Multi-Dimensional Interpretations for Termination of Term Rewriting

Akihisa Yamada[0000−0001−8872−2240]

National Institute of Advanced Industrial Science and Technology

Abstract. Interpretation methods constitute a foundation of termination analysis for term rewriting. From time to time remarkable instances of interpretation methods appeared, such as polynomial interpretations, matrix interpretations, arctic interpretations, and their variants. In this paper we introduce a general framework, the multi-dimensional interpretation method, that subsumes these variants as well as many previously unknown interpretation methods as instances. Employing the notion of derivers, we prove the soundness of the proposed method in an elegant way. We implement the proposed method in the termination prover NaTT and verify its significance through experiments.

1 Introduction

Term rewriting [2] is a formalism for reasoning about function definitions or functional programs. For instance, a term rewrite system (TRS) $\mathcal{R}_{\mathsf{fact}}$ [7] consisting of the following rewrite rules defines the factorial function:

$$\mathsf{fact}(0) \to \mathsf{s}(0) \quad \mathsf{fact}(\mathsf{s}(x)) \to \mathsf{mul}(\mathsf{s}(x), \mathsf{fact}(\mathsf{p}(\mathsf{s}(x)))) \quad \mathsf{p}(\mathsf{s}(x)) \to x$$

assuming that s, p, and mul are interpreted respectively as the successor, predecessor, and multiplication functions.

Analyzing whether a TRS *terminates*, meaning that the corresponding functional program responds or the function is well defined, has been an active research area for decades. Consequently, several fully automatic termination provers have been developed, e.g., AProVE [10], T$_T$T$_2$ [20], CiME [5], MU-TERM [23], and NaTT [34], and have been competing in the annual Termination Competitions (TermCOMP) [11].

Throughout their history, interpretation methods [25] have been foundational in termination analysis. They are categorized by the choice of well-founded carriers and the class of functions as which symbols are interpreted. *Polynomial interpretations* [22] use the natural numbers $\mathbb{N}$ as the carrier and interpretations are monotone polynomials, i.e., every variable has coefficient at least 1. Weakly monotone polynomials, i.e., zero coefficients, are allowed in the *dependency pair* method [1]. *Negative constants* are allowed using the max operator [15]. General combinations of polynomials and the max operator are proposed in both the standard [37] and the dependency pair settings [9]. *Negative coefficients* and thus

non-monotone polynomials are also allowed, but in a more elaborated theoretical framework [15,9].

These methods share the common carrier $\mathbb{N}$. In contrast, *matrix interpretations* [16,8] choose vectors over $\mathbb{N}$ as the carrier, and interpret symbols as affine maps over it. Although the carrier is generalized, matrix interpretations do not properly generalize polynomial interpretations, since not all polynomials are affine. This gap can be filled by *improved matrix interpretations*, that further generalize the carrier to square matrices [6], so that natural polynomial interpretations can be subsumed by matrix polynomials over 1×1 matrices. In *arctic interpretations* [19], the carrier consists of vectors over arctic naturals ($\mathbb{N} \cup \{-\infty\}$) or integers ($\mathbb{Z} \cup \{-\infty\}$), and interpretations are affine maps over it, where affinity is with respect to the *max/plus semiring*.

Having this many variations would be welcome if you are a user of a termination tool in which someone else has already implemented all of them. It would not be so if you are the developer of a termination tool in which you will have to implement all of them. Also, to ultimately trust termination tools, one needs to formalize proof methods using proof assistants and obtain trusted certifier that validates outputs of termination tools, see, e.g., IsaFoR/CeTA [31] or CoLoR/Rainbow [4] frameworks. Although some interpretation methods have already been formalized [28,30], adding missing variants one by one would cost a significant effort.

In this paper, we introduce a general framework for interpretation methods, which subsumes most of the above-mentioned methods as instances, namely, (max-)polynomial interpretations (with negative constants), (improved) matrix interpretations, and arctic interpretations, as well as a syntactic method called *argument filtering* [1,21]. Moreover, we obtain a bunch of previously unexplored interpretation methods as other instances.

After preliminaries, we start with a convenient fact about *reduction pairs*, a central tool in termination proving with dependency pairs (Section 3).

The first step to the main contribution is the use of *derivers* [24,33], which allow us to abstract away the mathematical details of polynomials or max-polynomials. We will obtain a key soundness result that derivers derive monotone interpretations from monotone interpretations (Section 4).

The second step is to extend derivers to multi-dimensional ones. This setting further generalizes (improved) matrix interpretations, so that max-polynomials, negative constants, and negative entries are allowed (Section 5). It will also be hinted that multi-dimensional derivers can emulate the effect of negative coefficients, although theoretical comparison is left for future work. We also show that our approach subsumes arctic interpretations by adding a treatment for $-\infty$ (Section 6). Although the original formulation by Koprowski and Waldmann [19] has some trickiness, we will show that our simpler formulation is sufficient.

As *strict monotonicity* is crucial for proving termination without dependency pairs, and is still useful with dependency pairs, we will see how to ensure strict monotonicity (Section 7). At this point, the convenient fact we have seen in Section 3 becomes crucial.

Finally, the proposed method is implemented in the termination prover NaTT, and experimental results are reported (Section 8). We evaluate various instances of our method, some corresponding to known interpretation methods and many others not. We choose two new instances to integrate to the NaTT strategy. The new strategy proved the termination of 20 more benchmarks than the old one, and five of them were not proved by any tool in TermCOMP 2020.

2   Preliminaries

We start with *order-sorted* algebras. Let $\mathcal{S} = \langle S, \sqsubseteq \rangle$ be a partially ordered set, where elements in S are called *sorts* and $\sqsubseteq$ is called the *subsort relation*. An *$\mathcal{S}$-sorted set* is an S-indexed family $A = \{A^\sigma\}_{\sigma \in S}$ such that $\sigma \sqsubseteq \tau$ implies $A^\sigma \subseteq A^\tau$. We write $A^{(\sigma_1, \ldots, \sigma_n)}$ for the set $A^{\sigma_1} \times \cdots \times A^{\sigma_n}$. A *sorted map* between $\mathcal{S}$-sorted sets X and A is a mapping f, written $f : X \to A$, such that $x \in X^\sigma$ implies $f(x) \in A^\sigma$.

An *$\mathcal{S}$-sorted signature* is an $S^* \times S$-indexed family $\mathcal{F} = \{\mathcal{F}_{\vec{\sigma}, \tau}\}_{\langle \vec{\sigma}, \tau \rangle \in S^* \times S}$ of function symbols.[1] When $f \in \mathcal{F}_{(\sigma_1, \ldots, \sigma_n), \tau}$, we say f has *rank* $(\sigma_1, \ldots, \sigma_n) \to \tau$ and *arity* n in $\mathcal{F}$. We may also view sorted sets and signatures as sets: having $a : \sigma \in A$ means $a \in A^\sigma$, and $f : \vec{\sigma} \to \tau \in \mathcal{F}$ means $f \in \mathcal{F}_{\vec{\sigma}, \tau}$.

Example 1. Consider sort Nat. We define the following {Nat}-sorted signatures:

- $\mathcal{N} := \{0 : () \to \mathtt{Nat}, \ 1 : () \to \mathtt{Nat}, \ 2 : () \to \mathtt{Nat}, \ \ldots\}$
- $\mathcal{N}_* := \mathcal{N} \cup \{* : (\mathtt{Nat}, \mathtt{Nat}) \to \mathtt{Nat}\}$
- $\mathcal{N}_+ := \mathcal{N} \cup \{+ : (\mathtt{Nat}, \mathtt{Nat}) \to \mathtt{Nat}\}$
- $\mathcal{N}_{\max} := \mathcal{N} \cup \{\mathtt{max} : (\mathtt{Nat}, \mathtt{Nat}) \to \mathtt{Nat}\}$

Let us abbreviate unions of signatures by concatenations of subscripts: for instance $\mathcal{N}_{*+\max}$ denotes $\mathcal{N}_* \cup \mathcal{N}_+ \cup \mathcal{N}_{\max}$. Next consider sorts Neg and Int with $\mathtt{Nat}, \mathtt{Neg} \sqsubseteq \mathtt{Int}$. We define the following {Nat, Neg, Int}-sorted signatures:

