledast(t_1) \stackrel{\mathsf{i}}{\to}^!_{\mathcal{R}} \circledast(s_2) \to^\epsilon_{\circledast(\mathcal{P})} \circledast(t_2) \stackrel{\mathsf{i}}{\to}^!_{\mathcal{R}} \cdots$$

and vice versa. This is obvious from the first observation in the proof of Theorem 13 and the following two facts: (1) $t_i \stackrel{\mathsf{i}}{\to}^*_{\mathcal{R}} s_{i+1}$ if and only if $\circledast(t_i) \stackrel{\mathsf{i}}{\to}^*_{\mathcal{R}} \circledast(s_{i+1})$ and (2) $s_i \in \mathit{NF}(\mathcal{R})$ if and only if $\circledast(s_i) \in \mathit{NF}(\mathcal{R})$ for all $i \geqslant 1$ (which follow from the proof of the second observation in the proof of Theorem 13). □

Corollary 4 *The DP processor $\mathcal{U}_2$ is sound for innermost termination.*

Proof Immediate from Theorems 14 and 15 together with the fact that the composition of sound DP processors yields a sound processor. □

The next example shows that $\mathcal{U}_1$ is not complete for innermost DP problems. (Note that Example 11 on page 15 does not provide a counterexample.)

Example 23 Consider the ATRS $\mathcal{R}$

$$\mathsf{f}\ x \to \mathsf{g}\ \mathsf{a}\ x \qquad \mathsf{g}\ x \to \mathsf{f} \qquad \mathsf{g} \to \mathsf{h}$$

which is innermost terminating because the rule $\mathsf{g}\ x \to \mathsf{f}$ cannot be used in an innermost sequence and without this rule $\mathcal{R}$ is easily seen to be terminating. Hence also the DP problem $(\mathcal{P}, \mathcal{R})$ with $\mathcal{P}$ consisting of

$$\mathsf{f}^\sharp\ x \to \mathsf{g}\ \mathsf{a}^\sharp\ x \qquad \mathsf{f}^\sharp\ x \to \mathsf{g}^\sharp\ \mathsf{a} \qquad \mathsf{g}^\sharp\ x \to \mathsf{f}$$

Table 1 Full termination for 195 ATRSs.

	direct			as processor						
	1	3	3+5	none	$\mathcal{A}$	$\mathcal{A}'$	$\mathcal{U}_1$	$\mathcal{U}_2$	3+5+$\mathcal{U}_2$	$\sum$
subterm criterion	1	47	48	41	–	–	41	58	61	61
matrix (1)	4	90	101	66	68	86	95	101	109	110
matrix (2)	7	108	131	108	111	128	133	134	138	138
matrix (3)	9	109	132	110	114	133	136	138	140	142

is innermost terminating. However, after applying the DP processor $\mathcal{U}_1$ there is an infinite innermost sequence:

$$\mathsf{f}^\sharp\, x \to^{\epsilon}_{\mathcal{P}\downarrow_{\mathcal{U}}} \mathsf{g}_1(\mathsf{a})^\sharp\, x \xrightarrow{\mathsf{i}}_{\mathcal{U}^+_\eta(\mathcal{R})} \mathsf{f}^\sharp\, x \to^{\epsilon}_{\mathcal{P}\downarrow_{\mathcal{U}}} \cdots$$

Note that $\mathsf{f}^\sharp\, x \in NF(\mathcal{U}^+_\eta(\mathcal{R}))$.

Because of the completeness of $\circledast$, $\mathcal{U}_2$ inherits incompleteness for innermost termination from $\mathcal{U}_1$.

7 Experiments

The transformations presented in this paper are implemented in the termination prover T$_\text{T}$T$_2$ [25]. For experimentation version 7.0.2 of the termination problem data base (TPDB)[3] has been considered which contains 195 ATRSs for full and 18 for innermost rewriting. All tests have been performed on a single core of a server equipped with eight dual-core AMD Opteron® processors 885 running at a clock rate of 2.6 GHz and 64 GB of main memory. Comprehensive details of the experiments[4] give evidence that the proposed transformations ease proving termination and upper bounds on the derivational complexity.

