J Autom Reasoning (2008) 40:195–220
DOI 10.1007/s10817-007-9087-9

Matrix Interpretations for Proving Termination of Term Rewriting

Jörg Endrullis · Johannes Waldmann · Hans Zantema

Received: 10 October 2007 / Accepted: 12 October 2007 / Published online: 11 December 2007

Abstract We present a new method for automatically proving termination of term rewriting. It is based on the well-known idea of interpretation of terms where every rewrite step causes a decrease, but instead of the usual natural numbers we use vectors of natural numbers, ordered by a particular nontotal well-founded ordering. Function symbols are interpreted by linear mappings represented by matrices. This method allows us to prove termination and relative termination. A modification of the latter, in which strict steps are only allowed at the top, turns out to be helpful in combination with the dependency pair transformation. By bounding the dimension and the matrix coefficients, the search problem becomes finite. Our implementation transforms it to a Boolean satisfiability problem (SAT), to be solved by a state-of-the-art SAT solver.

Keywords Term rewriting · Termination · Matrix interpretations · Satisfiability

1 Introduction

In the past few years the emphasis in the research area of termination of term rewriting has been on proving termination automatically. Several tools have been

J. Endrullis
Department of Computer Science, Vrije Universiteit Amsterdam,
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
e-mail: joerg@few.vu.nl

J. Waldmann
Hochschule für Technik, Wirtschaft und Kultur (FH) Leipzig,
Fb IMN, PF 30 11 66, 04251 Leipzig, Germany
e-mail: waldmann@imn.htwk-leipzig.de

H. Zantema (✉)
Department of Computer Science, Technische Universiteit Eindhoven,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
e-mail: h.zantema@tue.nl

developed for this purpose. In the annual Termination Competition [3] these tools are compared. This competition has given a new drive to the quest for automated methods to obtain termination proofs for term rewriting.

The tools do apply established methods (path orderings, dependency pairs, interpretations, labelings) as well as new methods (RFC match bounds). Two insights are that general methods can be restricted to special cases, gaining efficiency without losing too much power, and that combining methods may lead to strong improvements. We present here one such phenomenon: termination proofs from interpretations into a well-founded monotone algebra. This is a well-known general theme, but our point is

- The special choice of the algebra and
- The special implementation of how to find suitable interpretations.

The carrier of the algebra consists of vectors of natural numbers on which we define a well-founded ordering that is not total. Each function symbol is interpreted by a suitable linear mapping. This method allows us to prove termination and relative termination. It has been proposed for string rewriting by Hofbauer and Waldmann [14]. In the present paper, we discuss its extension to term rewriting and a modification that allows to prove relative top-termination, that is, a variant of relative termination where the strict steps are allowed only on the top level. The latter is very helpful when using the dependency pair transformation. In order to cover the two-sorted nature of the dependency pair transformation, our monotone algebra setting is presented many-sorted.

We have implemented the method by bounding the dimension and the matrix coefficients, resulting in a search problem with a finite but typically huge search space. This is solved by transforming this finite search problem to a Boolean satisfiability problem (SAT) and using the state-of-the-art SAT solver Minisat, version 2 [5]. This performs surprisingly well on the Termination Problem Data Base [2]; see Section 8.

The main part of the paper is organized as follows. We present a many-sorted monotone algebra framework for relative termination and relative top-termination in Section 3, generalizing earlier results on monotone algebras. In Section 4 we choose the matrix instance of this framework. In Section 5 we combine this with the dependency pair method. Next, in Section 6 we compare our method with the matrix method for string rewriting. Our implementation is described in Section 7, and its performance is discussed in Section 8. In Section 9 we give some limitations of the approach and discuss bounds on reduction lengths.

Our methods are illustrated by examples. They are kept simple for the sake of presentation. Nevertheless, some of them cannot be proved to be terminating by any of the tools that participated in the Termination Competition 2006 [3] and do not use the techniques described in this paper.

A preliminary version of this paper appeared as [6].

2 Preliminaries

Let S be a nonempty set of sorts, and let Σ be an S-sorted signature, being a set of operation symbols each having a fixed arity in $S^* \times S$. An *S-sorted set A* is defined to consist of a set A_s for every $s \in S$. For an S-sorted set $\mathcal{X}$ of variable symbols let

$\mathcal{T}(\Sigma, \mathcal{X})$ be the S-sorted set of terms over Σ and $\mathcal{X}$, that is, the smallest S-sorted set satisfying

- $x_s \in \mathcal{T}(\Sigma, \mathcal{X})_s$ for all $x_s \in \mathcal{X}_s$, and
- If the arity of $f \in \Sigma$ is $((s_1, \ldots, s_n), s)$ and $t_i \in \mathcal{T}(\Sigma, \mathcal{X})_{s_i}$ for $i = 1, \ldots, n$, then $f(t_1, \ldots, t_n) \in \mathcal{T}(\Sigma, \mathcal{X})_s$.

A *term rewriting system* (TRS) R over $\Sigma, \mathcal{X}$ is an S-sorted set in which for every $s \in S$ the set R_s consists of pairs $(\ell, r) \in \mathcal{T}(\Sigma, \mathcal{X})_s \times \mathcal{T}(\Sigma, \mathcal{X})_s$, for which $\ell \notin \mathcal{X}_s$ and all variables in r occur in ℓ. Pairs (ℓ, r) are called *rewrite rules* of sort s and are usually written as $\ell \to r$.

An S-sorted relation $\to$ over an S-sorted set A is defined to be an S-sorted set for which $\to_s \subseteq A_s \times A_s$ for every $s \in S$.

A substitution $\sigma : \mathcal{X} \to \mathcal{T}(\Sigma, \mathcal{X})$ is defined by a map $\sigma_s : \mathcal{X}_s \to \mathcal{T}(\Sigma, \mathcal{X})_s$ for every $s \in S$. These extend to terms in the obvious way.

For a TRS R the (S-sorted) *top rewrite relation* $\overset{\text{top}}{\to}_R$ on $\mathcal{T}(\Sigma, \mathcal{X})$ is defined by $t \overset{\text{top}}{\to}_{R,s} u$ if and only if there is a rewrite rule $\ell \to r \in R_s$ and a substitution $\sigma : \mathcal{X} \to \mathcal{T}(\Sigma, \mathcal{X})$ such that $t = \ell\sigma$ and $u = r\sigma$. The (S-sorted) *rewrite relation* $\to_R$ is defined to be the smallest S-sorted relation satisfying

- If $t \overset{\text{top}}{\to}_R u$ then $t \to_R u$, and
- If $t_i \to_{R,s_i} u_i$ and $t_j = u_j$ for $j \neq i$, then $f(t_1, \ldots, t_n) \to_{R,s} f(u_1, \ldots, u_n)$ for every $f \in \Sigma$ of arity $((s_1, \ldots, s_n), s)$ and every $i = 1, \ldots, n$.

For S-sorted binary relations we write $\cdot$ for sortwise relation composition and * for sortwise transitive reflexive closure.

An S-sorted relation $\to$ is called *well-founded* or *terminating* if for no $s \in S$ an infinite sequence $t_1, t_2, t_3, \ldots$ exists such that $t_i \to_s t_{i+1}$ for all $i = 1, 2, 3, \ldots$.

A TRS R is called *terminating* if $\to_R$ is well-founded. Termination is also called *strong normalization*; therefore, the property of R being terminating is written as $\mathsf{SN}(R)$.

A binary relation $\to_1$ is called *terminating relative to* a binary relation $\to_2$, written as $\mathsf{SN}(\to_1 / \to_2)$, if for no $s \in S$ an infinite sequence $t_1, t_2, t_3, \ldots$ exists such that

- $t_i \to_{1,s} t_{i+1}$ for infinitely many values of i, and
- $t_i \to_{2,s} t_{i+1}$ for all other values of i.

We use the notation $\to_1 / \to_2$ to denote $\to_2^* \cdot \to_1 \cdot \to_2^*$; it is easy to see that $\mathsf{SN}(\to_1 / \to_2)$ coincides with well-foundedness of $\to_1 / \to_2$. We write $\mathsf{SN}(R/S)$ as a shorthand for $\mathsf{SN}(\to_R / \to_S)$, and we write $\mathsf{SN}(R_{\text{top}}/S)$ as a shorthand for $\mathsf{SN}\left(\overset{\text{top}}{\to}_R / \to_S\right)$.

For S consisting of one element all of these S-sorted notions coincide with the well-known corresponding notions in the one-sorted setting.

3 Monotone Algebras

A Σ-algebra $(A, [\cdot])$ is defined to consist of an S-sorted set A, and for every $f \in \Sigma$ a function $[f] : A_{s_1} \times \cdots \times A_{s_n} \to A_s$, where $((s_1, \ldots, s_n), s)$ is the arity of f. This function $[f]$ is called the *interpretation* of f.

Let $\alpha_s : \mathcal{X}_s \to A_s$ for every $s \in S$; this collection of maps α_s is written as $\alpha : \mathcal{X} \to A$. We define the term evaluation $[\cdot, \alpha] : \mathcal{T}(\Sigma, \mathcal{X}) \to A$ inductively by

$$[x, \alpha] = \alpha_s(x),$$
$$[f(t_1, \ldots, t_n), \alpha] = [f]([t_1, \alpha], \ldots, [t_n, \alpha])$$

for $f \in \Sigma$ and $x \in \mathcal{X}_s$.

Definition 1 An operation $[f] : A_{s_1} \times \cdots \times A_{s_n} \to A_s$ is *monotone* with respect to an S-sorted binary relation $\to$ on A if for all $a_i, b_i \in A_{s_i}$ for $i = 1, \ldots, n$ with $a_i \to_{s_i} b_i$ for some i and $a_j = b_j$ for all $j \neq i$ we have

$$[f](a_1, \ldots, a_n) \ \to_s \ [f](b_1, \ldots, b_n).$$

A *weakly monotone* Σ*-algebra* $(A, [\cdot], >, \gtrsim)$ is a Σ-algebra $(A, [\cdot])$ equipped with two S-sorted relations $>, \gtrsim$ on A such that

- $>$ is well-founded;
- $> \cdot \gtrsim \subseteq >$;
- For every $f \in \Sigma$ the operation $[f]$ is monotone with respect to $\gtrsim$.