- $\mathcal{Z} := \mathcal{N} \cup \{0 : () \to \mathtt{Neg}, \ \mathtt{-1} : () \to \mathtt{Neg}, \ \mathtt{-2} : () \to \mathtt{Neg}, \ \ldots\}$
- $\mathcal{Z}_* := \mathcal{Z} \cup \mathcal{N}_* \cup \{* : (\mathtt{Neg}, \mathtt{Neg}) \to \mathtt{Nat}, \ * : (\mathtt{Int}, \mathtt{Int}) \to \mathtt{Int}\}$
- $\mathcal{Z}_+ := \mathcal{Z} \cup \mathcal{N}_+ \cup \{+ : (\mathtt{Neg}, \mathtt{Neg}) \to \mathtt{Neg}, \ + : (\mathtt{Int}, \mathtt{Int}) \to \mathtt{Int}\}$
- $\mathcal{Z}_{\max} := \mathcal{Z} \cup \mathcal{N}_{\max} \cup$
  $\{\mathtt{max} : (\mathtt{Nat}, \mathtt{Int}) \to \mathtt{Nat}, \ \mathtt{max} : (\mathtt{Int}, \mathtt{Nat}) \to \mathtt{Nat}, \ \mathtt{max} : (\mathtt{Int}, \mathtt{Int}) \to \mathtt{Int}\}$

For an $\mathcal{S}$-sorted signature $\mathcal{F}$, an *$\mathcal{F}$-algebra* $\langle A, [\cdot] \rangle$ consists of an $\mathcal{S}$-sorted set A called the *carrier* and a family $[\cdot]$ of mappings called the *interpretation* such that $[f] : A^{\vec{\sigma}} \to A^\tau$ whenever $f \in \mathcal{F}_{\vec{\sigma}, \tau}$.

Example 2. We consider the following *standard* interpretation $[\![\cdot]\!]$:

$$\cdots \quad [\![\mathtt{-2}]\!] := -2 \quad [\![\mathtt{-1}]\!] := -1 \quad [\![\mathtt{0}]\!] := 0 \quad [\![\mathtt{1}]\!] := 1 \quad [\![\mathtt{2}]\!] := 2 \quad \cdots$$

$$[\![*]\!](a, b) := a \cdot b \qquad [\![+]\!](a, b) := a + b \qquad [\![\mathtt{max}]\!](a, b) := \max(a, b)$$

Notice that $\langle \mathbb{N}, [\![\cdot]\!] \rangle$ is an $\mathcal{N}_{*+\max}$-algebra and $\langle \mathbb{Z}, [\![\cdot]\!] \rangle$ is a $\mathcal{Z}_{*+\max}$-algebra. Here, the {Nat}-sorted set $\mathbb{N}$ is defined by $\mathbb{N}^{\mathtt{Nat}} := \mathbb{N}$ and the {Nat, Neg, Int}-sorted set $\mathbb{Z}$ is defined by $\mathbb{Z}^{\mathtt{Nat}} := \mathbb{N}$, $\mathbb{Z}^{\mathtt{Neg}} := \{0, -1, -2, \ldots\}$ and $\mathbb{Z}^{\mathtt{Int}} := \mathbb{Z}$.

---

[1] In the literature, sorted signatures are given more assumptions such as monotonicity or regularity. For the purpose of this paper, these assumptions are not necessary.

Sorted Terms: Given an $\mathcal{S}$-sorted signature $\mathcal{F}$ and an $\mathcal{S}$-sorted set $\mathcal{V}$ of *variables*, the $\mathcal{S}$-sorted set $\mathcal{T}(\mathcal{F}, \mathcal{V})$ of *terms* is inductively defined as follows:

- $v \in \mathcal{T}(\mathcal{F}, \mathcal{V})^\sigma$ if $v \in \mathcal{V}^\sigma$;
- $f(s_1, \ldots, s_n) \in \mathcal{T}(\mathcal{F}, \mathcal{V})^\rho$ if $f \in \mathcal{F}_{\vec{\sigma},\tau}$, $(s_1, \ldots, s_n) \in \mathcal{T}(\mathcal{F}, \mathcal{V})^{\vec{\sigma}}$, and $\tau \sqsubseteq \rho$.

An interpretation $[\cdot]$ is extended over terms as follows: given $\alpha : \mathcal{V} \rightarrow A$, $[x]\alpha := \alpha(x)$ if $x \in \mathcal{V}^\sigma$, and $[f(s_1, \ldots, s_n)]\alpha := [f]([s_1]\alpha, \ldots, [s_n]\alpha)$. The $\mathcal{F}$-algebra $\langle \mathcal{T}(\mathcal{F}, \mathcal{V}), \cdot \rangle$ (which interprets f as the mapping that takes $(s_1, \ldots, s_n)$ and returns $f(s_1, \ldots, s_n)$) is called the *term algebra*, and a sorted map $\theta : \mathcal{V} \rightarrow \mathcal{T}(\mathcal{F}, \mathcal{V})$ is called a *substitution*. The term obtained by replacing every variable x by $\theta(x)$ in s is thus $s\theta$.

Term Rewriting: This paper is concerned with termination analysis for plain term rewriting. In this setting, there is only one sort 1, and we may identify a $\{1\}$-sorted set A and the set A^1. The set of variables appearing in a term s is denoted by $\mathsf{Var}(s)$. A *context* C is a term with a special variable $\square$ occurring exactly once. We denote by $C[s]$ the term obtained by substituting $\square$ by s in C. A *rewrite rule* is a pair of terms l and r, written $l \rightarrow r$, such that $l \notin \mathcal{V}$ and $\mathsf{Var}(l) \supseteq \mathsf{Var}(r)$. A *term rewrite system (TRS)* is a set $\mathcal{R}$ of rewrite rules, which induces the *root rewrite step* $\xrightarrow[\mathcal{R}]{\epsilon}$ and the *rewrite step* $\xrightarrow[\mathcal{R}]{}$ as the least relations such that $l\theta \xrightarrow[\mathcal{R}]{\epsilon} r\theta$ and $C[l\theta] \xrightarrow[\mathcal{R}]{} C[r\theta]$, for any rule $l \rightarrow r \in \mathcal{R}$, substitution θ, and context C. A TRS $\mathcal{R}$ is *terminating* iff no infinite rewriting $s_1 \xrightarrow[\mathcal{R}]{} s_2 \xrightarrow[\mathcal{R}]{} s_3 \xrightarrow[\mathcal{R}]{} \cdots$ is possible.

The dependency pair (DP) framework [1,14,13] is a *de facto* standard among automated termination provers for term rewriting. Here we briefly recapitulate its essence. The *root symbol* of a term $s = f(s_1, \ldots, s_n)$ is f and is denoted by $\mathsf{root}(s)$. The set of *defined* symbols in $\mathcal{R}$ is $\mathcal{D}_\mathcal{R} := \{\mathsf{root}(l) \mid l \rightarrow r \in \mathcal{R}\}$. We assume a fresh *marked* symbol $f^\sharp$ for every $f \in \mathcal{D}_\mathcal{R}$, and write $s^\sharp$ to denote the term $f^\sharp(s_1, \ldots, s_n)$ for $s = f(s_1, \ldots, s_n)$. A *dependency pair* of a TRS $\mathcal{R}$ is a rule $l^\sharp \rightarrow r^\sharp$ such that $\mathsf{root}(r) \in \mathcal{D}_\mathcal{R}$ and $l \rightarrow C[r] \in \mathcal{R}$ for some context C. The set of all dependency pairs of $\mathcal{R}$ is denoted by $\mathsf{DP}(\mathcal{R})$. A *DP problem* $\langle \mathcal{P}, \mathcal{R} \rangle$ is just a pair of TRSs.

Theorem 1 ([1]). *A TRS $\mathcal{R}$ is terminating iff the DP problem $\langle \mathsf{DP}(\mathcal{R}), \mathcal{R} \rangle$ is finite, i.e., there is no infinite chain $s_0 \xrightarrow[\mathsf{DP}(\mathcal{R})]{\epsilon} t_0 \xrightarrow[\mathcal{R}]{}{}^* s_1 \xrightarrow[\mathsf{DP}(\mathcal{R})]{\epsilon} t_1 \xrightarrow[\mathcal{R}]{}{}^* \cdots$.*

A number of techniques called *DP processors* that simplify or decompose DP problems are proposed; see [13] for a list of such processors. Among them, the central technique for concluding the finiteness of DP problems is the *reduction pair* processor, which will be reformulated in the next section.