For proving (innermost) termination we considered two popular termination methods, namely the subterm criterion [17] and matrix interpretations [10] of dimensions one to three. For a matrix of dimension d the coefficients are represented by $5-d$ bits, one additional bit is allowed for intermediate results. Both methods are integrated within the dependency pair framework using dependency graph reasoning and usable rules as proposed in [13, 14, 16].

Table 1 differentiates between applying the transformations as a preprocessing step (direct) or within the dependency pair framework (as processor). For rows labeled "matrix", the numbers in parentheses refer to the dimension of the interpretations. The direct method of Corollary 1 (Theorem 3, Theorems 3 and 5) applies to 10 (141, 170) systems. If used directly, the numbers in the table refer to the systems that could be proved terminating in case of

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

[4] `http://cl-informatik.uibk.ac.at/software/ttt2/11jar/`

a successful transformation. Mirroring (when the original system is not left head variable free) does increase applicability of our (direct) transformation significantly.

The middle part of Table 1 states the number of successful termination proofs for transformation $\mathcal{A}$ ([13,40]) and the processors $\mathcal{U}_1$ (Definition 9) and $\mathcal{U}_2$ (Definition 12). Since transformation $\mathcal{A}$ does not preserve minimality (Example 24 in Section 8) one cannot use it together with the subterm criterion. In [40] it is shown that minimality is preserved when the transformation $\mathcal{A}$ is fused with the reduction pair and usable rules processors. Our implementation is based on the processor presented in [40, Theorem 6.17(D)]. In the column labeled $\mathcal{A}$ the transformation is fused with the reduction pair processor based on matrix interpretations while for column $\mathcal{A}'$ in addition the usable rules are computed based on TCAP. The first version is more suitable for a comparison with our processors (since $\mathcal{U}_1$ and $\mathcal{U}_2$ do also not incorporate usable rules) while the second version shows that transformation $\mathcal{A}$ can be combined with other termination criteria to obtain more advanced processors. Nevertheless the processors $\mathcal{U}_1$ and $\mathcal{U}_2$ admit more successful termination proofs. (In [40] further non-trivial extensions of the transformation $\mathcal{A}$ are considered.)

It is a trivial exercise to extend mirroring to DP problems. Our experiments revealed that (a) mirroring works better for the direct approach (hence we did not incorporate it into the middle block of the table) and (b) the uncurrying processors should be applied before other termination processors. Although Theorem 3 and the processor $\mathcal{U}_2$ are incomparable in power we recommend the usage of the processor. One reason is the increased strength and another one the modularity which allows to prevent pitfalls like Example 9. Last but not least, the processors $\mathcal{U}_1$ and $\mathcal{U}_2$ are not only sound but also complete (for full termination) which makes them suitable for non-termination analysis in principle. At least with T$_\mathsf{T}$T$_2$ we could not detect that this makes proving non-termination easier.

The right block of Table 1 gives the accumulated score for our transformations. As reference the total number of systems is given (labeled $\sum$) that any method in the corresponding row could prove terminating, showing that the cost for the auxiliary uncurrying rules is negligible compared to the gains in power. To see how the uncurrying transformation improves the power of a "full" termination prover we dropped it from the 2010 competition version of T$_\mathsf{T}$T$_2$. Then the number of successful termination proofs for applicative TRSs drops from 157 to 131.

Table 2 shows the results for innermost termination and admits similar conclusions. In contrast to the experiments reported in [44], for this table T$_\mathsf{T}$T$_2$ uses an approximation for the *innermost* dependency graph [13,16] and drops non-usable rules [13]. Due to the latter, the results for columns $\mathcal{A}$ and $\mathcal{A}'$ coincide.