An *extended monotone* Σ*-algebra* $(A, [\cdot], >, \gtrsim)$ is a weakly monotone Σ-algebra $(A, [\cdot], >, \gtrsim)$ in which, moreover, for every $f \in \Sigma$ the operation $[f]$ is monotone with respect to $>$.

The combination $>, \gtrsim$ is closely related to the notion of *reduction pair* in the dependency pair framework, for example, in [9]. A crucial difference is that the relations in a reduction pair are relations on terms that are closed under substitutions, while in our setting they are relations on the arbitrary (many-sorted) set A.

In the sequel we often omit sort information, for example, writing $[t, \alpha] > [u, \alpha]$ rather than $[t, \alpha] >_s [u, \alpha]$. A TRS given without sort information is assumed to be one-sorted; that is, S consists of one element.

The one-sorted version of extended monotone algebra where $\gtrsim$ is left implicit by defining it as the union of $>$ and equality is called *well-founded monotone algebra* in [18, 19]. A main theorem states that a TRS is terminating if and only if there is a well-founded monotone algebra $(A, [\cdot], >)$ such that $[\ell, \alpha] > [r, \alpha]$ for every rule $\ell \to r$ and every $\alpha : \mathcal{X} \to A$. First we show that for relative termination we have a similar characterization based on extended monotone algebras, but not on this earlier version of well-founded monotone algebras.

Theorem 2 *Let R, S be TRSs over a signature Σ. Then*

1. $\mathsf{SN}(R/S)$ *if and only if there exists an extended monotone Σ-algebra $(A, [\cdot], >, \gtrsim)$ such that $[\ell, \alpha] > [r, \alpha]$ for every rule $\ell \to r$ in R and $[\ell, \alpha] \gtrsim [r, \alpha]$ for every rule $\ell \to r$ in S, for every $\alpha : \mathcal{X} \to A$.*
2. $\mathsf{SN}(R_{\mathrm{top}}/S)$ *if and only if there exists a weakly monotone Σ-algebra $(A, [\cdot], >, \gtrsim)$ such that $[\ell, \alpha] > [r, \alpha]$ for every rule $\ell \to r$ in R and $[\ell, \alpha] \gtrsim [r, \alpha]$ for every rule $\ell \to r$ in S, for every $\alpha : \mathcal{X} \to A$.*

Proof For the "if" part of part 1 assume such an extended monotone algebra $(A, [\cdot], >, \gtrsim)$ exists; we have to prove $\mathsf{SN}(R/S)$. So, assume an infinite reduction

$$t_1 \to_{R \cup S} t_2 \to_{R \cup S} t_3 \to_{R \cup S} \cdots$$

containing infinitely many R-steps. Choose $\alpha : \mathcal{X} \to A$ arbitrary. Because of monotonicity with respect to $>$ we obtain $[t_i, \alpha] > [t_{i+1}, \alpha]$ if $t_i \to_R t_{i+1}$, and because of monotonicity with respect to $\gtrsim$ we obtain $[t_i, \alpha] \gtrsim [t_{i+1}, \alpha]$ if $t_i \to_S t_{i+1}$. Since $> \cdot \gtrsim \subseteq >$ we obtain $> \cdot \gtrsim^* \subseteq >$; hence for $t_i \to_R t_{i+1} \to_S^* t_j$ we obtain $[t_i, \alpha] > [t_j, \alpha]$. Since there are infinitely many R-steps, this gives rise to an infinite decreasing sequence with respect to $>$, contradicting well-foundedness.

The proof of the "if" part of part 2 is similar; now all $\to_R$-steps in the assumed infinite reduction are $\overset{\text{top}}{\to}_R$-steps, by which monotonicity with respect to $>$ is not required.

For the "only if" part, assume $\mathsf{SN}(R/S)$ (respectively, $\mathsf{SN}(R_{\text{top}}/S)$) holds. Choose $A = \mathcal{T}(\Sigma, \mathcal{X})$, and $[f](t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$ for all $f \in \Sigma$. Define $> = (\to_R / \to_S)^+$ and $\gtrsim = (\to_{R \cup S})^*$ $\left(\text{respectively, } > = \left(\overset{\text{top}}{\to}_R / \to_S\right)^+\right)$ and $\gtrsim = \to_S^*$. Then $(A, [\cdot], >, \gtrsim)$ satisfies all requirements; where well-foundedness of $>$ is concluded from the assumption $\mathsf{SN}(R/S)$ (respectively, $\mathsf{SN}(R_{\text{top}}/S)$). □

For the relations $>, \gtrsim$ we typically have in mind some more properties, such as transitivity of both $>$ and $\gtrsim$, reflexivity of $\gtrsim$, and $\gtrsim \cdot > \cdot \gtrsim \subseteq > \subseteq \gtrsim$. From the proof of Theorem 2, however, we see that these properties are not essential.

For this characterization of relative termination the general notion of extended monotone algebra is essential: it does not hold for the restricted case where $\gtrsim$ coincides with the union of $>$ and equality, as is shown by the following example.

Example 1 Let R consist of the rule $f(f(x)) \to f(g(f(x)))$, and let S consist of the rule $f(x) \to g(f(x))$. Define $(A, [\cdot], >, \gtrsim)$ by

- $A = \mathbf{N} \times \mathbf{N}$,
- $[f](m, n) = (m + n, 1)$ for $m, n \in \mathbf{N}$,
- $[g](m, n) = (m, 0)$ for $m, n \in \mathbf{N}$,
- $(m, n) > (m', n') \iff m > m' \wedge n \geq n'$,
- $(m, n) \gtrsim (m', n') \iff m \geq m' \wedge n \geq n'$.

It is easily checked that $(A, [\cdot], >, \gtrsim)$ is an extended monotone algebra. Moreover, if $\alpha(x) = (m, n)$ we obtain

$$[f(f(x)), \alpha] = (m + n + 1, 1) > (m + n, 1) = [f(g(f(x))), \alpha]$$

and

$$[f(x), \alpha] = (m + n, 1) \gtrsim (m + n, 0) = [g(f(x)), \alpha],$$

proving $\mathsf{SN}(R/S)$ by Theorem 2. The reader may observe the matrix/vector flavor of this proof; indeed this proof coincides with a proof found by the matrix method, as will be described in this paper.

Assume that for R, S an alternative extended monotone algebra exists in which $\gtrsim$ coincides with the union of $>$ and equality and the properties of Theorem 2 hold. Fix α arbitrarily. Then $[f(x), \alpha] \gtrsim [g(f(x)), \alpha]$, so either $[f(x), \alpha] = [g(f(x)), \alpha]$ or

$[f(x),\alpha] > [g(f(x)),\alpha]$. The first case contradicts $[f(f(x)),\alpha]) > [f(g(f(x))),\alpha]$; hence we have $[f(x),\alpha] > [g(f(x)),\alpha]$. But then by monotonicity of $[g]$ with respect to $>$ we obtain

$$[f(x),\alpha] > [g(f(x)),\alpha] > [g(g(f(x))),\alpha] > [g(g(g(f(x)))),\alpha] > \cdots,$$

contradicting well-foundedness. Hence for this example it is essential that $\gtrsim$ differs from the union of $>$ and equality.

Now we arrive at the general theorem for extended monotone algebras as we will use it for proving (relative) termination by matrix interpretations.

Theorem 3 *Let R, S be TRSs over a signature Σ.*

1. *Let $(A, [\cdot], >, \gtrsim)$ be an extended monotone Σ-algebra such that $[\ell,\alpha] \gtrsim [r,\alpha]$ for every rule $\ell \to r$ in $R \cup S$ and every $\alpha : \mathcal{X} \to A$. Let R' consist of all rules $\ell \to r$ from $R \cup S$ satisfying $[\ell,\alpha] > [r,\alpha]$ for every $\alpha : \mathcal{X} \to A$.*
   Then $\mathsf{SN}((R \setminus R')/(S \setminus R'))$ implies $\mathsf{SN}(R/S)$.
2. *Let $(A, [\cdot], >, \gtrsim)$ be a weakly monotone Σ-algebra such that $[\ell,\alpha] \gtrsim [r,\alpha]$ for every rule $\ell \to r$ in $R \cup S$ and every $\alpha : \mathcal{X} \to A$. Let R' consist of all rules $\ell \to r$ from R satisfying $[\ell,\alpha] > [r,\alpha]$ for every $\alpha : \mathcal{X} \to A$.*
   Then $\mathsf{SN}((R \setminus R')_{\mathrm{top}}/S)$ implies $\mathsf{SN}(R_{\mathrm{top}}/S)$.

Proof For part 1 assume $\mathsf{SN}((R \setminus R')/(S \setminus R'))$. Take any infinite reduction with respect to $R \cup S$. From Theorem 2 part 1 we conclude $\mathsf{SN}(R'/(R \cup S))$, so this infinite reduction contains only finitely many R'-steps. So after removing a finite initial part, this reduction consists of only $(R \cup S) \setminus R'$-steps. Since $\mathsf{SN}((R \setminus R')/(S \setminus R'))$, this remaining part contains only finitely many $R \setminus R'$-steps. So the original infinite reduction contains only finitely many R-steps. Hence we proved $\mathsf{SN}(R/S)$.

For part 2 assume $\mathsf{SN}((R \setminus R')_{\mathrm{top}}/S)$. Take any infinite reduction with respect to $\overset{\mathrm{top}}{\to}_R \cup \to_S$. From Theorem 2 part 2 we conclude $\mathsf{SN}(R'_{\mathrm{top}}/(R \cup S))$, so this infinite reduction contains only finitely many $\overset{\mathrm{top}}{\to}_{R'}$-steps. So after removing a finite initial part, this reduction consists of only $\overset{\mathrm{top}}{\to}_{R \setminus R'}$-steps and $\to_S$-steps. Since $\mathsf{SN}((R \setminus R')_{\mathrm{top}}/S)$, this remaining part contains only finitely many $\overset{\mathrm{top}}{\to}_{R \setminus R'}$-steps. So the original infinite reduction contains only finitely many $\overset{\mathrm{top}}{\to}_R$-steps, proving $\mathsf{SN}(R_{\mathrm{top}}/S)$. □

The basic way to apply Theorem 3 is as follows. If $\mathsf{SN}(R/S)$ (or $\mathsf{SN}(R_{\mathrm{top}}/S)$) has to be proved, then try to find an extended (or weakly) monotone Σ-algebra satisfying the conditions for which R' is not empty. Then the proof obligation is weakened to $\mathsf{SN}((R \setminus R')/(S \setminus R'))$ (or $\mathsf{SN}((R \setminus R')_{\mathrm{top}}/S)$). For this we again apply Theorem 3 in the same way. This is repeated until $R \setminus R' = \emptyset$, for which the remaining proof obligation $\mathsf{SN}((R \setminus R')/(S \setminus R'))$ (or $\mathsf{SN}((R \setminus R')_{\mathrm{top}}/S)$) trivially holds. Proving termination rather than relative termination is a special case of this approach: then S is empty in $\mathsf{SN}(R/S)$.