3   Notes on Reduction Pairs

A reduction pair is a pair $\langle \succsim, \succ \rangle$ of order-like relations over terms with some conditions. Here we introduce two formulations of reduction pairs, one demanding

natural assumptions of orderings, and the other, reduction pair seed, demanding only essential requirements. The first formulation is useful when proving properties of reduction pairs, while the latter is useful when devising new reduction pairs. We will show that the two notions are essentially equivalent: one can always extend a reduction pair seed into a reduction pair of the former sense. Existing formulations of reduction pairs lie strictly in between the two.

Definition 1 (reduction pair). *A* (quasi-)order pair $\langle \succsim, \succ \rangle$ *is a pair of a quasi-order* $\succsim$ *and an irreflexive relation* $\succ \subseteq \succsim$ *satisfying* compatibility*:* $\succsim; \succ; \succsim \subseteq \succ$*. The order pair is* well-founded *if* $\succ$ *is well-founded.*

A reduction pair *is a well-founded order pair* $\langle \succsim, \succ \rangle$ *on terms, such that both* $\succsim$ *and* $\succ$ *are closed under substitutions, and* $\succsim$ *is closed under contexts. Here, a relation* $\sqsupset$ *is* closed under substitutions (resp. contexts) *iff* $s \sqsupset t$ *implies* $s\theta \sqsupset t\theta$ *for every substitution* θ *(resp.* $C[s] \sqsupset C[t]$ *for every context* C*).*

The above formulation of reduction pairs is strictly subsumed by standard definitions (e.g., [1,14,13]), where $\succ$ is not necessarily a subset of $\succsim$, and compatibility is weakened to either $\succsim; \succ \subseteq \succ$ or $\succ; \succsim \subseteq \succ$. Instead, $\succ$ is required to be transitive but this follows from our assumptions $\succ \subseteq \succsim$ and compatibility: $\succ; \succ \subseteq \succsim; \succ \subseteq \succ$. On one hand, this means that we can safely import existing results of reduction pairs into our formulation.

Theorem 2 (reduction pair processor [14,13]). *Let* $\langle \mathcal{P}, \mathcal{R} \rangle$ *be a DP problem and* $\langle \succsim, \succ \rangle$ *be a reduction pair such that* $\mathcal{P} \cup \mathcal{R} \subseteq \succsim$*. Then the DP problem* $\langle \mathcal{P}, \mathcal{R} \rangle$ *is finite if and only if* $\langle \mathcal{P} \setminus \succ, \mathcal{R} \rangle$ *is.*

Example 3. Consider again the TRS $\mathcal{R}_{\mathtt{fact}}$ of the introduction. Proving that $\mathcal{R}_{\mathtt{fact}}$ terminates in the DP framework boils down to finding a reduction pair $\langle \succsim, \succ \rangle$ satisfying (considering *usable rules* [1]):

$$\mathtt{p}(\mathtt{s}(x)) \succsim x \qquad\qquad \mathtt{fact}^{\sharp}(\mathtt{s}(x)) \succ \mathtt{fact}^{\sharp}(\mathtt{p}(\mathtt{s}(x)))$$

On the other hand, one may wonder whether Definition 1 might be too restrictive. We justify our formulation by uniformly extending general "reduction pairs" into reduction pairs that comply with Definition 1. This is possible for even more general pairs of relations than standard reduction pairs.

Definition 2 (reduction pair seed). *A* well-founded order seed *is a pair* $\langle W, S \rangle$ *of relations such that* S *is well-founded and* $S; W \subseteq S^{+}$*. A reduction pair seed is a well-founded order seed on terms such that both* W *and* S *are closed under substitutions, and* W *is closed under contexts.*

Now we show that every reduction pair seed $\langle W, S \rangle$ can be extended to a reduction pair $\langle \succsim, \succ \rangle$ such that $W \subseteq \succsim$ and $S \subseteq \succ$. Before that, the assumption $S; W \subseteq S^{+}$ of Definition 2 is generalized as follows.

Lemma 1. *If* $\langle W, S \rangle$ *is a well-founded order seed, then* $S; W^{*} \subseteq S^{+}$*.*

Proof. By induction on the number of W steps.   □

Theorem 3. *Let $\langle W, S \rangle$ be a well-founded order seed. Then $\langle \succsim, \succ \rangle$ is a well-founded order pair, where $\succsim := (W \cup S)^*$ and $\succ := (W^*; S)^+$.*

Proof. It is trivial that $\succsim$ is a quasi-order and $\succ \subseteq \succsim$ by definition. We show the well-foundedness of $\succ$ as follows: Suppose on the contrary we have an infinite sequence:

$$a_1 \ W^* \ b_1 \ S \ a_2 \ W^* \ b_2 \ S \ a_3 \ W^* \ b_2 \ S \ \cdots$$

Then using Lemma 1 $(S; W^* \subseteq S^+)$ we obtain $a_1 \ W^* \ b_1 \ S^+ \ b_2 \ S^+ \ \cdots$, which contradicts the well-foundedness of S.

Now we show compatibility. By definition we have $\succsim; \succ \subseteq \succ$, so it suffices to show $\succ; \succsim \subseteq \succ$. By induction we reduce the claim to $\succ; (W \cup S) \subseteq \succ$, that is, both $\succ; W \subseteq \succ$ and $\succ; S \subseteq \succ$. Using $S; W \subseteq S^+ = S; S^*$ we have

$$\succ; W = (W^*; S)^+; W = (W^*; S)^*; W^*; S; W$$
$$\subseteq (W^*; S)^*; W^*; S; S^* \subseteq \succ$$

The other case $\succ; S \subseteq \succ$ is easy from the definition. $\square$

Now we obtain the following corollary of Theorem 2 and Theorem 3.

Corollary 1. *Let $\langle \mathcal{P}, \mathcal{R} \rangle$ be a DP problem and $\langle W, S \rangle$ a reduction pair seed such that $\mathcal{P} \cup \mathcal{R} \subseteq W$. Then $\langle \mathcal{P}, \mathcal{R} \rangle$ is finite if and only if $\langle \mathcal{P} \setminus S, \mathcal{R} \rangle$ is.*

Notice that Definition 2 does not demand any order-like property, most notably transitivity. This is beneficial when developing new reduction pairs; for instance, *higher-order recursive path orders* [17] are known to be non-transitive, but form a reduction pair seed with their reflexive closure. Throughout the paper we use Definition 1, since it provides more useful and natural properties of orderings, which becomes crucial in Section 7.

4   Interpretation Methods as Derivers

Interpretation methods construct reduction pairs from $\mathcal{F}$-algebras, where $\mathcal{F}$ is the {1}-sorted signature of an input TRS or DP problem, and the carrier is a mathematical structure where a well-founded ordering $>$ is known. In the DP framework, weakly monotone $\mathcal{F}$-algebras play an important role.

Definition 3 (weakly monotone algebra). *A mapping $f : A_1 \times \cdots \times A_n \to A$ is* monotone *with respect to $\sqsupset$ if $f(a_1, \ldots, a_i, \ldots, a_n) \sqsupset f(a_1, \ldots, a_i', \ldots, a_n)$ whenever $a_1 \in A_1, \ldots, a_n \in A_n$, $a_i' \in A_i$, and $a_i \sqsupset a_i'$. A* weakly monotone *$\mathcal{F}$-algebra $\langle A, [\cdot], \geq, > \rangle$ consists of an $\mathcal{F}$-algebra $\langle A, [\cdot] \rangle$ and an order pair $\langle \geq, > \rangle$ such that every $[f]$ is monotone with respect to $\geq$.*

Example 4. Continuing Example 2, $\langle \mathbb{N}, [\![\cdot]\!], \geq, > \rangle$ is a weakly monotone $\mathcal{N}_{*+\max}$-algebra with the standard ordering $\langle \geq, > \rangle$. Notice that $\langle \mathbb{Z}, [\![\cdot]\!], \geq, > \rangle$ is not a weakly monotone $\mathcal{Z}_{*+\max}$-algebra, since multiplication on integers is not necessarily monotone. Nevertheless, it is a weakly monotone $\mathcal{Z}_{+\max} \cup \mathcal{N}_*$-algebra.

To ease presentation, from now on we assume that $\mathcal{F}$ is a $\{1\}$-sorted signature, while $\mathcal{G}$ is an $\mathcal{S}$-sorted signature. It is easy nevertheless to generalize our results to an arbitrary order-sorted signature $\mathcal{F}$.