Table 3 reports the performance of T$_\mathsf{T}$T$_2$ for derivational complexity. Since T$_\mathsf{T}$T$_2$ has no special methods for proving *innermost* derivational complexity, the numbers for dc and idc coincide. In this table columns labeled "–" do not use any preprocessing transformation whereas 1, 8/10 indicate ap-

Table 2 Innermost termination for 213 ATRSs.

	direct			as processor					
	1	7	7+5	none	$\mathcal{A}$	$\mathcal{U}_1$	$\mathcal{U}_2$	7+5+$\mathcal{U}_2$	$\sum$
subterm criterion	2	60	62	52	–	53	79	83	83
matrix (1)	4	101	112	76	97	114	120	126	126
matrix (2)	7	120	143	120	140	145	146	151	151
matrix (3)	9	121	144	122	144	149	150	153	154

Table 3 (Innermost) derivational complexity for 195 (213) ATRSs.

	TMI (1)			TMI (2)			TMI (3)			TMI (4)		
	–	1	8/10	–	1	8/10	–	1	8/10	–	1	8/10
dc/idc	3	3	4	10	10	14	12	14	26	12	16	28

plications of Corollary 1 and Theorems 8/10, respectively. We remark that Corollary 1 preserves derivational complexity. This is straightforward from [23, Lemma 2.1(3)]. Here TMIs of dimension one to four as presented in Theorem 1 are considered. Coefficients of TMIs are represented with $\max\{2, 5-d\}$ bits; again an additional bit is allowed for intermediate results. If Theorem 8 is used as preprocessing transformation, TMIs can, e.g., show 26 systems to have at most cubic derivational complexity while without uncurrying (with Theorem 1) the method only applies to 12 (14) systems. Especially for larger dimensions in Table 3 our transformation admits significant gains in power. We only tested the direct transformations, because proofs with dependency pairs give upper bounds on the derivational complexity much beyond exponential [32]. Since many of the ATRSs in this testbed contain partially applied terms or head variables in the right-hand sides this hampers applicability of Corollary 1.

8 Related Work

The transformation $\mathcal{A}$ of Giesl *et al.* [13] requires *proper* applicative DP problems, which are DP problems with the property that all occurrences of each constant have the same number of arguments. No uncurrying rules are added to the processed DP problems. This destroys minimality which means that not all DP processors (i.e., only those not relying on minimality) may be applied after transformation $\mathcal{A}$. The following example is from [40].

Example 24 Consider the ℓ-applicative DP problem $(\mathcal{P}, \mathcal{R})$ with $\mathcal{P}$ consisting of the rewrite rule $(\mathsf{g}\ x)\ (\mathsf{h}\ y)^\sharp\ z \to z\ z^\sharp\ z$ and $\mathcal{R}$ consisting of the rules

$$\mathsf{c}\ x\ y \to x \qquad \mathsf{c}\ (\mathsf{g}\ x)\ y \to \mathsf{c}\ (\mathsf{g}\ x)\ y$$
$$\mathsf{c}\ x\ y \to y \qquad \mathsf{c}\ x\ (\mathsf{g}\ y) \to \mathsf{c}\ x\ (\mathsf{g}\ y)$$

The DP problem $(\mathcal{P}, \mathcal{R})$ is not finite because of the following minimal rewrite sequence:

$$\begin{aligned}(\mathsf{g}\ x)\ (\mathsf{h}\ x)^\sharp\ (\mathsf{c}\ \mathsf{g}\ \mathsf{h}\ x) &\to^\epsilon_{\mathcal{P}} (\mathsf{c}\ \mathsf{g}\ \mathsf{h}\ x)\ (\mathsf{c}\ \mathsf{g}\ \mathsf{h}\ x)^\sharp\ (\mathsf{c}\ \mathsf{g}\ \mathsf{h}\ x)\\ &\to_{\mathcal{R}} (\mathsf{g}\ x)\ (\mathsf{c}\ \mathsf{g}\ \mathsf{h}\ x)^\sharp\ (\mathsf{c}\ \mathsf{g}\ \mathsf{h}\ x)\\ &\to_{\mathcal{R}} (\mathsf{g}\ x)\ (\mathsf{h}\ x)^\sharp\ (\mathsf{c}\ \mathsf{g}\ \mathsf{h}\ x)\end{aligned}$$