If this approach finishes in one step, that is, $R \setminus R' = \emptyset$, then also Theorem 2 could have been applied with the same result, as we did for Example 1.

In Example 4 we will show that in the setting of Theorem 2 part 2 it is not allowed to remove rules from S: $\mathsf{SN}((R \setminus R')_{\text{top}}/(S \setminus S'))$ does not imply $\mathsf{SN}(R_{\text{top}}/S)$, where S' consists of the rules $\ell \to r$ from S satisfying $[\ell, \alpha] > [r, \alpha]$ for every $\alpha : \mathcal{X} \to A$.

Application of Theorem 3 is well known for the case where A consists of the natural numbers, or natural numbers ≥ 2, all functions $[f]$ are polynomials, and $>$ and $\gtrsim$ have their usual meaning. For part 1 strict monotonicity is required, while for part 2 weak monotonicity is sufficient. In this polynomial case $\gtrsim$ coincides with the union of $>$ and equality. In the matrix interpretations in the vector algebras considered in this paper, this is not the case for dimensions > 1.

4 Matrix Interpretations

In this paper we focus on interpretations based on matrices. For the basic version this means that we fix a dimension d and construct a one-sorted extended monotone algebra $(A, [\cdot], >, \gtrsim)$ in which $A = \mathbf{N}^d$. Without any complication this extends to the many-sorted setting in which every sort has its own dimension. To keep the presentation simple, here we restrict to the one-sorted case.

The relations $>$ and $\gtrsim$ on A are defined as follows:

$$(v_1, \ldots, v_d) > (u_1, \ldots, u_d) \iff v_1 > u_1 \wedge v_i \geq u_i \text{ for } i = 2, 3, \ldots, d,$$

$$(v_1, \ldots, v_d) \gtrsim (u_1, \ldots, u_d) \iff v_i \geq u_i \text{ for } i = 1, 2, \ldots, d.$$

All requirements for $>$ and $\gtrsim$ from Definition 1 trivially hold. Note that $\gtrsim$ does not coincide with the union of $>$ and equality. Of course other orders on vectors could have been chosen, too, but many of them are not suitable for our purpose. For instance, choosing a lexicographic order fails because then multiplication by a constant matrix is not monotone in general.

For the interpretation $[c]$ of a symbol $c \in \Sigma$ of arity 0 we choose any element of A. For the interpretation $[f]$ of a symbol $f \in \Sigma$ of arity $n \geq 1$ we choose n matrices $F_1, F_2, \ldots, F_n$ over $\mathbf{N}$, each of size $d \times d$, such that the upper left elements $(F_i)_{1,1}$ are positive for all $i = 1, 2, \ldots, n$, and a vector $\vec{f} \in \mathbf{N}^d$. Now we define

$$[f](\vec{v}_1, \ldots, \vec{v}_n) = F_1\vec{v}_1 + \cdots + F_n\vec{v}_n + \vec{f}$$

for all $\vec{v}_1, \ldots, \vec{v}_n \in A$. One easily checks that $[f]$ is monotone with respect to $\gtrsim$. Because of positiveness of the upper left matrix elements we also conclude that $[f]$ is monotone with respect to $>$. So, by choosing all $[f]$ of this shape, all requirements of an extended monotone algebra are fulfilled.

To apply Theorem 3, part 1, we should be able to check whether $[\ell, \alpha] \gtrsim [r, \alpha]$ or $[\ell, \alpha] > [r, \alpha]$ for all $\alpha : \mathcal{X} \to A$, for given rewrite rules $\ell \to r$. Let $x_1, \ldots, x_k$ be the variables occurring in ℓ, r. Then, because of the linear shape of the functions $[f]$, we can compute matrices $L_1, \ldots, L_k, R_1, \ldots, R_k$ and vectors $\vec{l}, \vec{r}$ such that

$$[\ell, \alpha] = L_1\vec{x}_1 + \cdots + L_k\vec{x}_k + \vec{l}$$

and

$$[r, \alpha] = R_1\vec{x}_1 + \cdots + R_k\vec{x}_k + \vec{r},$$

where $\alpha(x_i) = \vec{x}_i$ for $i = 1, \ldots, k$.

For matrices $B, C \in \mathbf{N}^{d\times d}$ write

$$B \gtrsim C \iff \forall i, j : (B)_{i,j} \geq (C)_{i,j}.$$

The following lemma states how the conditions of Theorem 3 can be checked.

Lemma 4 *Let $L_1, \ldots, L_k$, $R_1, \ldots, R_k$, and $\vec{l}, \vec{r}$ correspond to a rewrite rule $\ell \to r$ as described above. Then*

1. $[\ell, \alpha] \gtrsim [r, \alpha]$ *for every $\alpha : \mathcal{X} \to A$ if and only if*

$$L_i \gtrsim R_i \text{ for } i = 1, \ldots, k, \text{ and } \vec{l} \gtrsim \vec{r},$$

2. $[\ell, \alpha] > [r, \alpha]$ *for every $\alpha : \mathcal{X} \to A$ if and only if*

$$L_i \gtrsim R_i \text{ for } i = 1, \ldots, k, \text{ and } \vec{l} \gtrsim \vec{r}, \text{ and } l_1 > r_1.$$

Proof By definition $[\ell, \alpha] \gtrsim [r, \alpha]$ holds for every $\alpha : \mathcal{X} \to A$ if and only if for all $\vec{x}_1, \ldots, \vec{x}_k \in \mathbf{N}^d$ the vector

$$\sum_{i=1}^{k} (L_i - R_i)\vec{x}_i + (\vec{l} - \vec{r})$$

consists of non-negative numbers. If $L_i \gtrsim R_i$ for $i = 1, \ldots, k$, and $\vec{l} \gtrsim \vec{r}$, then this property holds because all entries of $L_i - R_i$ and $\vec{l} - \vec{r}$ are non-negative.

Conversely, assume that this property holds. Then $\vec{l} \gtrsim \vec{r}$ holds by choosing all $\vec{x}_i$ to be zero. Assume some entry of $L_i - R_i$ is strictly negative. Then choosing $\vec{x}_j$ to be zero except for $j = i$ and choosing all entries of $\vec{x}_i$ to be zero except for one chosen to be large, we obtain a negative entry in $\sum_{i=1}^{k}(L_i - R_i)\vec{x}_i + (\vec{l} - \vec{r})$, contradiction. Hence all entries of $L_i - R_i$ are non-negative.

This proves the first item of the lemma; the second is similar, with the only difference of strict inequality in the first argument. □

Now the approach of applying Theorem 3, part 1, for proving $\mathsf{SN}(R/S)$ is as follows:

- Fix a dimension d.
- For every symbol $f \in \Sigma$ choose matrices $F_i \in \mathbf{N}^{d\times d}$ for $i = 1, 2, \ldots, n$ for n being the arity of f, such that the upper left elements $(F_i)_{1,1}$ are positive for all $i = 1, 2, \ldots, n$, and a vector $\vec{f} \in \mathbf{N}^d$.
- For every rule $\ell \to r \in R \cup S$ we check whether $L_i \gtrsim R_i$ for $i = 1, \ldots, k$ and $\vec{l} \gtrsim \vec{r}$ for the corresponding matrices L_i, R_i and vectors $\vec{l}, \vec{r}$ as defined above.
- If this does not hold, the method fails.
- If this holds, then we remove all rules from R and S moreover satisfying $l_1 > r_1$.
- If the remaining R is empty, we are finished; otherwise we repeat the process for the reduced TRSs R, S or apply any other technique for proving (relative) termination.

In the sequel, we refer to this approach as the *direct method*.

Note that for our matrix interpretations after choosing the interpretation checking whether a left-hand side is greater than (or greater than or equal to) a right-hand side is decidable from Lemma 4, in contrast to nonlinear polynomial interpretations. For dimension $d = 1$ our matrix interpretations coincide with linear polynomial interpretations.

In the implementation we transform the whole scheme into a SAT problem rather than choosing F_i and $\vec{f}$ several times and checking the conditions for L_i, R_i, $\vec{l}$, $\vec{r}$ separately for every choice.

Example 1 can be seen as an instance of this approach, corresponding to the choices

$$[f](\vec{x}) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad [g](\vec{x}) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Example 2 Consider the TRS consisting of the following rules.

$$h(g(s(x), y), g(z, u)) \to h(g(u, s(z)), g(s(y), x))$$
$$s(s(x)) \to s(x)$$

We choose $A = \mathbf{N}^2$ together with the symbol interpretations.

$$[h](\vec{x}, \vec{y}) = \begin{pmatrix} 3 & 0 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 1 & 4 \\ 0 & 0 \end{pmatrix} \cdot \vec{y}$$

$$[g](\vec{x}, \vec{y}) = \begin{pmatrix} 3 & 3 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 1 & 1 \\ 2 & 2 \end{pmatrix} \cdot \vec{y}$$

$$[s](\vec{x}) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Let $\alpha : \mathcal{X} \to A$ be arbitrary; write $\alpha(x) = \vec{x}$, $\alpha(y) = \vec{y}$, $\alpha(z) = \vec{z}$ and $\alpha(u) = \vec{u}$. Then we obtain

$$\begin{aligned}
&[h(g(s(x), y), g(z, u)), \alpha] \\
&\quad = \begin{pmatrix} 9 & 9 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 3 & 3 \\ 0 & 0 \end{pmatrix} \cdot \vec{y} + \begin{pmatrix} 3 & 3 \\ 0 & 0 \end{pmatrix} \cdot \vec{z} + \begin{pmatrix} 9 & 9 \\ 0 & 0 \end{pmatrix} \cdot \vec{u} + \begin{pmatrix} 9 \\ 0 \end{pmatrix} \\
&\quad > \begin{pmatrix} 9 & 9 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 3 & 3 \\ 0 & 0 \end{pmatrix} \cdot \vec{y} + \begin{pmatrix} 3 & 3 \\ 0 & 0 \end{pmatrix} \cdot \vec{z} + \begin{pmatrix} 9 & 9 \\ 0 & 0 \end{pmatrix} \cdot \vec{u} + \begin{pmatrix} 6 \\ 0 \end{pmatrix} \\
&\quad = [h(g(u, s(z)), g(s(y), x)), \alpha]
\end{aligned}$$

and

$$[s(s(x)), \alpha] = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} > \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} = [s(x), \alpha].$$

By Theorem 3 we conclude that the system is terminating.