Theorem 4 ([14]). *Let $\langle A, [\cdot], \geq, > \rangle$ be a weakly monotone $\mathcal{F}$-algebra such that $>$ is well-founded in A. Then $\langle [\geq], [>] \rangle$ is a reduction pair on $\mathcal{T}(\mathcal{F}, \mathcal{V})$, where*
$$s \, [\sqsupset] \, t \;:\Longleftrightarrow\; \forall \alpha : \mathcal{V} \to A. \; [s]\alpha \sqsupset [t]\alpha.$$

Moreover, using the term algebra any reduction pair $\langle \succsim, \succ \rangle$ on $\mathcal{T}(\mathcal{F}, \mathcal{V})$ can be seen as a well-founded $\mathcal{F}$-algebra $\langle \mathcal{T}(\mathcal{F}, \mathcal{V}), \cdot, \succsim, \succ \rangle$.

Example 5. Continuing Example 4, $\langle [\![\geq]\!], [\![>]\!] \rangle$ forms a reduction pair for signature $\mathcal{N}_{*+\texttt{max}}$. Notice that it does not for $\mathcal{Z}_{+\texttt{max}} \cup \mathcal{N}_*$, essentially because $>$ is not well-founded in $\mathbb{Z}$.

In order to prove the finiteness of a given DP problem, we need a weakly monotone $\mathcal{F}$-algebra for the signature $\mathcal{F}$ indicated by this problem, rather than for a predefined signature like $\mathcal{N}_{*+\texttt{max}}$. We fill the gap by employing the notion of *derivers* [24,33] to derive an $\mathcal{F}$-algebra from one of another signature $\mathcal{G}$.

Definition 4 (deriver). *An $\mathcal{F}/\mathcal{G}$-deriver is a pair of a sort $\delta \in \mathcal{S}$ and a mapping d, such that $d(f) \in \mathcal{T}(\mathcal{G}, \{x_1 : \delta, \ldots, x_n : \delta\})^\delta$ when f has arity n in $\mathcal{F}$. Given a* base $\mathcal{G}$-algebra $\langle A, [\cdot] \rangle$, we define the *derived $\mathcal{F}$-algebra $\langle A^\delta, d[\cdot] \rangle$ by*
$$d[f](a_1, \ldots, a_n) \; := \; [d(f)](x_1 \mapsto a_1, \ldots, x_n \mapsto a_n)$$

Example 6. Define a $\{\texttt{fact}^\sharp, \texttt{p}, \texttt{s} : 1 \to 1\}/\mathcal{Z}_{+\texttt{max}}$-deriver $\langle \texttt{Nat}, d \rangle$ by
$$d(\texttt{fact}^\sharp) \; := \; x_1 \qquad d(\texttt{s}) \; := \; x_1 + 1 \qquad d(\texttt{p}) \; := \; \texttt{max}(x_1 - 1, 0)$$

Note that $d(\texttt{p})$ has sort $\texttt{Nat}$, thanks to the rank $(\texttt{Int}, \texttt{Nat}) \to \texttt{Nat}$ of $\texttt{max}$ in $\mathcal{Z}_{\texttt{max}}$. The order pair $\langle d[\![\geq]\!], d[\![>]\!] \rangle$ satisfies the constraints given in Example 3.

Now we show that an $\mathcal{F}/\mathcal{G}$-deriver yields a weakly monotone $\mathcal{F}$-algebra if the base $\mathcal{G}$-algebra is known to be weakly monotone. Thus, Example 6 proves that $\mathcal{R}_{\texttt{fact}}$ is terminating. The next result about monotonicity is folklore:

Lemma 2. *A mapping $f : A^n \to A$ is monotone with respect to a quasi-order $\geq$ if and only if $a_1 \geq b_1, \ldots, a_n \geq b_n$ implies $f(a_1, \ldots, a_n) \geq f(b_1, \ldots, b_n)$.*

Proof. The "if" direction is due to the reflexivity of $\geq$, and the "only if" direction is easy by induction on n and the transitivity of $\geq$. □

Then monotonicity is carried over to the interpretation of terms, in the following sense. For two sorted maps $\alpha : X \to A$ and $\beta : X \to A$, we write $\alpha \geq \beta$ to mean that $\alpha(x) \geq \beta(x)$ for any $x \in X^\sigma$ and sort σ.

Lemma 3. *Let $\langle A, [\cdot], \geq, > \rangle$ be a weakly monotone $\mathcal{G}$-algebra and $s \in \mathcal{T}(\mathcal{G}, \mathcal{V})^\sigma$. If $\alpha \geq \beta$ then $[s]\alpha \geq [s]\beta$.*

Proof. By structural induction on s. The claim is trivial if s is a variable. Consider $s = f(s_1, \ldots, s_n)$. We have $[s_i]\alpha \geq [s_i]\beta$ for each $i \in \{1, \ldots, n\}$ by induction hypothesis. With Lemma 2 and the monotonicity of $[f]$, we conclude:

$$[s]\alpha = [f]([s_1]\alpha, \ldots, [s_n]\alpha) \ \geq \ [f]([s_1]\beta, \ldots, [s_n]\beta) = [s]\beta \qquad \qquad \square$$

Lemma 4. *Let $\langle \delta, d \rangle$ be an $\mathcal{F}/\mathcal{G}$-deriver and $\langle A, [\cdot], \geq, > \rangle$ a weakly monotone $\mathcal{G}$-algebra. Then $\langle A^\delta, d[\cdot], \geq, > \rangle$ is a weakly monotone $\mathcal{F}$-algebra.*

Proof. Suppose that f has arity n in $\mathcal{F}$, and for every $i \in \{1, \ldots, n\}$ that $a_i, b_i \in A^\delta$ and $a_i \geq b_i$. Then from Lemma 3,

$$\begin{aligned} d[f](a_1, \ldots, a_n) &= [d(f)](x_1 \mapsto a_1, \ldots, x_n \mapsto a_n) \\ &\geq [d(f)](x_1 \mapsto b_1, \ldots, x_n \mapsto b_n) = d[f](b_1, \ldots, b_n) \end{aligned}$$

With Lemma 2 we conclude that every $d[f]$ is monotone with respect to $\geq$, and hence $\langle A^\delta, d[\cdot], \geq, > \rangle$ is a weakly monotone $\mathcal{F}$-algebra. $\qquad \square$

Thus we conclude the soundness of the deriver-based interpretation method:

Theorem 5. *If $\langle \delta, d \rangle$ is a $\mathcal{F}/\mathcal{G}$-deriver, $\langle A, [\cdot], \geq, > \rangle$ is a weakly monotone $\mathcal{G}$-algebra and $>$ is well-founded in A^δ, then $\langle d[\geq], d[>] \rangle$ is a reduction pair.*

Proof. Immediate consequence of Lemma 4 and Theorem 4. $\qquad \square$

It should be clear that Theorem 5 with $\mathcal{G} = \mathcal{Z}_{+\max} \cup \mathcal{N}_*$ subsumes the polynomial interpretation method with negative constants [15, Lemma 4]. Their trick is to turn integers into naturals by applying $\max(\cdot, 0)$, as demonstrated in Example 6 in a syntactic manner. Theorem 5 gives a slightly more general fact that one can mix max and negative constants and still get a reduction pair. As far as the author knows, this fact has not been reported elsewhere, although natural max-polynomials without negative constants are known to yield reduction pairs [9, Section 4.1].

In addition, a syntactic technique known as *argument filtering* [1,21] is also a special case of Theorem 5. In the context of higher-order rewriting, Kop and van Raamsdonk generalized argument filters into *argument functions* [18, Definition 7.7], which, in the first-order case, correspond to derivers with $\mathcal{G}$ being a variant of $\mathcal{F}$. In these applications, base signatures and algebras are not *a priori* known, but are subject to be synthesized and analyzed.