Applying the DP processor $\mathcal{U}_1$ produces $(\mathcal{P}{\downarrow_{\mathcal{U}}}, \mathcal{U}^+_\eta(\mathcal{R}))$ with $\mathcal{P}{\downarrow_{\mathcal{U}}}$ consisting of the rewrite rule $\mathsf{g}_1(x) \star \mathsf{h}_1(y) \star^\sharp z \to z \star z \star^\sharp z$ and $\mathcal{U}^+_\eta(\mathcal{R})$ consisting of the rules

$$\begin{array}{lll}\mathsf{c}_2(x,y) \to x & \mathsf{c}_2(\mathsf{g}_1(x),y) \to \mathsf{c}_2(\mathsf{g}_1(x),y) & \mathsf{g} \star x \to \mathsf{g}_1(x)\\ \mathsf{c}_2(x,y) \to y & \mathsf{c}_2(x,\mathsf{g}_1(y)) \to \mathsf{c}_2(x,\mathsf{g}_1(y)) & \mathsf{h} \star x \to \mathsf{h}_1(x)\\ \mathsf{c} \star x \to \mathsf{c}_1(x) & \mathsf{c}_1(x) \star y \to \mathsf{c}_2(x,y) & \end{array}$$

This DP problem is not finite:

$$\begin{aligned}\mathsf{g}_1(x) \star \mathsf{h}_1(x) \star^\sharp (\mathsf{c}_2(\mathsf{g},\mathsf{h}) \star x) &\to^\epsilon_{\mathcal{P}{\downarrow_{\mathcal{U}}}} (\mathsf{c}_2(\mathsf{g},\mathsf{h}) \star x) \star (\mathsf{c}_2(\mathsf{g},\mathsf{h}) \star x) \star^\sharp (\mathsf{c}_2(\mathsf{g},\mathsf{h}) \star x)\\ &\to^*_{\mathcal{U}^+_\eta(\mathcal{R})} (\mathsf{g} \star x) \star (\mathsf{h} \star x) \star^\sharp (\mathsf{c}_2(\mathsf{g},\mathsf{h}) \star x)\\ &\to^*_{\mathcal{U}^+_\eta(\mathcal{R})} \mathsf{g}_1(x) \star \mathsf{h}_1(x) \star^\sharp (\mathsf{c}_2(\mathsf{g},\mathsf{h}) \star x)\end{aligned}$$

Note that $\mathsf{c}_2(\mathsf{g},\mathsf{h}) \star x$ is terminating with respect to $\mathcal{U}^+_\eta(\mathcal{R})$.

The uncurrying rules are essential in this example, even though in the original DP problem all occurrences of each constant have the same number of arguments. Indeed, transformation $\mathcal{A}$ leaves out the uncurrying rules, resulting in a DP problem that admits infinite rewrite sequences but no minimal ones since one has to instantiate the variable z in $\mathsf{g}_1(x) \star \mathsf{h}_1(y) \star^\sharp z \to z \star z \star^\sharp z$ by a term that contains a subterm of the form $\mathsf{c}_2(\mathsf{g}_1(s), t)$ or $\mathsf{c}_2(s, \mathsf{g}_1(t))$ and the rules $\mathsf{c}_2(\mathsf{g}_1(x), y) \to \mathsf{c}_2(\mathsf{g}_1(x), y)$ and $\mathsf{c}_2(x, \mathsf{g}_1(y)) \to \mathsf{c}_2(x, \mathsf{g}_1(y))$ ensure that these terms are non-terminating.

Thiemann [40, Sections 6.2 and 6.3] addresses the loss of minimality by incorporating reduction pairs, usable rules, and argument filterings into the transformation $\mathcal{A}$. (The first two refinements were considered in the column labeled $\mathcal{A}'$ in Table 1.) In [40] it is further remarked that transformation $\mathcal{A}$ works better for innermost termination than for termination. This also holds for our processors $\mathcal{U}_1$ and $\mathcal{U}_2$ (cf. Section 7).