As indicated by Lemma 4, in general we conclude $[\ell, \alpha] > [r, \alpha]$ for arbitrary $\alpha : \mathcal{X} \to A$ if we have a strict decrease in the first vector coefficient and $\geq$ for all matrix coefficients and all other vector coefficients.

We conclude this section by an example of relative termination.

Example 3 Define R, S as follows; we want to prove $\mathsf{SN}(R/S)$.

$$R = \{f(a, g(y), z) \to f(a, y, g(y)),\ f(b, g(y), z) \to f(a, y, z),\ a \to b\}$$

$$S = \{f(x, y, z) \to f(x, y, g(z))\}.$$

We choose the following symbol interpretations:

$$[a] = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad [b] = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$[f](\vec{x}, \vec{y}, \vec{z}) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \cdot \vec{y} + \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \cdot \vec{z} + \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$[g](\vec{x}) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Thereby all rules in $R \cup S$ are weakly decreasing, i.e. all matrix coefficients in the left hand side are greater or equal to the corresponding coefficients in the right hand side. Moreover, all upper left matrix coefficients are nonzero and the rules in R are strictly decreasing in the first coefficient. Hence by Theorem 3 all rules from R may be removed, proving $\mathsf{SN}(R/S)$.

5 Top Reduction and Dependency Pairs

For a one-sorted TRS R a symbol $f \in \Sigma$ is called a *defined symbol* if f is the root symbol of a left hand side of a rule of R. For every defined symbol $f \in \Sigma$ a new marked symbol $f_\#$ is added having the same arity as f. If $f(s_1, \ldots, s_n) \to C[g(t_1, \ldots, t_m)]$ is a rule in R and g is a defined symbol of R, then the rewrite rule $f_\#(s_1, \ldots, s_n) \to g_\#(t_1, \ldots, t_m)$ is called a *dependency pair* of R. The TRS consisting of all dependency pairs of R is denoted by $\mathsf{DP}(R)$. We consider these TRSs R and $\mathsf{DP}(R)$ to be S-sorted for $S = \{s, \#\}$, and every $f \in \Sigma$ has arity $((s, \ldots, s), s)$ and its marked version $f_\#$ has arity $((s, \ldots, s), \#)$. So all rules in R are of sort s and all rules of $\mathsf{DP}(R)$ are of sort #.

The main theorem about dependency pairs is the following, due to Arts and Giesl [1].

Theorem 5 *Let R be a one-sorted TRS. Then* $\mathsf{SN}(R)$ *if and only if* $\mathsf{SN}(\mathsf{DP}(R)_{\mathrm{top}}/R)$.

We will use this theorem for proving $\mathsf{SN}(R)$ by proving $\mathsf{SN}(\mathsf{DP}(R)_{\mathrm{top}}/R)$ using part 2 of Theorem 3. Many improvements of Theorem 5 are known, such as dependency graph approximation and the usable rules criterion [8, 9], the subterm criterion [9], and restriction to strongly connected components [1, 10]. By all of these improvements $\mathsf{SN}(R)$ is proved by proving one or more instances of the shape $\mathsf{SN}(D_{\mathrm{top}}/R')$, where $D \subseteq \mathsf{DP}(R)$ and $R' \subseteq R$. Our approach using part 2 of Theorem 3 applies on these instances as well. For the presentation here we focus on the basic version $\mathsf{SN}(\mathsf{DP}(R)_{\mathrm{top}}/R)$.

For doing so by matrix interpretations we fix a dimension d as before and construct a weakly monotone algebra $(A, [\cdot], >, \gtrsim)$ in which $A_s = \mathbf{N}^d$ and $A_\# = \mathbf{N}$. The reason for choosing $A_\# = \mathbf{N}$ rather than $A_\# = \mathbf{N}^d$ is that this yields simpler interpretations without losing power, as we will see.

The relation $\gtrsim$ on $A_s = \mathbf{N}^d$ is defined as before:

$$(v_1, \ldots, v_d) \gtrsim (u_1, \ldots, u_d) \iff v_i \gtrsim u_i \text{ for all } i = 1, 2, \ldots, d;$$

the relation $\gtrsim$ on $A_\# = \mathbf{N}$ is the usual $\geq$ on $\mathbf{N}$. However, for $>$ on $A_s = \mathbf{N}^d$ we now choose another relation than before: we choose $>$ to be the empty relation. The relation $>$ on $A_\# = \mathbf{N}$ is the usual $>$ on $\mathbf{N}$. All requirements for $>$ and $\gtrsim$ from Definition 1 trivially hold.

For the interpretation $[f]$ of a symbol $f \in \Sigma$ of arity $n \geq 0$ we define

$$[f](\vec{x}_1, \ldots, \vec{x}_n) = F_1\vec{x}_1 + \cdots + F_n\vec{x}_n + \vec{f}$$

for n matrices $F_1, F_2, \ldots, F_n$ over $\mathbf{N}$ of size $d \times d$, and a vector $\vec{f} \in \mathbf{N}^d$. Note that now we no longer require that the upper-left elements of the matrices be positive. For the interpretation $[f_\#]$ of a marked symbol $f_\#$ corresponding to f of arity $n \geq 0$ we define

$$[f_\#](\vec{x}_1, \ldots, \vec{x}_n) = \vec{f}_1\vec{x}_1 + \cdots + \vec{f}_n\vec{x}_n + c_f$$

for n row vectors $\vec{f}_1, \ldots, \vec{f}_n$ over $\mathbf{N}$ of size d, and a constant $c_f \in \mathbf{N}$. Here $\vec{f}_i\vec{v}_i$ denotes the inner product, corresponding to matrix multiplication of a row vector by a column vector.

As before $[f]$ is monotone with respect to $\gtrsim$. The same holds for $[f_\#]$. By choosing all $[f]$ and $[f_\#]$ of this shape all requirements of a weakly monotone algebra are fulfilled.

To apply Theorem 3, part 2, for rules in R, we check whether $[\ell, \alpha] \gtrsim [r, \alpha]$ for all $\alpha : \mathcal{X} \to A$ for given rewrite rules as before. Checking whether $[\ell, \alpha] > [r, \alpha]$ for all α is required only for rules $\ell \to r$ in $\mathsf{DP}(R)$ being of sort #. This restriction can be written as $\vec{l}\vec{x} + c_l > \vec{r}\vec{x} + c_r$ for every vector $\vec{x}$ over $\mathbf{N}$, for vectors $\vec{l}, \vec{r}$ and numbers c_l, c_r implied by $\ell \to r$. This is equivalent to $\vec{l} \gtrsim \vec{r} \wedge c_l > c_r$. Similarly, for rules $\ell \to r$ in $\mathsf{DP}(R)$ the requirement $[\ell, \alpha] \gtrsim [r, \alpha]$ for all α is equivalent to $\vec{l} \gtrsim \vec{r} \wedge c_l \geq c_r$.

As before, it is not required to do this in one run, having $[\ell, \alpha] > [r, \alpha]$ for all α for all rules $\ell \to r$ in $\mathsf{DP}(R)$. If this holds for some of the rules $\ell \to r$ in $\mathsf{DP}(R)$, and for the others we have $[\ell, \alpha] \geq [r, \alpha]$, then by Theorem 3, part 2, we may remove the rules with "$>$" from $\mathsf{DP}(R)$, and continue with the rest. Now we observe that it is **not** allowed to remove rules from R: if we do then we get invalid results as is shown by the next example.

Example 4 Let R consist of the two rules $f(g(x)) \to f(h(x))$ and $h(x) \to g(x)$; obviously R is not terminating.

The TRS $\mathsf{DP}(R)$ consists of the rules $f_\#(g(x)) \to f_\#(h(x))$ and $f_\#(g(x)) \to h_\#(x)$. By choosing $A = \mathbf{N}$, and $[f_\#](n) = [h](n) = 1$ and $[h_\#](n) = [f](n) = [g](n) = 0$ for all $n \in \mathbf{N}$ we have a weakly monotone algebra, in which $[\ell, \alpha] \geq [r, \alpha]$ for all α for all rules $\ell \to r \in \mathsf{DP}(R) \cup R$, and "$>$" for the rules $h(x) \to g(x)$ and $f_\#(g(x)) \to h_\#(x)$. By removing these rules with "$>$" both from R and $\mathsf{DP}(R)$, the remaining proof obligation would be $\mathsf{SN}(\{f_\#(g(x)) \to f_\#(h(x))\}/\{f(g(x)) \to f(h(x))\})$, which is easily shown to hold.

It is also possible to keep the treatment of $\mathsf{SN}(\mathsf{DP}(R)_{\text{top}}/R)$ one-sorted on vectors of size d, choosing $>$ to be the strict part of $\gtrsim$. However, then the search space is much bigger because for every $f_\#$ n matrices of size $d \times d$ plus a vector have to be chosen, instead of n vectors of size d plus a constant, where n is the arity of f. Every termination proof in this one-sorted setting also yields a termination proof in the two-sorted setting as presented here, with the same bound on matrix and vector elements. This can be seen as follows. If there is a proof in the one-sorted setting then for at least one dependency pair the interpretation of the left-hand side strictly exceeds the interpretation of the right-hand side. Since $>$ is the strict part of $\gtrsim$, there is at least one dimension in which strict inequality appears. Then, by eliminating all other dimensions an interpretation in our two-sorted setting is found by which this particular dependency pair can be removed. By repeating the argument, the full termination proof in the one-sorted setting can be mimicked in our two-sorted setting. So, the two-sorted approach is as powerful but yields much smaller search spaces; hence, this two-sorted approach is preferred.

Before giving an example, we summarize the approach. To prove $\mathsf{SN}(R)$, we try to prove $\mathsf{SN}(D_{\text{top}}/R)$, where initially $D = \mathsf{DP}(R)$, or any subset of $\mathsf{DP}(R)$ as implied by variants of the dependency pair approach.