5   Multi-Dimensional Interpretations

The *matrix interpretation method* [8] uses a well-founded weakly monotone algebra $\langle \mathbb{N}^m, [\cdot]_{\mathcal{M}at}, \geqq, \ggg \rangle$ over natural vectors, with an affine interpretation:

$$[f]_{\mathcal{M}at}(\vec{a}_1, \ldots, \vec{a}_n) = C_1 \vec{a}_1 + \cdots + C_n \vec{a}_n + \vec{c}$$

where $C_1, \ldots, C_n \in \mathbb{N}^{m \times m}$ and $\vec{c} \in \mathbb{N}^m$, and the following ordering:

Definition 5 ([8,19]). *Given an order pair $\langle \geq, > \rangle$ on A and a dimension $m \in \mathbb{N}$, we define the order pair $\langle \gg, \ggg \rangle$ on A^m as follows:*

$$(a_1, \ldots, a_m) \underset{(}{\gg}_{)} (b_1, \ldots, b_m) :\Longleftrightarrow a_1 \underset{(}{\geq}_{)} b_1 \wedge a_2 \geq b_2 \wedge \cdots \wedge a_m \geq b_m$$

Improved matrix interpretations [6] consider square matrices instead of vectors, and thus, in principle, matrix polynomials can be considered. Now we generalize these methods by extending derivers to multi-dimensional ones.

Definition 6 (multi-dimensional derivers). *An m-dimensional $\mathcal{F}/\mathcal{G}$-deriver consists of an m-tuple $\vec{\delta} \in \mathcal{S}^m$ of sorts and a mapping $\vec{d}$ such that $\vec{d}(f) \in \mathcal{T}(\mathcal{G}, \mathcal{X})^{\vec{\delta}}$, where $\mathcal{X} := \{x_{i,j} : (\vec{\delta})_j \mid i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}\}$ if f has arity n in $\mathcal{F}$. Given a $\mathcal{G}$-algebra $\langle A, [\cdot] \rangle$, the derived $\mathcal{F}$-algebra $\langle A^{\vec{\delta}}, \vec{d}[\cdot] \rangle$ is defined by*

$$\vec{d}[f](\vec{a}_1, \ldots, \vec{a}_n) := \left([(\vec{d}(f))_1]\alpha, \ldots, [(\vec{d}(f))_m]\alpha \right)$$

where α is defined by $\alpha(x_{i,j}) := (\vec{a}_i)_j$.

Example 7 ([8, Example 1]). The TRS of the single rule $\mathtt{f}(\mathtt{f}(x)) \to \mathtt{f}(\mathtt{g}(\mathtt{f}(x)))$ can be shown terminating by the following 2-dimensional matrix interpretation:

$$[\mathtt{f}]_{\mathcal{M}at}(\vec{a}) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \vec{a} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad [\mathtt{g}]_{\mathcal{M}at}(\vec{a}) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \vec{a} + \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

The 2-dimensional $\{\mathtt{f}, \mathtt{g}\}/\mathcal{N}_+$-deriver $\langle (\mathtt{Nat}, \mathtt{Nat}), \vec{d} \rangle$ defined by

$$\vec{d}(\mathtt{f}) = \begin{pmatrix} x_{11} + x_{12} \\ 1 \end{pmatrix} \qquad\qquad \vec{d}(\mathtt{g}) = \begin{pmatrix} x_{11} \\ 0 \end{pmatrix}$$

represents $[\cdot]_{\mathcal{M}at}$ as $\vec{d}[\![\cdot]\!]$, that is, $[\gg]_{\mathcal{M}at} = \vec{d}[\![\gg]\!]$ and $[\ggg]_{\mathcal{M}at} = \vec{d}[\![\ggg]\!]$.

Now we prove a counterpart of Theorem 5 for multi-dimensional derivers. The following lemma is one of the main results of this paper, which is somewhat surprisingly easy to prove.

Lemma 5. *For an m-dimensional $\mathcal{F}/\mathcal{G}$-deriver $\langle \vec{\delta}, \vec{d} \rangle$ and a weakly monotone $\mathcal{G}$-algebra $\langle A, [\cdot], \geq, > \rangle$, $\langle A^{\vec{\delta}}, \vec{d}[\cdot], \gg, \ggg \rangle$ is a weakly monotone $\mathcal{F}$-algebra.*

Proof. Let f have arity n in $\mathcal{F}$ and $\vec{a}_1, \ldots, \vec{a}_n, \vec{b}_1, \ldots, \vec{b}_n \in A^{\vec{\delta}}$ satisfy $\vec{a}_i \gg \vec{b}_i$. Define α and β by $\alpha(x_{i,j}) := (\vec{a}_i)_j$ and $\beta(x_{i,j}) := (\vec{b}_i)_j$. By assumption we have $\alpha \geq \beta$, and with Lemma 3 we have

$$\left(\vec{d}[f](\vec{a}_1, \ldots, \vec{a}_n) \right)_j = [(\vec{d}(f))_j]\alpha \geq [(\vec{d}(f))_j]\beta = \left(\vec{d}[f](\vec{b}_1, \ldots, \vec{b}_n) \right)_j$$

for every $j \in \{1, \ldots, m\}$. Hence $\vec{d}[f](\vec{a}_1, \ldots, \vec{a}_n) \gg \vec{d}[f](\vec{b}_1, \ldots, \vec{b}_n)$, and this concludes the proof due to Lemma 2. $\qquad\square$

Theorem 6. *For a multi-dimensional $\mathcal{F}/\mathcal{G}$-deriver $\langle \vec{\delta}, \vec{d} \rangle$ and a weakly mono-tone $\mathcal{G}$-algebra $\langle A, [\cdot], \geq, > \rangle$ such that $>$ is well-founded in $A^{(\vec{\delta})_1}$, $\langle \vec{d}[\geqq], \vec{d}[\ggg] \rangle$ is a reduction pair.*

Proof. Thanks to Lemma 5 and Theorem 4, it suffices to show that $\ggg$ is well-founded in $A^{\vec{\delta}}$. Suppose on the contrary that there exists an infinite sequence $\vec{a}_1 \ggg \vec{a}_2 \ggg \cdots$ with $\vec{a}_1, \vec{a}_2, \ldots \in A^{\vec{\delta}}$. Then we have $(\vec{a}_1)_1 > (\vec{a}_2)_1 > \cdots$ and $(\vec{a}_1)_1, (\vec{a}_2)_1, \ldots \in A^{(\vec{\delta})_1}$, contradicting the well-foundedness of $>$ in $A^{(\vec{\delta})_1}$. $\square$

It should be clear that every m-dimensional (improved) matrix interpretation can be expressed as an m-dimensional (or m^2-dimensional) $\mathcal{F}/\mathcal{N}_{*+}$-deriver. There are two more important consequences of Theorem 6: First, we can interpret symbols as non-affine maps even including max-polynomials; and second, since $>$ is not required to be well-founded in $A^{(\vec{\delta})_2}, \ldots, A^{(\vec{\delta})_m}$, examples that previously required non-monotone interpretations—and hence a stronger condition than Theorem 2—can be handled.

Example 8 (Excerpt of `AProVE_08/log`*).* Consider the TRS $\mathcal{R}_/$ consisting of

$$x - 0 \rightarrow x \qquad\qquad 0 \mathbin{/} y \rightarrow 0$$
$$\mathtt{s}(x) - \mathtt{s}(y) \rightarrow x - y \qquad \mathtt{s}(x) \mathbin{/} \mathtt{s}(y) \rightarrow (\mathtt{s}(x) - \mathtt{s}(y)) \mathbin{/} \mathtt{s}(y)$$

which defines (for simplicity, rounded up) natural division. Proving $\mathcal{R}_/$ terminating using dependency pairs boils down to finding a reduction pair $\langle \succeq, \succ \rangle$ such that (again considering usable rules)

$$x - 0 \succeq x \qquad \mathtt{s}(x) - \mathtt{s}(y) \succeq x - y \qquad \mathtt{s}(x) \mathbin{/^{\sharp}} \mathtt{s}(y) \succ (\mathtt{s}(x) - \mathtt{s}(y)) \mathbin{/^{\sharp}} \mathtt{s}(y)$$

A polynomial interpretation $[\cdot]_{\mathcal{P}ol}$ with negative coefficients such that

$$[0]_{\mathcal{P}ol} = 0 \quad [\mathtt{s}]_{\mathcal{P}ol}(x) = x + 1 \quad [/^{\sharp}]_{\mathcal{P}ol}(x, y) = x \quad [-]_{\mathcal{P}ol}(x, y) = \max(x - y, 0)$$

satisfies the above constraints, but one must validate the requirements of [15, Theorem 11]. In our setting, an $\mathcal{F}/\mathcal{Z}_{+\max}$-deriver $\langle (\mathtt{Nat}, \mathtt{Neg}), \vec{d} \rangle$ such that

$$\vec{d}(0) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \vec{d}(\mathtt{s}) = \begin{pmatrix} x_{1,1} + 1 \\ x_{1,2} - 1 \end{pmatrix} \quad \vec{d}(-) = \begin{pmatrix} \max(x_{1,1} + x_{2,2}, 0) \\ 0 \end{pmatrix} \quad \vec{d}(/^{\sharp}) = \begin{pmatrix} x_{1,1} \\ 0 \end{pmatrix}$$

yields a reduction pair satisfying the above constraints.