Recently Sternagel and Thiemann have generalised uncurrying to relative rewriting and non-applicative signatures and formalised it in the theorem prover Isabelle/HOL [38]. In particular their work comprises the certification of Theorem 3 (see [38, Corollary 11]) and the soundness direction of Theorem 12 (see [38, Theorem 15]) and Corollary 3 (see [38, Theorem 20]), respectively.

Aoto and Yamada [1,2] present transformation techniques for proving termination of simply typed ATRSs. After performing η-saturation, head variables are eliminated by instantiating them with 'template' terms of the appropriate type. In a final step, the resulting ATRS is translated into functional form.

Example 25 Consider again the ATRS $\mathcal{R}$ of Example 2. Suppose we adopt the following type declarations: $\mathsf{0} : \mathsf{int}$, $\mathsf{s} : \mathsf{int} \to \mathsf{int}$, $\mathsf{nil} : \mathsf{list}$, $(:) : \mathsf{int} \to \mathsf{list} \to \mathsf{list}$, $\mathsf{id} : \mathsf{int} \to \mathsf{int}$, $\mathsf{add} : \mathsf{int} \to \mathsf{int} \to \mathsf{int}$, and $\mathsf{map} : (\mathsf{int} \to \mathsf{int}) \to \mathsf{list} \to \mathsf{list}$. The head variable f in the right-hand side $: (f\ x)\ (\mathsf{map}\ f\ y)$ has type $\mathsf{int} \to \mathsf{int}$. There are three template terms of this type: s, id, and $\mathsf{add}\ z$. Instantiating f by these three terms in $\mathcal{R}_\eta$ produces the ATRS $\mathcal{R}'$:

$$\begin{array}{rr}
\mathsf{id}\ x \to x & \mathsf{map}\ f\ \mathsf{nil} \to \mathsf{nil} \\
\mathsf{add}\ \mathsf{0} \to \mathsf{id} & \mathsf{map}\ \mathsf{s}\ (:\ x\ y) \to :\ (\mathsf{s}\ x)\ (\mathsf{map}\ \mathsf{s}\ y) \\
\mathsf{add}\ \mathsf{0}\ y \to \mathsf{id}\ y & \mathsf{map}\ \mathsf{id}\ (:\ x\ y) \to :\ (\mathsf{id}\ x)\ (\mathsf{map}\ \mathsf{id}\ y) \\
\mathsf{add}\ (\mathsf{s}\ x)\ y \to \mathsf{s}\ (\mathsf{add}\ x\ y) & \mathsf{map}\ (\mathsf{add}\ z)\ (:\ x\ y) \to :\ (\mathsf{add}\ z\ x)\ (\mathsf{map}\ (\mathsf{add}\ z)\ y)
\end{array}$$

The TRS $\mathcal{R}'{\downarrow_{\mathcal{U}}}$ is terminating because its rules are oriented from left to right by the lexicographic path order. According to the main result of [2], the *simply typed* ATRS $\mathcal{R}$ is terminating, too.

The advantage of the simply typed approach is that no uncurrying rules are necessary because the application symbol has been eliminated from $\mathcal{R}'{\downarrow_{\mathcal{U}}}$. This typically results in simpler termination proofs. It is worthwhile to investigate whether a version of head variable instantiation can be developed for the untyped case. We would like to stress that with the simply typed approach one obtains termination only for those terms which are simply typed. Our approach, when it works, provides termination for all terms, irrespective of *any* typing discipline. In [3] the dependency pair method is adapted to deal with simply typed ATRSs. Again, head variable instantiation plays a key role.

Applicative term rewriting is not the only model for capturing higher-order aspects. The S-expression rewrite systems of Toyama [41] have a richer structure than applicative systems, which makes proving termination often easier. The notion of strong computability is often employed for proving termination of typed lambda calculi and variations like typed rewriting calculi [8]. Recent methods (e.g. [7, 21]) use types to exploit strong computability, leading to powerful termination methods which are directly applicable to higher-order systems. In [28] strong computability is used to analyse the termination of simply typed ATRSs with the dependency pair method, and recently this approach was extended to higher-order rewrite systems [27]. Finally, in [24] many concepts from the dependency pair framework have been lifted to algebraic functional systems, a higher-order concept based on simple types and explicit β-reduction.