- Fix a dimension d.
- For every symbol $f \in \Sigma$ choose matrices $F_i \in \mathbf{N}^{d\times d}$ for $i = 1, 2, \ldots, n$ for n being the arity of f, and a vector $\vec{f} \in \mathbf{N}^d$.
- For every defined symbol $f \in \Sigma$ choose n row vectors $\vec{f}_1, \ldots, \vec{f}_n$ over $\mathbf{N}$ of size d for n being the arity of f, and a constant $c_f \in \mathbf{N}$, yielding the interpretation for $f_\#$.
- For every rule $\ell \to r \in R$ we check whether $L_i \gtrsim R_i$ for $i = 1, \ldots, k$ and $\vec{l} \gtrsim \vec{r}$ for the corresponding matrices L_i, R_i and vectors $\vec{l}, \vec{r}$ as defined above, similar to Section 4.
- For every rule $\ell \to r \in D$ we check whether $\vec{l} \gtrsim \vec{r} \ \wedge \ c_l \geq c_r$, for the corresponding vectors $\vec{l}, \vec{r}$ and scalars c_l, c_r defined by matrix / vector multiplications and inner products as described above.
- If all these requirements hold, then we remove all rules from D moreover satisfying $c_l > c_r$.
- If the remaining D is empty, we are finished; otherwise we repeat the process.

Example 5 Consider the TRS consisting of the following rule.

$$g(g(s(x), y), g(z, u)) \to g(g(u, s(z)), g(s(z), x))$$

Using the dependency pairs transformation, we get three dependency pairs, two of which do not contribute to a cycle within the approximated dependency graph. The remaining dependency pair is

$$g_\#(g(s(x), y), g(z, u)) \to g_\#(g(u, s(z)), g(s(z), x)).$$

We choose the following interpretation with dimension $d = 3$ (i.e., $A_s = \mathbf{N}^3$, $A_\# = \mathbf{N}$).

$$[g_\#](\vec{x}, \vec{y}) = (0,\ 0,\ 1) \cdot \vec{x} + (0,\ 1,\ 0) \cdot \vec{y}$$

$$[g](\vec{x}, \vec{y}) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \cdot \vec{y}$$

$$[s](\vec{x}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

For the rule $g(g(s(x), y), g(z, u)) \to g(g(y, z), g(x, s(u)))$ we obtain

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \gtrsim \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix},$$

and for the remaining dependency pair

$$g_\#(g(s(x), y), g(z, u)) \to g_\#(g(u, s(z)), g(s(z), x))$$

we obtain

$$(1,\ 0,\ 0) \cdot \vec{x} + (0,\ 0,\ 0) \cdot \vec{y} + (0,\ 0,\ 0) \cdot \vec{z} + (1,\ 0,\ 0) \cdot \vec{u} + 1$$

$$> (1,\ 0,\ 0) \cdot \vec{x} + (0,\ 0,\ 0) \cdot \vec{y} + (0,\ 0,\ 0) \cdot \vec{z} + (1,\ 0,\ 0) \cdot \vec{u}.$$

So, all rules are weakly decreasing, and the dependency pair is strictly decreasing and thus can be removed. Hence the system is terminating.

In Section 8 we will see that in experiments this dependency pair approach often succeeds where the basic matrix approach from Section 4 fails.

6 String Rewriting

Proving termination by matrix interpretations was first developed for string rewriting, as reported in [14, 15]. In this section we compare that approach with ours. In particular we consider $E_{\{1,d\}}$ termination proofs as described in [14]. These are of the following shape. To every symbol a a matrix $A \in \mathbf{N}^{d \times d}$ is assigned, satisfying $A_{1,1} > 0$ and $A_{d,d} > 0$. To every left-hand side ℓ a matrix $L \in \mathbf{N}^{d \times d}$ is assigned, obtained by replacing every a in ℓ by the corresponding matrix A, and interpreting concatenation as matrix multiplication. Similarly for every right hand side r a matrix R is assigned, being the identity matrix in case r is empty. For every rule $\ell \to r$ it is required that $L \gtrsim R$. Proving termination of type $E_{\{1,d\}}$ now means that all rules are removed for which $L_{i,j} > R_{i,j}$ for some i, j with $i, j \in \{1, d\}$.

We identify string rewriting with the special case of term rewriting in which all symbols have arity 1.

Theorem 6 *Let R be a string rewriting system having a termination proof in the way we described in Section 4 in dimension d. Then R has an $E_{\{1,d+1\}}$ termination proof in dimension $d+1$, as described in* [14].

Proof For $A \in \mathbf{N}^{d\times d}$ and $\vec{a} \in \mathbf{N}^d$ let $M(A, \vec{a})$ be the $(d+1)\times(d+1)$-matrix obtained from A by adding $\vec{a}$ as a column to the right of A, and next adding the row $(0, \ldots, 0, 1)$ below:

$$M\left(\begin{pmatrix} A_{1,1} & \cdots & A_{1,d} \\ \vdots & & \vdots \\ A_{d,1} & \cdots & A_{d,d} \end{pmatrix}, \begin{pmatrix} a_1 \\ \vdots \\ a_d \end{pmatrix}\right) = \begin{pmatrix} A_{1,1} & \cdots & A_{1,d} & a_1 \\ \vdots & & \vdots & \vdots \\ A_{d,1} & \cdots & A_{d,d} & a_d \\ 0 & \cdots & 0 & 1 \end{pmatrix}.$$

In our setting in dimension d we assign to every symbol a a matrix $A_1 \in \mathbf{N}^{d\times d}$ and a vector $\vec{a} \in \mathbf{N}^d$, with $[a](\vec{v}) = A_1\vec{v} + \vec{a}$. Since we assume a termination proof, we have for every rule $\ell \to r$ corresponding $d \times d$ matrices L_1, R_1 and size d vectors $\vec{l}, \vec{r}$ such that $L_1 \gtrsim R_1$ and $\vec{l} \gtrsim \vec{r}$, while we remove the rules for which moreover $l_1 > r_1$. Now for the $E_{\{1,d+1\}}$ termination proof we assign to a symbol a the corresponding matrix $M(A_1, \vec{a})$. Because of the shape of these matrices one easily checks that for every rule $\ell \to r$ for the corresponding matrices L, R we have $L = M(L_1, \vec{l})$ and $R = M(R_1, \vec{r})$. Now the requirements $L \gtrsim R$ follow from $L_1 \gtrsim R_1$ and $\vec{l} \gtrsim \vec{r}$, and for the rules to be removed we have the extra property

$$L_{1,d+1} = M(L_1, \vec{l})_{1,d+1} = l_1 > r_1 = M(R_1, \vec{r})_{1,d+1} = R_{1,d+1},$$

proving that $L_{i,j} > R_{i,j}$ for some i, j with $i, j \in \{1, d+1\}$. □

A challenging example was the system `Zantema-z086` from TPDB [2] consisting of the three rules

$$aa \to bc, \quad bb \to ac, \quad cc \to ab,$$

occurring in the RTA list of open problems [16] as number 104. A solution of this termination problem has been presented as Example 4 in [14] and first appeared as [15]. Up to renaming, swapping of coordinates and transposing, the solution in dimension 5 given there can be obtained by applying the transformation from the proof of Theorem 6 to the following proof by our basic approach:

$$[a](\vec{x}) = \begin{pmatrix} 1 & 0 & 0 & 3 \\ 0 & 0 & 2 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \quad [b](\vec{x}) = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 2 \\ 0 \\ 0 \end{pmatrix},$$

$$[c](\vec{x}) = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 2 & 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 1 \\ 0 \\ 3 \\ 0 \end{pmatrix}.$$

It is not clear how our basic approach relates to other instances of the matrix method for string rewriting as presented in [14]. Neither it is clear how our dependency pair version relates to the versions from [14] from a theoretical point of view. But applying the techniques in practice, our dependency pair version often outperforms the other versions.

For instance, using dependency pairs, we find the following termination proof for `Zantema-z086`. Here the dimension is 3 rather than 4. Because of this smaller dimension, this proof was found much faster: in 3.5 s rather than 102 s for the proof as presented above.

As a first step by simple counting arguments four of the six dependency pairs can be removed; this can be seen as matrix interpretations of dimension 1. It remains to prove $\mathsf{SN}(D_{\text{top}}/R)$, where R consists of the original rules

$$aa \to bc, \quad bb \to ac, \quad cc \to ab\,,$$

and D consists of the two rules

$$a_{\#}a \to b_{\#}c, \quad b_{\#}b \to a_{\#}c.$$

For these five symbols we get the following matrices and vectors:

$$[a](\vec{x}) = \begin{pmatrix} 0 & 0 & 0 \\ 2 & 0 & 1 \\ 2 & 1 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix}, \quad [b](\vec{x}) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 2 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix},$$

$$[c](\vec{x}) = \begin{pmatrix} 0 & 0 & 1 \\ 2 & 0 & 2 \\ 1 & 0 & 0 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 4 \\ 0 \end{pmatrix},$$

$$[a_{\#}](\vec{x}) = (0, 0, 1) \cdot \vec{x} + 1, \qquad [b_{\#}](\vec{x}) = (0, 0, 1) \cdot \vec{x}.$$

Now indeed we obtain "$\gtrsim$" for all rules in R and "$>$" for all rules in D, proving termination of R by Theorem 5.

Although in this latter proof in the substantial step five operation symbols are involved rather than three, because of the smaller dimension in total only 44 entries have to be established rather than 60 in the former proof. This is an indication why this latter proof could be found so much faster.

We stress that, as far as we know, no termination proof of `Zantema-z086` has been found not using matrix interpretations. Although the system itself is very small and has a nice symmetrical pattern, none of the known proofs provides any intuition why this system is terminating.

7 Implementation

The method described in the previous sections has been implemented as follows.

7.1 Goal

The basic algorithm finds a matrix interpretation that allows to remove rules from a termination problem, according to Theorem 3. It is called repeatedly until all rules have been removed.