The intuition here is that the two dimensional interpretation of $\mathtt{s}^n(0)$ records n in the first coordinate and $-n$ in the second. Hence, one does not have to reconstruct $-n$ from n using the non-monotonic minus operation.

It seems plausible to the author that negative coefficients can be eliminated using the above idea; however, the increase of the dimension leads to more freedom in variables (the variable introduced to represent $-n$ may take values other than that) and so the ordering over terms may be different. It is left for future work to investigate whether this idea always works or not.

6 Arctic Interpretations

An *arctic interpretation* [19] $[\cdot]_{\mathcal{A}}$ is a matrix interpretation on the *arctic semiring*; that is, every interpretation $[f]_{\mathcal{A}}(\vec{x}_1, \ldots, \vec{x}_n)$ is of the form

$$C_1 \otimes \vec{x}_1 \oplus \cdots \oplus C_n \otimes \vec{x}_n \oplus \vec{c} \qquad (1)$$

where $\otimes$ and $\oplus$ denote the matrix multiplication and matrix addition in which the scalar addition is replaced by the max operation, and the scalar multiplication by addition; and entries of C_i and $\vec{c}$ are *arctic naturals* ($\mathbb{N}_{-\infty} := \mathbb{N} \cup \{-\infty\}$) or *arctic integers* ($\mathbb{Z}_{-\infty} := \mathbb{Z} \cup \{-\infty\}$). In addition, (1) must be *absolute positive*: $(\vec{c})_1 \geq 0$, so that $\langle \mathbb{N} \times \mathbb{N}_{-\infty}^{m-1}, [\cdot]_{\mathcal{A}}, \gtrapprox, \ggg \rangle$ or $\langle \mathbb{N} \times \mathbb{Z}_{-\infty}^{m-1}, [\cdot]_{\mathcal{A}}, \gtrapprox, \ggg \rangle$ forms a well-founded weakly monotone algebra.

The above formulation deviates from the original [19] in two ways. First, we do not introduce the special relation such that $-\infty \ggg -\infty$. Koprowski and Waldmann demanded this to ensure closure under general substitutions, but such a comparison cannot occur as we only need to consider substitutions that respect the carrier $\mathbb{N} \times \mathbb{Z}_{-\infty}^{m-1}$. Second, for arctic natural interpretations they relax absolute positiveness to *somewhere finiteness*: $(\vec{c})_1 \neq -\infty$ or $(C_i)_{1,1} \neq -\infty$ for some i. However, the two assumptions turn out to be equivalent.

Proposition 1. *Every arctic natural interpretation of form* (1) *is absolute positive iff it is somewhere finite.*

Proof. Clearly, absolute positiveness implies somewhere finiteness. For the other direction, since $(\vec{c})_1 \neq -\infty$ trivially implies absolute positiveness, suppose that $(\vec{c})_1 = -\infty$ and $(C_i)_{1,1} \neq -\infty$ for some i. We then know $(\vec{y})_1 \geq 0$, where $\vec{y} := C_1 \otimes \vec{x}_1 \oplus \cdots \oplus C_n \otimes \vec{x}_n$. Hence, by $\vec{c'} := (0, (\vec{c})_2, \ldots, (\vec{c})_m)$, we have $[f]_{\mathcal{A}}(\vec{x}_1, \ldots, \vec{x}_n) = \vec{y} \oplus \vec{c'}$, and this representation is absolute positive. □

One can easily obtain arctic interpretations via multi-dimensional derivers: consider a sort $\texttt{ANat}$ with $\texttt{Nat} \sqsubseteq \texttt{ANat}$ and $\{\texttt{Nat}, \texttt{ANat}\}$-sorted signature $\mathcal{N}_{\texttt{+max-}\infty}$, extending $\mathcal{N}_{\texttt{+max}}$ with

$$\texttt{-}\infty : () \to \texttt{ANat} \qquad \texttt{+} : (\texttt{ANat}, \texttt{ANat}) \to \texttt{ANat}$$

$$\texttt{max} : (\texttt{Nat}, \texttt{ANat}) \to \texttt{Nat} \quad \texttt{max} : (\texttt{ANat}, \texttt{Nat}) \to \texttt{Nat} \quad \texttt{max} : (\texttt{ANat}, \texttt{ANat}) \to \texttt{ANat}$$

and extend the standard interpretation $[\![\cdot]\!]$ accordingly. We omit the easy proof of the following fact and the counterpart for arctic integer interpretations.

Proposition 2. *Every absolute positive arctic natural interpretation* $[\cdot]_{\mathcal{A}}$ *is represented as* $\vec{d}[\![\cdot]\!]$ *via an* $\mathcal{F}/\mathcal{N}_{\texttt{+max-}\infty}$-*deriver* $\langle (\texttt{Nat}, \texttt{ANat}, \ldots, \texttt{ANat}), \vec{d} \rangle$.

Notice that, in practice, this requires us to deal with $-\infty$ by ourselves since there is no standard SMT theory [3] that supports arithmetic with $-\infty$.

7   Strict Monotonicity

Before the invention of dependency pairs [1], strictly monotone algebras were necessary for proving termination by interpretation methods, and they constitute a sound and complete method for proving termination of TRSs.

Definition 7. *A* strictly *monotone $\mathcal{F}$-algebra is a weakly monotone $\mathcal{F}$-algebra $\langle A, [\cdot], \geq, > \rangle$ such that $\langle A, [\cdot] \rangle$ is monotone with respect to both $\geq$ and $>$.*

Theorem 7 (cf. [36]). *A TRS $\mathcal{R}$ is terminating if and only if there is a strictly monotone well-founded $\mathcal{F}$-algebra $\langle A, [\cdot], \geq, > \rangle$ such that $\mathcal{R} \subseteq [>]$.*

Moreover, strict monotonicity is a desirable property in the DP framework as it allows one to remove not only dependency pairs but also rewrite rules.

Theorem 8 ([12]). *A DP problem $\langle \mathcal{P}, \mathcal{R} \rangle$ is finite if $\langle \mathcal{P} \setminus [>], \mathcal{R} \setminus [>] \rangle$ is, where $\langle A, [\cdot], \geq, > \rangle$ is a strictly monotone well-founded $\mathcal{F}$-algebra such that $\mathcal{P} \cup \mathcal{R} \subseteq [\geq]$.*

We now state a criterion that ensures the strict monotonicity of multi-dimensional interpretation obtained via derivers. Below we write d_i to mean the mapping defined by $d_i(f) := \big(\vec{d}(f)\big)_i$.

Theorem 9. *Let $\langle \vec{\delta}, \vec{d} \rangle$ be an m-dimensional $\mathcal{F}/\mathcal{G}$-deriver and $\langle A, [\cdot], \geq, > \rangle$ a weakly monotone $\mathcal{G}$-algebra. Suppose that when f has arity n in $\mathcal{F}$ and $i \in \{1, \ldots, n\}$, $\alpha(x_{i,1}) > a$ implies $[d_1(f)]\alpha > [d_1(f)]\alpha(x_{i,1} \mapsto a)$ for any $\alpha : \mathcal{X} \to A$ and $a \in A$. Then $\langle A^{\vec{\delta}}, \vec{d}[\cdot], \geqq, \gg \rangle$ is a strictly monotone $\mathcal{F}$-algebra.*