While in this article we used termination techniques for ordinary TRSs to show termination of uncurried TRSs, it is worth noting that there is a specialised technique for uncurried TRSs. Van Bakel and Fernández [6] introduced the class of *curryfied* TRSs and its type system for normalisation. This class contains almost all uncurried TRSs. Notable exceptions are ℓ-ATRSs that contain a rule $\ell \to r$ with $\mathrm{aa}(\ell) > 0$. In [6] it is shown that a typable curryfied TRS is terminating if it satisfies the *general scheme* of [22].

We are not aware of other investigations dedicated to (derivational) complexity analysis of ATRSs.

Acknowledgements We thank the reviewers and Georg Moser for their helpful comments and René Thiemann for suggesting that the proof of [18, Theorem 33] contains a gap.

References

1. Aoto, T., Yamada, T.: Termination of simply typed term rewriting by translation and labelling. In: Nieuwenhuis, R. (ed.) Proc. 14th International Conference on Rewriting Techniques and Applications. LNCS, vol. 2706, pp. 380–394 (2003)
2. Aoto, T., Yamada, T.: Termination of simply-typed applicative term rewriting systems. In: Proc. 2nd International Workshop on Higher-Order Rewriting. Technical Report AIB-2004-03, RWTH Aachen. pp. 61–65 (2004)
3. Aoto, T., Yamada, T.: Dependency pairs for simply typed term rewriting. In: Giesl, J. (ed.) Proc. 16th International Conference on Rewriting Techniques and Applications. LNCS, vol. 3467, pp. 120–134 (2005)
4. Arts, T., Giesl, J.: Termination of term rewriting using dependency pairs. Theoretical Computer Science 236(1-2), 133–178 (2000)
5. Baader, F., Nipkow, T.: Term Rewriting and All That. Cambridge University Press (1998)
6. van Bakel, S., Fernández, M.: Normalization results for typeable rewrite systems. Information and Computation 133(2), 73–116 (1997)
7. Blanqui, F., Jouannaud, J.P., Rubio, A.: HORPO with computability closure: A reconstruction. In: Dershowitz, N., Voronkov, A. (eds.) Proc. 14th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. LNCS (LNAI), vol. 4790, pp. 138–150 (2007)
8. Cirstea, H., Kirchner, C.: The simply typed rewriting calculus. In: Proc. 3rd International Workshop on Rewriting Logic and its Applications. Electronic Notes in Theoretical Computer Science, vol. 36, pp. 24–42 (2000)
9. Dershowitz, N.: 33 Examples of termination. In: Comon, H., Jouannaud, J.P. (eds.) French Spring School of Theoretical Computer Science. LNCS, vol. 909, pp. 16–26 (1995)
10. Endrullis, J., Waldmann, J., Zantema, H.: Matrix interpretations for proving termination of rewrite systems. Journal of Automated Reasoning 40(2-3), 195–220 (2008)
11. Giesl, J., Raffelsieper, M., Schneider-Kamp, P., Swiderski, S.: Automated termination proofs for Haskell by term rewriting. ACM Transactions on Programming Languages and Systems 33(2), 39 pages (2011)
12. Giesl, J., Thiemann, R., Schneider-Kamp, P.: The dependency pair framework: Combining techniques for automated termination proofs. In: Baader, F., Voronkov, A. (eds.) Proc. 11th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. LNCS (LNAI), vol. 3452, pp. 301–331 (2004)
13. Giesl, J., Thiemann, R., Schneider-Kamp, P.: Proving and disproving termination of higher-order functions. In: Gramlich, B. (ed.) Proc. 5th International Workshop on Frontiers of Combining Systems. LNCS (LNAI), vol. 3717, pp. 216–231 (2005)
14. Giesl, J., Thiemann, R., Schneider-Kamp, P., Falke, S.: Mechanizing and improving dependency pairs. Journal of Automated Reasoning 37(3), 155–203 (2006)
15. Gramlich, B.: Relating innermost, weak, uniform and modular termination of term rewriting systems. In: Voronkov, A. (ed.) Proc. 3rd International Conference on Logic Programming and Automated Reasoning. LNCS (LNAI), vol. 624, pp. 285–296 (1992)
16. Hirokawa, N., Middeldorp, A.: Automating the dependency pair method. Information and Computation 199(1-2), 172–199 (2005)
17. Hirokawa, N., Middeldorp, A.: Tyrolean termination tool: Techniques and features. Information and Computation 205(4), 474–511 (2007)
18. Hirokawa, N., Middeldorp, A., Zankl, H.: Uncurrying for termination. In: Cervesato, I., Veith, H., Voronkov, A. (eds.) Proc. 15th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. LNCS, vol. 5330, pp. 667–681 (2008)
19. Hirokawa, N., Moser, G.: Automated complexity analysis based on the dependency pair method. In: Armando, A., Baumgartner, P., Dowek, G. (eds.) Proc. 4th International Joint Conference on Automated Reasoning. LNCS, vol. 5195, pp. 364–379 (2008)