Algorithm *Remove*:

- inputs
  - a pair of rewrite systems (R, S) over signature Σ
  - a flag $f \in \{\text{Full}, \text{Top}\}$
  - numbers d, b, b'
- outputs a matrix interpretation $[\cdot]$ such that
  - if $f = \text{Full}$, then the interpretation fulfills the conditions of Theorem 3, part *1*, for a nonempty TRS R';
  - if $f = \text{Top}$, then the interpretation fulfills the conditions of Theorem 3, part *2*, for a nonempty TRS R';

  and with the side conditions that

  - the interpretation $[\cdot]$ uses matrices of dimension $d \times d$;
  - all the coefficients in the matrices in the interpretations of operation symbols are in the range $0 \ldots 2^b - 1$;
  - all the coefficients in the matrices in the interpretations of rules are in the range $0 \ldots 2^{b'} - 1$.

We do this in such a way that an interpretation is found if and only if it exists with the given requirements. As described in Sections 4 and 5 the conditions for Theorem 3 give rise to constraints on coefficients in vectors and matrices that constitute the interpretations of the rules.

7.2 Steps

The implementation performs the following steps:

- Compress the rewriting systems (eliminate common subexpressions).
- Produce to a constraint system M for matrices.
- Transform to a constraint system N for numbers (integers).
- Transform to a constraint system B for Booleans.
- Transform to conjunctive normal form C.
- Call external SAT solver to find a satisfying assignment for C. If successful, apply reverse transformations to reconstruct solutions of B, N, M.

At each level, we consider *constraint systems* that are a collection of

- Declarations of variables;
- Definitions (of a "variable" whose value is given by an expression involving operations on other variables);
- Assertions (boolean combinations of inequalities between expressions).

We use the following example to present these steps in some detail.

Example 6

$$h(x, c(y, z)) \to h(c(s(y), x), z)$$

$$h(c(s(x), c(s(0), y)), z) \to h(y, c(s(0), c(x, z)))$$

7.3 Matrix Constraints

For each function symbol f of arity n from the signature, we declare variables $f_1, \dots, f_n$ for square matrices of dimension $d \times d$ and a variable f_0 for a (column) vector of dimension $d \times 1$. In Section 4 we wrote $F_1, \dots, F_n$ for $f_1, \dots, f_n$ and $\vec{f}$ for f_0. Identifying operation symbols with their interpretations we write $f = (f_1, \dots, f_n; f_0)$.

Then, for each term, its interpretation can be computed as a symbolic expression. For example, the translation of $l = h(x, c(y, z))$ is

$$h_0 + h_1 \cdot x + h_2 \cdot (c_0 + c_1 \cdot y + c_2 \cdot z),$$

and from $r = h(c(s(y), x), z)$ we get

$$h_0 + h_1 \cdot (c_0 + c_1 \cdot (s_0 + s_1 \cdot y) + c_2 \cdot x) + h_2 \cdot z.$$

For any variable v, denote by t_v the coefficient of v in the interpretation of t. If v does not occur, then this coefficient is 0 (the zero matrix). Denote by t_0 the absolute part of the interpretation of t.

Then the following constraint expresses that the interpretation is weakly compatible with the rewriting systems:

$$\bigwedge \{ l_v \gtrsim r_v \mid (l \to r) \in R \cup S, v \in \{0\} \cup \mathrm{Var}(l) \cup \mathrm{Var}(r) \}.$$

For example, for the first rule of the example system, we obtain the following.

$$\underbrace{h_0 + h_2 c_0 \gtrsim h_0 + h_1 c_0 + h_1 c_1 s_0}_{\text{absolute parts}}$$
$$\wedge \quad \underbrace{h_1 \gtrsim h_1 c_2}_{\text{coefficients of } x} \quad \wedge \quad \underbrace{h_2 c_1 \gtrsim h_1 c_1 s_1}_{\text{coefficients of } y} \quad \wedge \quad \underbrace{h_2 c_2 \gtrsim h_2}_{\text{coefficients of } z}$$

Finally, we require

$$\bigvee \{ l_0 \neq_1 r_0 \mid (l \to r) \in R \cup S \},$$

where $x \neq_1 y$ is defined for column vectors x, y by inequality in their first components. Together with $l_0 \gtrsim r_0$ this implies $l_0 > r_0$ as defined in Section 4. This implies that Theorem 3 can be applied for a nonempty set R'.

There are several possibilities for optimizations, that is, producing an equivalent constraint system that is smaller (contains fewer arithmetical operations). In the following, we describe one such method.

7.4 Compression

We present a method for "common subexpression elimination." It works by preprocessing the rewrite systems. Each occurrence of a common subexpression, called a pattern, is replaced by a fresh symbol that is added to the signature.

A *pattern* is a triple $(f, k, g) \in \Sigma \times \mathbb{N} \times \Sigma$. A pattern *occurs* at position p in a term t if f is the root symbol of the subterm of t at position p and the kth child of that occurrence of f is g. Assume f has arity m and g has arity n. If a pattern occurs more than once in $R \cup S$, then we define a new function symbol h of arity $m - 1 + n = q$.

Then, the translation $h \mapsto (h_1, \ldots, h_q; h_0)$ satisfies

$$\forall 0 \leq i < k : h_i = f_i,$$
$$\forall k \leq i \leq k+n-1 : h_i = f_k \cdot g_{i+1-k},$$
$$\forall k+n \leq i \leq q : h_i = f_{i+1-n}.$$

This means that we introduce $n+1$ additional variables to represent $f_k \cdot g_0, \ldots, f_k \cdot g_n$.

We replace each (nonoverlapping) occurrence of

$$f(t_1, \ldots, t_{k-1}, g(s_1, \ldots, s_m), t_{k+1}, \ldots, t_n),$$

for terms t_i, s_i, with

$$h(t_1, \ldots, t_{k-1}, s_1, \ldots, s_m, t_{k+1}, \ldots, t_n).$$

In the example, we use a new name a for the pattern $(c, 1, s)$, which occurs four times, obtaining

$$h(x, c(y, z)) \to h(a(y, x), z)$$
$$h(a(x, a(0, y)), z) \to h(y, a(0, c(x, z))).$$

The symbol a is represented by $(c_1 s_1, c_2; c_0 + c_1 s_0)$.

In the next steps, b stands for the pattern $(a, 1, 0)$ and d for $(h, 1, a)$, the result being

$$h(x, c(y, z)) \to d(y, x, z)$$
$$d(x, b(y), z) \to h(y, b(c(x, z))).$$

We will also apply the pattern $e = (h, 1, b)$. It occurs only once but still allows for a smaller matrix constraint system, since it allows to share the evaluation of $h_2 \cdot b_1$. The resulting rewrite system is

$$h(x, c(y, z)) \to d(y, x, z)$$
$$d(x, b(y), z) \to e(y, c(x, z)).$$

In all, the complete constraint system consists of the following parts. It has unknowns for the interpretations of the symbols in the signature

$$0 = (0_0), c = (c_1, c_2; c_0), h = (h_1, h_2; h_0), s = (s_1; s_0),$$

definitions for patterns

$$a = (c_1 \cdot s_1, c_2; c_0 + c_1 \cdot s_0) \qquad b = (c_2; a_0 + a_1 \cdot 0_0)$$
$$d = (h_1 \cdot a_1, h_1 c_2; h_0 + h_1 \cdot a_0) \quad e = (h_1, h_2 \cdot c_2; h_0 + h_2 \cdot b_0),$$

definitions for left- and right-hand sides of rules

$$l^1 = (h_1, h_2 c_1, h_2 \cdot c_2; h_0 + h_2 \cdot c_0) \quad r^1 = (d_2, d_1, h_2; d_0)$$
$$l^2 = (d_1, d_2 \cdot c_2, h_2; d_0 + d_2 \cdot b_0) \qquad r^2 = (e_2 \cdot c_1, h_1, e_2 \cdot c_2; e_0 + e_2 \cdot c_0),$$

and constraints between values

$$\begin{aligned}&\left(l_0^1 \gtrsim r_0^1 \wedge l_1^1 \gtrsim r_1^1 \wedge l_2^1 \gtrsim r_2^1 \wedge l_3^1 \gtrsim r_3^1\right)\\ &\quad\wedge \left(l_0^2 \gtrsim r_0^2 \wedge l_1^2 \gtrsim r_1^2 \wedge l_2^2 \gtrsim r_2^2 \wedge l_3^2 \gtrsim r_3^2\right)\\ &\quad\wedge \left(l_0^1 \neq_1 r_0^1 \vee l_0^2 \neq_1 r_0^2\right).\end{aligned}$$

In general, the number of unknowns of the constraint system is bounded by $|\Sigma| \cdot (a+1)$, where a is the maximal arity; the number of arithmetic operations is bounded by $\|R \cup S\| \cdot (a+1)$, where $\|R\|$ denotes the total size of a rewriting system (sum of sizes of terms in left- and right-hand sides); and the number of assertions is bounded by $2 \cdot |R \cup S| \cdot (a+1)$, where $|R|$ denotes the number of rules of R.

7.5 Numbers

In the following step, this matrix constraint system is transformed into a constraint system for numbers (integers). As expressions we allow variables, constants (for zero and one), and sums and products of expressions.

Each matrix variable a of dimensions $m \times n$ results in $m \cdot n$ integer variables $a_{i,j}$ with $1 \leq i \leq m, 1 \leq j \leq n$. Matrix addition and multiplication are translated according to their definitions.

An assertion $a \gtrsim b$ for matrices is translated into $\bigwedge\{a_{i,j} \geq b_{i,j} \mid i, j\}$. An assertion $a \neq_1 b$ for column vectors is translated into $a_1 \neq b_1$.

7.6 Bits

In this step, the integer constraint system is transformed into a Boolean constraint system. Each integer variable is translated into a sequence of propositional variables, denoting the number's binary expansion. Then, addition and multiplication are implemented as circuits. In this translation, we use a fixed bit width. The full results of addition and multiplication will have additional bits. Restricting to a fixed width implies that additional bits will be asserted to be zero.

An example for binary addition of bit width 3 is $(x_0, x_1, x_2)+(y_0, y_1, y_2)$. We introduce fresh variables r_0, r_1, r_2 for the result and c_0, c_1, c_2 for the carry bits, defined by

$$\begin{array}{ll} c_0 = x_0 \wedge y_0 & r_0 = x_0 \,\texttt{xor}\, y_0 \\ c_1 = \geq_2 (x_1, y_1, c_0) & r_1 = x_1 \,\texttt{xor}\, y_1 \,\texttt{xor}\, c_0 \\ c_2 = \geq_2 (x_2, y_2, c_1) & r_2 = x_2 \,\texttt{xor}\, y_2 \,\texttt{xor}\, c_1 \end{array}$$

together with the assertion "$c_2 =$ False" to ensure that the result (r_0, r_1, r_2) is correct (without overflow). Here $\geq_2 (x, y, z)$ is True iff at least two of the inputs are True; that is, $\geq_2 (x, y, z) = (x \vee y) \wedge (x \vee z) \wedge (y \vee z)$.