Proof. We only prove strict monotonicity as we already know weak monotonicity by Lemma 5. So suppose that f has arity n in $\mathcal{F}$, $\vec{a}_1, \ldots, \vec{a}_i, \ldots, \vec{a}_n, \vec{a}_i' \in A^{\vec{\delta}}$ and $\vec{a}_i \gg \vec{a}_i'$. For the first coordinate, define α by $\alpha(x_{k,j}) := (\vec{a}_k)_j$. Then, first using the assumption, and then Lemma 3, we conclude

$$
\begin{aligned}
d_1[f](\vec{a}_1 \ldots, \vec{a}_i, \ldots, \vec{a}_n) &= [d_1(f)]\alpha \\
&> [d_1(f)]\alpha(x_{i,1} \mapsto (\vec{a}_i')_1) \\
&\geq [d_1(f)]\alpha(x_{i,1} \mapsto (\vec{a}_i')_1, x_{i,2} \mapsto (\vec{a}_i')_2, \ldots, x_{i,m} \mapsto (\vec{a}_i')_m) \\
&= d_1[f](\vec{a}_1, \ldots, \vec{a}_i', \ldots, \vec{a}_n)
\end{aligned}
$$

For the other coordinates, thanks to the "new" assumption $> \subseteq \geq$ in Definition 1 we have $\vec{a}_i \geqq \vec{a}_i'$. Then the weak monotonicity ensures $\vec{d}[f](\vec{a}_1, \ldots, \vec{a}_i, \ldots \vec{a}_n) \geqq \vec{d}[f](\vec{a}_1, \ldots, \vec{a}_i', \ldots, \vec{a}_n)$, from which we deduce for each $j \in \{2, \ldots, m\}$,

$$
d_j[f](\vec{a}_1, \ldots, \vec{a}_i, \ldots, \vec{a}_n) \geq d_j[f](\vec{a}_1, \ldots, \vec{a}_i', \ldots, \vec{a}_n) \qquad \square
$$

Although the above result and proof do not look surprising, it would be worth noticing that the statement is false in the standard formulation allowing $> \not\subseteq \geq$ (as even in [8]).

Example 9. Consider the following apparently monotone matrix interpretation:

$$[\mathtt{f}]\left(\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}\right) := \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_1 \\ a_1 \end{pmatrix}$$

If one had $a_1 > b_1$ but $a_1 \not\geq b_1$, then

$$[\mathtt{f}]\left(\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}\right) = \begin{pmatrix} a_1 \\ a_1 \end{pmatrix} \begin{matrix} > \\ \not\geq \end{matrix} \begin{pmatrix} b_1 \\ b_1 \end{pmatrix} = [\mathtt{f}]\left(\begin{pmatrix} b_1 \\ a_2 \end{pmatrix}\right) \quad \text{even though} \quad \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \gg \begin{pmatrix} b_1 \\ a_2 \end{pmatrix}.$$

So $[\mathtt{f}]$ would not be monotone with respect to $\gg$.

8 Implementation and Experiments

Multi-dimensional interpretations are implemented in the termination prover NaTT version 2.0^2, using a *template*-based approach.

Definition 8. *An m-dimensional $\mathcal{F}/\mathcal{G}$-deriver* template $\langle \vec{\delta}, \vec{d} \rangle$ *with $\mathcal{S}$-sorted set $\mathcal{W}$ of template variables is defined as in Definition 6, but allowing $\vec{d}(f) \in \mathcal{T}(\mathcal{G}, \mathcal{W} \cup \mathcal{X})^{\vec{\delta}}$. Its* instance *according to a substitution $\theta : \mathcal{W} \to \mathcal{T}(\mathcal{G}, \emptyset)$ is the $\mathcal{F}/\mathcal{G}$-deriver $\langle \vec{\delta}, \vec{d}\theta \rangle$, defined by $\vec{d}\theta(f) := (d_1(f)\theta, \dots, d_m(f)\theta)$.*

In the implementation, we fix $\mathcal{G} = \mathcal{Z}_{\mathtt{+max}} \cup \mathcal{N}_*$ and the base weakly monotone $\mathcal{G}$-algebra $\langle \mathbb{Z}, [\![\cdot]\!], \geq, > \rangle$. Given an m-dimensional deriver template $\langle \vec{\delta}, \vec{d} \rangle$ with $\mathcal{W}$, our interest is now to find $\theta : \mathcal{W} \to \mathbb{Z}$ such that $\vec{d}\theta[s] \geq \vec{d}\theta[t]$ for every $(s,t) \in \mathcal{P} \cup \mathcal{R}$ for the DP problem $\langle \mathcal{P}, \mathcal{R} \rangle$ of concern, thanks to Theorem 6. NaTT reduces this problem into an SMT problem and passes it to a backend SMT solver. The page limit is not enough to detail the reduction; in short, the constraint $\vec{d}\theta[s] \gtrapprox \vec{d}\theta[t]$ is reduced into a Boolean formula over atoms of form $a * \langle v_1, i_1 \rangle * \cdots * \langle v_n, i_n \rangle \geq b * \langle v_1, i_1 \rangle * \cdots * \langle v_n, i_n \rangle$, where $a, b \in \mathcal{T}(\mathcal{G}, \mathcal{W})$, and $\langle v_1, i_1 \rangle \dots, \langle v_n, i_n \rangle \in (\mathsf{Var}(s) \cup \mathsf{Var}(t)) \times \{1, \dots, m\}$ are seen as variables. Internally NaTT uses a distribution approach [30], whose soundness crucially relies on the fact that the only rank of $*$ is $(\mathtt{Nat}, \mathtt{Nat}) \to \mathtt{Nat}$ in the signature $\mathcal{G}$. Then each atom is further reduced to (1) $a = b$ if $(\vec{\delta})_{i_j} = \mathtt{Int}$ for some j, (2) $a \geq b$ if $\left| \{j \mid (\vec{\delta})_{i_j} = \mathtt{Neg}\} \right|$ is even, and (3) $a \leq b$ otherwise. Due to the last step, having coordinates of sort $\mathtt{Int}$ leads to a stronger constraint when ordering terms. Finally, the resulting formula, containing only template variables, is passed to the SMT solver Z3 4.8.10 [26] and a satisfying solution $\theta : \mathcal{W} \to \mathbb{Z}$ is a desired substitution.

To verify the practical significance of the method, we evaluated various templates in a simple dependency pair setting. For a function symbol f of arity $n \geq 2$, the k-th coordinate of template $\vec{d}(f)$ is chosen from

− sum: $w + \sum_{i=1}^{n}(b * x_{i,k})$,

Table 1. Evaluation of 2-dimensional templates.

#	Coordinate 1		Coordinate 2		YES	New	Time	Known as
1	sum	Nat	-	-	512	-	00:36:12	polynomial [1]
2	sum	Int	-	-	559	-	00:52:37	negative constant [15]
3	sum-sum	Nat	sum-sum	Nat	636	-	04:18:05	matrix [8]
4	sum-sum	Int	sum	Neg	602	10	04:00:05	new
5	sum-sum	Int	sum-sum	Int	542	0	25:07:04	new
6	sum-sum	Int	max	Neg	585	8	14:58:41	new
7	max	Int	-	-	560	-	00:58:58	max-polynomial [9][3]
8	max-max	Nat	max-max	Nat	552	3	12:33:43	arctic natural [19][4]
9	max-max	Int	max-max	Int	580	2	22:35:29	arctic integer [19][4]
10	max-max	Nat	sum	Nat	577	0	03:48:46	new
11	max-max	Int	sum	Neg	584	2	06:53:34	new
12	max-sum	Int	sum	Neg	592	4	06:59:22	new
13	heuristic	Int	sum	Neg	648	9	04:55:43	new

- max: $\max_{i=1}^{n} b * (w + x_{i,k})$,
- sum-sum: $w + \sum_{i=1}^{n} \sum_{j=1}^{m} b * x_{i,j}$,
- max-max: $\max_{i=1}^{n} \max_{j=1}^{m} b * (w + x_{i,j})$,
- sum-max: $\sum_{i=1}^{n} \max_{j=1}^{m} b * (w + x_{i,j})$,
- max-sum: $\max_{i=1}^{n} (w + \sum_{j=1}^{m} b * x_{i,j})$, and
- a heuristic choice [35] between sum-sum and max-sum,

where b and w introduce fresh template variables, b ranges over $\{0, 1\}$ and the sort of w is up to further choice. The sort of the first coordinate is turned to Nat by applying $\max(\cdot, 0)$ if necessary.

Experiments are run on the StarExec environment [29], with timeout of 300 seconds. The benchmarks are the 1507 TRSs from the TRS Standard category of the *termination problem database* 11 [32]. Due to the huge search space, we evaluate templates of dimensions up to 2. A part of the results are summarized in Table 1. Full details of the experiments are made available at `http://www.trs.cm.is.nagoya-u.ac.jp/NaTT/multi/`.