20. Hofbauer, D., Lautemann, C.: Termination proofs and the length of derivations (preliminary version). In: Dershowitz, N. (ed.) Proc. 3rd International Conference on Rewriting Techniques and Applications. LNCS, vol. 355, pp. 167–177 (1989)
21. Jouannaud, J.P., Rubio, A.: Polymorphic higher-order recursive path orderings. Journal of the ACM 54(1) (2007). doi 10.1145/1206035.1206037
22. Jouannaud, J., Okada, M.: A computation model for executable higher-order algebraic specification languages. In: Proc. 6th International Annual IEEE Symposium on Logic in Computer Science. pp. 350–361 (1991). doi 10.1109/LICS.1991.151659
23. Kennaway, R., Klop, J.W., Sleep, M.R., de Vries, F.J.: Comparing curried and uncurried rewriting. Journal of Symbolic Computation 21(1), 15–39 (1996)
24. Kop, C., van Raamsdonk, R.: Higher order dependency pairs for algebraic functional systems. In: Schmidt-Schauß, M. (ed.) Proc. 22th International Conference on Rewriting Techniques and Applications. LIPIcs, vol. 10, pp. 203–218 (2011)
25. Korp, M., Sternagel, C., Zankl, H., Middeldorp, A.: Tyrolean Termination Tool 2. In: Treinen, R. (ed.) Proc. 20th International Conference on Rewriting Techniques and Applications. LNCS, vol. 5595, pp. 295–304 (2009)
26. Krishna Rao, M.R.K.: Some characteristics of strong innermost normalization. Theoretical Computer Science 239, 141–164 (2000)
27. Kusakari, K., Isogai, Y., Sakai, M., Blanqui, F.: Static dependency pair method based on strong computability for higher-order rewrite systems. IEICE Transactions 92-D(10), 2007–2015 (2009)
28. Kusakari, K., Sakai, M.: Enhancing dependency pair method using strong computability in simply-typed term rewriting. Applicable Algebra in Engineering, Communication and Computing 18(5), 407–431 (2007)
29. Middeldorp, A.: Modular Properties of Term Rewriting Systems. PhD thesis, Vrije Universiteit, Amsterdam (1990)
30. Middeldorp, A., Ohsaki, H., Zantema, H.: Transforming termination by self-labelling. In: McRobbie, M.A., Slaney, J.K. (eds.) Proc. 13th International Conference on Automated Deduction. LNCS (LNAI), vol. 1104, pp. 373–387 (1996)
31. Middeldorp, A., Moser, G., Neurauter, F., Waldmann, J., Zankl, H.: Spectral radius theory for automated complexity analysis of rewrite systems. In: Winkler, F. (ed.) Proc. 4th International Conference on Algebraic Informatics. LNCS, vol. 6742, pp. 1–20 (2011)
32. Moser, G., Schnabl, A.: The derivational complexity induced by the dependency pair method. In: Treinen, R. (ed.) Proc. 20th International Conference on Rewriting Techniques and Applications. LNCS, vol. 5595, pp. 255–267 (2009)
33. Moser, G., Schnabl, A., Waldmann, J.: Complexity analysis of term rewriting based on matrix and context dependent interpretations. In: Hariharan, R., Mukund, M., Vinay, V. (eds.) Proc. 28th Annual Conference on Foundations of Software Technology and Theoretical Computer Science. LIPIcs, vol. 2, pp. 304–315 (2008)
34. Neurauter, F., Zankl, H., Middeldorp, A.: Revisiting matrix interpretations for polynomial derivational complexity of term rewriting. In: Fermüller, C.G., Voronkov, A. (eds.) Proc. 17th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. LNCS (ARCoSS), vol. 6397, pp. 550–564 (2010)
35. Ohlebusch, E.: A simple proof of sufficient conditions for the termination of the disjoint union of term rewriting systems. Bulletin of the EATCS 50, 223–228 (1993)
36. van Oostrom, V.: Random descent. In: Baader, F. (ed.) Proc. 18th International Conference on Rewriting Techniques and Applications. LNCS, vol. 4533, pp. 314–328 (2007)
37. Sternagel, C., Thiemann, R.: Signature extensions preserve termination – an alternative proof via dependency pairs. In: Dawar, A., Veith, H. (eds.) Proc. 19th Annual Conference of the European Association for Computer Science Logic. LNCS, vol. 6247, pp. 514–528 (2010)
38. Sternagel, C., Thiemann, R.: Generalized and formalized uncurrying. In: Tinelli, C., Sofronie-Stokkermans, V. (eds.) Proc. 8th International Workshop on Frontiers of Combining Systems. LNCS, vol. 6989, pp. 243–258 (2011)
39. Terese: Term Rewriting Systems. vol. 55 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press (2003)