Multiplication uses iterated addition and shift operations. The product $(x_0, x_1, \ldots, x_n) \cdot (y_0, \ldots, y_n)$ is defined (recursively) to be $(a_0, c_1, \ldots, c_n)$, where the a_i, b_i, c_i are given by

$$\begin{gathered}
a_0 = x_0 \wedge y_0 \quad a_1 = x_0 \wedge y_1 \quad \ldots \quad a_n = x_0 \wedge y_n \\
(b_1, \ldots, b_n) = (x_1, \ldots, x_n) \cdot (y_0, \ldots, y_{n-1}) \\
(c_1, \ldots, c_n) = (a_1, \ldots, a_n) + (b_1, \ldots, b_n),
\end{gathered}$$

and we assert $x_1 \wedge y_n =$ False to prohibit overflow.

7.7 Conjunctive Normal Form

Finally, this Boolean constraint system is brought into conjunctive normal form (CNF) by a Tseitin transform: for each subexpression, we introduce a variable, together with CNF constraints that make the value of the variable equal to the value of the expression.

We use the following translations.

- Logical or: $x = (x_1 \vee \ldots \vee x_n)$ gives $(\neg x_1 \vee x) \wedge \ldots \wedge (\neg x_n \vee x) \wedge (x_1 \vee \ldots \vee v_n \vee \neg x)$,
- Logical and: $x = (x_1 \wedge \ldots \wedge x_n)$ gives $(\neg x_1 \vee \ldots \vee \neg x_n \vee x) \wedge (x_1 \vee \neg x) \wedge \ldots \wedge (x_n \vee \neg x)$,
- Exclusive-or: $x = (x_1 \texttt{ xor } x_2)$ gives $(x_1 \vee x_2 \vee \neg x) \wedge (x_1 \vee \neg x_2 \vee x) \wedge (\neg x_1 \vee x_2 \vee x) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x)$
- Majority (for three arguments) $\geq_2 (x_1, x_2, x_3) = x$ gives $(x_1 \vee x_2 \vee \neg x) \wedge (x_1 \vee x_3 \vee \neg x) \wedge (x_2 \vee x_3 \vee \neg x) \wedge (\neg x_1 \vee \neg x_2 \vee x) \wedge (\neg x_1 \vee \neg x_3 \vee x) \wedge (\neg x_2 \vee \neg x_3 \vee x)$

The algorithm for n-bit addition gives rise to $2n$ fresh variables and $14n$ clauses in the conjunctive normal form, and n-bit multiplication requires $\frac{3}{2}n^2$ fresh Boolean variables and $9n^2$ clauses.

For the running example of this section, we give the sizes of the constraint systems that are obtained when using matrix dimension 3, bit width 2 in interpretations, and bit width 3 in computations.

Constraint System for	Matrices	Numbers	Booleans	CNF
Size	93	1280	23530	60293

Here, “size” means total program size: sum of sizes of expressions in definitions and assertions. The conjunctive normal form has 5754 variables and 22268 clauses. A satisfying assignment is found in less than one second.

8 Performance Measurements

In this section we will analyze the performance of the matrix method under various setting on the TRS part of the Termination Problem Database 2006 (TPDB). This problem set was the basis of the 2006 Termination Competition and is available via [3]. It consists of 865 TRS, among which 686 could be proved to be terminating by any of the six participating tools; the rest contains both nonterminating TRSs and TRSs for which the termination behavior is unknown or established only by a human.

By *direct method* we mean pure matrix interpretations, that is, without use of any other termination methods such as dependency pairs. Likewise the method with *dependency pairs* stands for the combination of matrix interpretations with the dependency pairs framework. A huge number of methods have been developed for the dependency pairs framework. In our implementation we restrict ourselves to the most basic methods, since our goal is to analyze the strength of the matrix method. In particular, we use dependency graph approximation and the usable rules criterion [8, 9] and the subterm criterion [9] and compute strongly connected components as in [10]. Table 1 presents our results. We emphasize that we did not apply any of the following techniques: recursive path order, argument filtering, and semantic labeling, as they were considered sometimes to be essential for any serious termination tool. In the table, *dependency pairs* + stands for 'Termination Competition 2006' version of Jambox. In particular it is the extension by the transformation of applicative TRSs into functional form as described in [12], and rewriting of right-hand sides [20], lexicographic path order with argument filtering and semantic labeling.

For these results we took the time limit of 1 min, just as in the Termination Competition. However, this time was hardly ever consumed; the average computation time for all proofs is around 2 s. The full results, including all proofs generated by Jambox, are available via [17].

In the Termination Competition 2006 there were 17 examples that could be proved only by using our approach (all other participating tools failed).

In the subcategory "Relative Termination" of the Termination Competition 2006, Jambox scored 27 points, 24 of which were achieved by using only the *direct method*. Among these 24 proofs 12 are done with dimension one, 10 with dimension two, and 2 with dimension three.

Table 2 shows the average size of SAT encodings of the constraints for various dimensions d. It was calculated over the TRS part of the TPDB 2006 by using the bit-length $b = b' = 3$ (for all dimensions) and 1-min timeout per method. Minisat performs very well on the small dimensions 1 and 2. It even recognizes 99% of the unsolvable instances in around 2 s. Increasing the dimension to 3 or 4 has a negative impact on the performance. For dimension 4 the probability of running into timeout skyrockets to 17%, bringing along a loss of 5% of the solutions found with smaller dimensions. Nevertheless, the average time for successful attempts is around 2.5 s for dimension 3 and 7 s for dimension 4. Therefore, it is usually a good strategy to increase the dimension d stepwise, adjusting individual timeouts for every dimension.

Table 1 Performance results of several methods

Method	Dimension d	Initial bits b	Result bits b'	Cumulative YES score
Direct	1	4	5	139
Direct	2	2	3	222
Direct	3	3	4	238
Dependency pairs	1	4	5	463
Dependency pairs	2	2	3	533
Dependency pairs	3	2	3	540
Dependency pairs	4	2	3	541
Dependency pairs +	4	2	3	626

Table 2 Average size of SAT encodings of constraints for various dimensions

Dimension	Number of variables	Number of clauses	Time	Timeout (%)
1	800	4,500	0.2 s	0
2	4,700	30,000	1.9 s	1.2
3	13,500	88,000	6.8 s	6.9
4	26,500	175,000	13.8 s	16.6

9 Limitations and Reduction Lengths

Because of the special linear shape of our interpretations, our approach is not always successful. In this section we give examples for which matrix interpretations fail, and we discuss bounds on reduction lengths. It turns out that if a direct matrix interpretation can be given yielding "$>$" for all rules, then reduction lengths of terms of size n are bounded by C^n for some constant C. If a sequence of matrix interpretations is given as described in Section 4, then this no longer holds, but the reduction length is still bounded by a primitive recursive function. For the approach of Section 5 using dependency pairs this does not hold any more: we give an example of a TRS with a termination proof in this style allowing reduction lengths dominating Ackermann's function.

First we give three examples of terminating TRSs for which we show by three different arguments that a termination proof using only our matrix interpretation approach does not exist, not even in the setting of dependency pairs as described in Section 5. For all three TRSs, termination is easy to prove by other methods, for example, by semantic labeling or by analysis of strongly connected components in the approximated dependency graph.

Example 7 Consider the ground TRS consisting of the following two rules:

$$f(a, b) \to f(b, b), \quad f(b, a) \to f(a, a).$$

Assume we have a termination proof in the style of Section 5, with

$$[f_\#](\vec{x}, \vec{y}) \;=\; \vec{f}_1\vec{x} + \vec{f}_2\vec{y} + c_f$$

and $[a] = \vec{a}$ and $[b] = \vec{b}$. Then we have

$$\vec{f}_1\vec{a} + \vec{f}_2\vec{b} + c_f = [f_\#](\vec{a}, \vec{b}) \geq [f_\#](\vec{b}, \vec{b}) = \vec{f}_1\vec{b} + \vec{f}_2\vec{b} + c_f,$$

yielding $\vec{f}_1\vec{a} \geq \vec{f}_1\vec{b}$, and

$$\vec{f}_1\vec{b} + \vec{f}_2\vec{a} + c_f = [f_\#](\vec{b}, \vec{a}) \geq [f_\#](\vec{a}, \vec{a}) = \vec{f}_1\vec{a} + \vec{f}_2\vec{a} + c_f,$$

yielding $\vec{f}_1\vec{b} \geq \vec{f}_1\vec{a}$, where at least one of the inequalities should be strict, which is clearly impossible.

Example 8 Consider the TRS consisting of the single rule

$$f(x, x) \to f(x, g(x)).$$

Assume we have a termination proof in the style of Section 5, with

$$[f_\#](\vec{x},\vec{y}) \;=\; \vec{f_1}\vec{x} + \vec{f_2}\vec{y} + c_f, \quad [g](\vec{v}) \;=\; G\vec{v} + \vec{g}.$$

Choosing $\alpha(x) = \vec{0}$, we have

$$c_f \;=\; [f_\#(x,x),\alpha] \;>\; [f_\#(x,g(x)),\alpha] \;=\; \vec{f_2}\vec{g} + c_f,$$

which is impossible because of the non-negative coefficients of $\vec{f_2}$ and $\vec{g}$.

Example 9 Consider the well-known TRS originating from [4] consisting of the single rule

$$f(a,b,x) \to f(x,x,x).$$

Assume we have a termination proof in the style of Section 5, with

$$[f_\#](\vec{x},\vec{y},\vec{z}) \;=\; \vec{f_1}\vec{x} + \vec{f_2}\vec{y} + \vec{f_3}\vec{z} + c_f,$$

and $[a] = \vec{a}$ and $[b] = \vec{b}$. Choosing $\alpha(x) = \vec{a} + \vec{b}$, we have

$$\begin{aligned}\vec{f_1}\vec{a} + \vec{f_2}\vec{b} + \vec{f_3}(\vec{a}+\vec{b}) + c_f &= [f_\#(a,b,x),\alpha]\\ &> [f_\#(x,x,x),\alpha]\\ &= \vec{f_1}(\vec{a}+\vec{b}) + \vec{f_2}(\vec{a}+\vec{b}) + \vec{f_3}(\vec{a}+\vec{b}) + c_f,\end{aligned}$$

yielding $\vec{0} > \vec{f_1}\vec{b} + \vec{f_2}\vec{a}$, which is impossible because of non-negative coefficients.