40. Thiemann, R.: The DP Framework for Proving Termination of Term Rewriting. PhD thesis, RWTH Aachen (2007). Available as Technical Report AIB-2007-17
41. Toyama, Y.: Termination of S-expression rewriting systems: Lexicographic path ordering for higher-order terms. In: van Oostrom, V. (ed.) Proc. 15th International Conference on Rewriting Techniques and Applications. LNCS, vol. 3091, pp. 40–54 (2004)
42. Waldmann, J.: Polynomially bounded matrix interpretations. In: Lynch, C. (ed.) Proc. 21st International Conference on Rewriting Techniques and Applications. LIPIcs, vol. 6, pp. 357–372 (2010)
43. Xi, H.: Towards automated termination proofs through "freezing". In: Nipkow, T. (ed.) Proc. 9th International Conference on Rewriting Techniques and Applications. LNCS, vol. 1379, pp. 271–285 (1998)
44. Zankl, H., Hirokawa, N., Middeldorp, A.: Uncurrying for innermost termination and derivational complexity. In: Proc. 5th International Workshop on Higher-Order Rewriting. Electronic Proceedings in Theoretical Computer Science, vol. 49, pp. 46–57 (2011)
45. Zankl, H., Korp, M.: Modular complexity analysis via relative complexity. In: Lynch, C. (ed.) Proc. 21st International Conference on Rewriting Techniques and Applications. LIPIcs, vol. 6, pp. 385–400 (2010)
46. Zankl, H., Middeldorp, A.: Satisfiability of non-linear (ir)rational arithmetic. In: Clarke, E.M., Voronkov, A. (eds.) Proc. 16th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning. LNCS (LNAI), vol. 6355, pp. 481–500 (2010)
47. Zantema, H.: Termination. In: Terese (ed.) Term Rewriting Systems. vol. 55 of Cambridge Tracts in Theoretical Computer Science. Cambridge University Press 181–259 (2003)