Note the difference in the arguments for these three examples: in Example 8 the variable is interpreted by the smallest vector $\vec{0}$, in Example 9 it is interpreted by a large vector, and in Example 7 no variable occurs at all.

Next we investigate reduction lengths in TRSs for which termination is proved by matrix interpretations.

Lemma 7 *Let R be a* TRS *for which there is a direct matrix interpretation satisfying $[\ell,\alpha] > [r,\alpha]$ for all rules $\ell \to r$ and all α. Then there is a constant C such that reduction lengths of terms of depth n are bounded by C^n.*

Proof Without loss of generality we may restrict to ground terms, by which α in $[t,\alpha]$ may be omitted. For a vector $\vec{v}$ we define its norm $N(\vec{v})$ by $N(\vec{v}) = \sum_{i=1}^{d} \vec{v}_i$. For a matrix $A \in \mathbf{N}^{d\times d}$ we define $C_A = \sum_{i=1}^{d}\sum_{j=1}^{d} A_{ij}$, yielding

$$N(A\vec{v}) = \sum_{i=1}^{d}\sum_{j=1}^{d} A_{ij}\vec{v}_i \le \sum_{i=1}^{d}\sum_{j=1}^{d} A_{ij}\sum_{k=1}^{d}\vec{v}_i = C_A N(\vec{v})$$

for all $\vec{v} \in \mathbf{N}^d$. Choose C to be the sum of all C_A and $N(\vec{f})$, where f ranges over all Σ and A ranges over all matrices occurring in the interpretations of symbols. Then it is easily proved by induction on n that $N([t]) \le C^n$ for every term t of depth n. From the definition of N we obtain $[t]_1 \le C^n$. Since in every reduction step the first coordinate of the interpretation of the term strictly decreases, the length of a reduction starting in t cannot be longer than $[t]_1 \le C^n$. □

The next example shows that for this lemma it is essential that the proof can be given in one round yielding "$>$" for all rules.

Example 10 Let the TRS consist of the following four rules:

$$d(0) \to 0,\ d(s(x)) \to s(s(d(x))),\ e(0) \to s(0),\ e(s(x)) \to d(e(x)).$$

Clearly d describes doubling of natural numbers, and e describes exponentiation, yielding a reduction from $e^n(0)$ to $s^{f(n)}(0)$ of length $\Omega(f(n))$, where f is the super exponential function defined by $f(0) = 1$, $f(k+1) = 2^{f(k)}$. Now we prove termination by the direct method in a number of rounds.

By choosing a one-dimensional interpretation in which $[s](x) = [d](x) = x$, $[e](x) = x + 1$ and $[0] = 1$, we remove the rule $e(0) \to s(0)$. Next, by choosing

$$[s](\vec{x}) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot \vec{x} + \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad [0] = \begin{pmatrix} 1 \\ 0 \end{pmatrix},$$

$$[d](\vec{x}) = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} \cdot \vec{x}, \qquad [e](\vec{x}) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \cdot \vec{x}$$

we remove the rule $e(s(x)) \to d(e(x))$. Finally, by choosing a one-dimensional interpretation in which $[s](x) = x + 1$, $[d](x) = 3x$, and $[0] = 1$, we remove the remaining two rules, proving termination.

Although not exponential any more, a similar argument as in Lemma 7 shows that if a termination proof is given by the direct method in a number of rounds, reduction lengths are still bounded by a primitive recursive function in the size of the initial term. However, the next example shows that this does not hold any more when combining the method with dependency pairs as in Section 5, even when using only dimension $d = 1$.

Example 11 In [13] it was shown that the TRS consisting of the two rules

$$s(x) + (y + z) \to x + (s(s(y)) + z),$$
$$s(x) + (y + (z + w)) \to x + (z + (y + w))$$

admits reduction lengths dominating Ackermann's function. Take the dependency pair transformation. By choosing $[+_\#](x, y) = x + y$, $[+](x, y) = x + y + 1$ and $[s](x) = x$, that is, counting the number of $+$-symbols, we remove the three dependency pairs decreasing the number of $+$-symbols. In a second round the remaining dependency pairs are removed by choosing $[+_\#](x, y) = x$, $[+]$ arbitrary and $[s](x) = x + 1$, proving termination.

10 Conclusions

The idea of using matrix interpretations for termination proofs for string rewriting was developed by Hofbauer and Waldmann [14, 15]. It allowed them to prove termination for $\{aa \to bc, bb \to ac, cc \to ab\}$. In this paper we showed how to extend this approach to term rewriting successfully. A crucial ingredient is taking linear combinations of matrix interpretations for symbols of arity > 1.

In the results on the benchmark database TPDB we see a big jump when increasing the dimension from 1 (representing linear polynomial interpretations) to 2. Increasing the dimension from 2 to higher values yields only a minor improvement, while then the sizes of the satisfiability formulas strongly increase. By adding the dependency pairs approach, an enormous jump is achieved again: then using only linear polynomial interpretations ($d = 1$) already reaches a score of 463 points. In the Termination Competition 2006 this would have been a remarkable second place, and a third place if TTT [11] had participated, too. Finally, our highest score of 626 for dependency pairs + indeed yielded the second place for Jambox in this competition: just below the winning score of 638 for AProVE [7].

We stress that among the 626 TRSs for which termination was proved by Jambox, for several (17) of them Jambox and/or Matchbox were the only tools that found a proof in the Termination Competition 2006.

About the success of our approach in the Termination Competition we observe the following:

- Apparently the old idea of well-founded interpretations applies well when applied to vectors and linear operations on them, represented by matrices.
- Apparently applying this method in the setting of dependency pairs makes this even more powerful.
- Typically the search space for the corresponding interpretations is huge and intractable; therefore, direct search in this space has to be replaced by a more dedicated way of constraint solving. Apparently transforming the search problems to satisfiability problems and applying a state-of-the-art SAT solver serve well for this goal.

References

1. Arts, T., Giesl, J.: Termination of term rewriting using dependency pairs. Theor. Comp. Sci. **236**, 133–178 (2000)
2. Termination problems data base. http://www.lri.fr/~marche/tpdb/
3. Termination competition. http://www.lri.fr/~marche/termination-competition/
4. Dershowitz, N.: Termination of linear rewriting systems. In: Even, S., Kariv, O. (eds.) Proc. 8th Int. Coll. on Automata, Languages and Programming (ICALP-81). Lecture Notes in Computer Science, vol. 115, pp. 448–458. Springer (1981)
5. Eén, N., Biere, A.: Effective preprocessing in SAT through variable and clause elimination. In: Bacchus, F., Walsh, T. (eds.) Proc. 8th Int. Conf. Theory and Applications of Satisfiability Testing SAT 2005. Lecture Notes in Computer Science, vol. 3569, pp. 61–75. Springer (2005). Tool: http://www.cs.chalmers.se/Cs/Research/FormalMethods/MiniSat/
6. Endrullis, J., Waldmann, J., Zantema, H.: Matrix interpretations for proving termination of term rewriting. In: Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR '06). Lecture Notes in Computer Science, vol. 4130, pp. 574–588. Springer (2006)
7. Giesl, J., Schneider-Kamp, P., Thiemann, R.: Aprove 1.2: automatic termination proofs in the dependency pair framework. In: Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR '06). Lecture Notes in Computer Science, vol. 4130, pp. 281–286. Springer (2006)
8. Giesl, J., Thiemann, R., Schneider-Kamp, P.: The dependency pair framework: combining techniques for automated termination proofs. In: Proceedings of the 11th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2004). Lecture Notes in Computer Science, vol. 3452, pp. 301–331. Springer (2005)
9. Hirokawa, N., Middeldorp, A.: Dependency pairs revisited. In: van Oostrom, V. (ed.) Proceedings of the 15th Conference on Rewriting Techniques and Applications (RTA). Lecture Notes in Computer Science, vol. 3091, pp. 249–268. Springer (2004)

10. Hirokawa, N., Middeldorp, A.: Automating the dependency pair method. Inf. Comput. **199**, 172–199 (2005)
11. Hirokawa, N., Middeldorp, A.: Tyrolean termination tool. In: Giesl, J. (ed.) Proceedings of the 16th Conference on Rewriting Techniques and Applications (RTA). Lecture Notes in Computer Science, vol. 3467, pp. 175–184. Springer (2005)
12. Hirokawa, N., Middeldorp, A.: Uncurrying for termination. In: Proceedings of 3rd International Workshop on Higher-Order Rewriting (HOR), pp. 19–24. (2006)
13. Hofbauer, D., Lautemann, C.: Termination proofs and the length of derivations (preliminary version). In: Dershowitz, N. (ed.) Proceedings of the 3rd Conference on Rewriting Techniques and Applications (RTA). Lecture Notes in Computer Science, vol. 355, pp, 167–177. Springer (1989)
14. Hofbauer, D., Waldmann, J.: Proving termination with matrix interpretations. In: Pfenning, F. (ed.) Proceedings of the 17th Conference on Rewriting Techniques and Applications (RTA). Lecture Notes in Computer Science, vol. 4098, pp. 328–342. Springer (2006)
15. Hofbauer, D., Waldmann, J.: Termination of $\{aa \to bc, bb \to ac, cc \to ab\}$. Inf. Process. Lett. **98**(4), 156–158 (2006)
16. The RTA list of open problems. http://www.lsv.ens-cachan.fr/rtaloop/
17. Full results of Jambox. http://joerg.endrullis.de/jar07/
18. Zantema, H.: Termination of term rewriting: interpretation and type elimination. J. Symb. Comput. **17**, 23–50 (1994)
19. Zantema, H.: Termination. In: Term Rewriting Systems, by Terese, pp. 181–259. Cambridge University Press (2003)
20. Zantema, H.: Reducing right-hand sides for termination. In: Middeldorp, A., van Oostrom, V., van Raamsdonk, F., de Vrijer, R. (eds.) Processes, Terms and Cycles: Steps on the Road to Infinity: Essays Dedicated to Jan Willem Klop on the Occasion of His 60th Birthday. Lecture Notes in Computer Science, vol. 3838, pp. 173–197. Springer (2005)