In the table, each coordinate is represented by the template and the sort of w. In terms of the number of successful termination proofs indicated in the "YES" column, the classical matrix interpretations (row #3) are impressively strong. Nevertheless, it is worth considering a negative coordinate (#4) as it gives 10 termination proofs that the previous version of NaTT could not find, indicated in the "New" column. In contrast, considering whole integers in the second coordinate (#5) does not look promising as the runtime grows significantly. Concerning "max", we observe that its use in the second coordinate (#6)

---

[3] This template is a subset of integer max-polynomials [9], although the fact that it yields a reduction pair is new.

[4] In our implementation, negative infinity is not supported. Instead, similar effect is emulated by zero coefficients.

Table 2. Experiments with combined strategies

Strategy	YES	New to NaTT	New to TermCOMP	Time
Old Strategy	861	0	0	3:46:12
With #4	874	13	3	4:14:09
With #13	871	10	1	4:26:14
With #4 and #13	881	20	5	4:49:50

degrades the performance. Using "max" in both coordinates *a la* arctic interpretations (#8, #9) gives a few new termination proofs, but the impact in the runtime is significant in the current implementation. The runtime improves by replacing some occurrences of "max" by "sum" (#10–12), while the power does not seem defected. In terms of the number of termination proofs, the heuristic choice of "sum-sum" and "max-sum" in the first coordinate (#13) performed the best among the evaluated templates.

From these experiments, we pick templates #4 and #13 to incorporate in the NaTT default strategy. The final results are summarized in Table 2. Although the runtime noticeably increases, adding both #4 and #13 gives 20 more examples solved, and five of them (`AProVE_09_Inductive/log` and four in `Transformed_CSR_04/`) were not solved by any tool in the TermCOMP 2020.

9   Conclusion

In this paper we introduced a deriver-based multi-dimensional interpretation method. The author expects that the result makes the relationships between existing interpretation methods cleaner, and eases the task of developing and maintaining termination tools. Moreover, it yields many previously unknown interpretation methods as instances, proving the termination of some standard benchmarks that state-of-the-art termination provers could not.

Theoretical comparison with negative coefficients is left for future work, and the use of $-\infty$ is not implemented yet. Also since this work broadens the search space, it is interesting to heuristically search for derivers rather than fixing some templates. Derivers of higher dimensions seem also interesting to explore. Finally, although the proposed method is implemented in the termination prover NaTT, there is no guarantee that the implementation is correct. In order to certify termination proofs that use multi-dimensional derivers, one must formalize the proofs in this paper, extend the certifiable proof format [27], and implement a verified function to validate such proofs.

References

1. Arts, T., Giesl, J.: Termination of term rewriting using dependency pairs. Theor. Compt. Sci. **236**(1–2), 133–178 (2000). https://doi.org/10.1016/S0304-3975(99)00207-8
2. Baader, F., Nipkow, T.: Term rewriting and all that. Cambridge University Press (1998)
3. Barrett, C.W., Tinelli, C.: Satisfiability modulo theories. In: Clarke, E.M., Henzinger, T.A., Veith, H., Bloem, R. (eds.) Handbook of Model Checking, pp. 305–343. Springer (2018). https://doi.org/10.1007/978-3-319-10575-8_11
4. Blanqui, F., Koprowski, A.: CoLoR: a Coq library on well-founded rewrite relations and its application to the automated verification of termination certificates. Math. Struct. Comput. Sci. **21**(4), 827–859 (2011). https://doi.org/10.1017/S0960129511000120
5. Contejean, É., Courtieu, P., Forest, J., Pons, O., Urbain, X.: Automated certified proofs with CiME3. In: Schmidt-Schauß, M. (ed.) RTA 2011. LIPIcs, vol. 10, pp. 21–30. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2011). https://doi.org/10.4230/LIPIcs.RTA.2011.21
6. Courtieu, P., Gbedo, G., Pons, O.: Improved matrix interpretation. In: van Leeuwen, J., Muscholl, A., Peleg, D., Pokorný, J., Rumpe, B. (eds.) SOFSEM 2010. LNCS, vol. 5901, pp. 283–295. Springer (2010). https://doi.org/10.1007/978-3-642-11266-9_24
7. Dershowitz, N.: 33 examples of termination. In: Comon, H., Jounnaud, J.P. (eds.) Term Rewriting. pp. 16–26. Springer (1995). https://doi.org/10.1007/3-540-59340-3_2
8. Endrullis, J., Waldmann, J., Zantema, H.: Matrix interpretations for proving termination of term rewriting. J. Autom. Reason. **40**(2-3), 195–220 (2008). https://doi.org/10.1007/s10817-007-9087-9
9. Fuhs, C., Giesl, J., Middeldorp, A., Schneider-Kamp, P., Thiemann, R., Zankl, H.: Maximal termination. In: Voronkov, A. (ed.) RTA 2008. LNCS, vol. 5117, pp. 110–125. Springer (2008). https://doi.org/10.1007/978-3-540-70590-1_8
10. Giesl, J., Brockschmidt, M., Emmes, F., Frohn, F., Fuhs, C., Otto, C., Plücker, M., Schneider-Kamp, P., Ströder, T., Swiderski, S., Thiemann, R.: Proving termination of programs automatically with AProVE. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) IJCAR 2014. LNCS, vol. 8562, pp. 184–191. Springer (2014). https://doi.org/10.1007/978-3-319-08587-6_13
11. Giesl, J., Rubio, A., Sternagel, C., Waldmann, J., Yamada, A.: The termination and complexity competition. In: Beyer, D., Huisman, M., Kordon, F., Steffen, B. (eds.) TACAS 2019 (3). LNCS, vol. 11429, pp. 156–166. Springer (2019). https://doi.org/10.1007/978-3-030-17502-3_10
12. Giesl, J., Thiemann, R., Schneider-Kamp, P.: The dependency pair framework: Combining techniques for automated termination proofs. In: Baader, F., Voronkov, A. (eds.) LPAR 2004. LNCS, vol. 3452, pp. 301–331. Springer (2004). https://doi.org/10.1007/978-3-540-32275-7_21
13. Giesl, J., Thiemann, R., Schneider-Kamp, P., Falke, S.: Mechanizing and improving dependency pairs. J. Autom. Reason. **37**(3), 155–203 (2006). https://doi.org/10.1007/s10817-006-9057-7
14. Hirokawa, N., Middeldorp, A.: Dependency pairs revisited. In: van Oostrom, V. (ed.) RTA 2004. LNCS, vol. 3091, pp. 249–268. Springer (2004). https://doi.org/10.1007/978-3-540-25979-4_18

15. Hirokawa, N., Middeldorp, A.: Polynomial interpretations with negative coeffi-
    cients. In: Buchberger, B., Campbell, J.A. (eds.) AISC 2004. LNAI, vol. 3249, pp.
    185–198. Springer (2004). https://doi.org/10.1007/978-3-540-30210-0_16
16. Hofbauer, D., Waldmann, J.: Termination of string rewriting with matrix interpre-
    tations. In: Pfenning, F. (ed.) RTA 2006. LNCS, vol. 4098, pp. 328–342. Springer
    (2006). https://doi.org/10.1007/11805618_25
17. Jouannaud, J., Rubio, A.: The higher-order recursive path order-
    ing. In: LICS 1999. pp. 402–411. IEEE Computer Society (1999).
    https://doi.org/10.1109/LICS.1999.782635
18. Kop, C., van Raamsdonk, F.: Dynamic dependency pairs for algebraic functional
    systems. Log. Methods Comput. Sci. **8**(2) (2012). https://doi.org/10.2168/LMCS-
    8(2:10)2012
19. Koprowski, A., Waldmann, J.: Max/plus tree automata for termination of term
    rewriting. Acta Cybern. **19**(2), 357–392 (2009)
20. Korp, M., Sternagel, C., Zankl, H., Middeldorp, A.: Tyrolean Termination Tool 2.
    In: Treinen, R. (ed.) RTA 2009. LNCS, vol. 5595, pp. 295–304. Springer (2009).
    https://doi.org/10.1007/978-3-642-02348-4_21
21. Kusakari, K., Nakamura, M., Toyama, Y.: Argument filtering transformation. In:
    Nadathur, G. (ed.) PPDP 1999. LNCS, vol. 1702, pp. 47–61. Springer (1999).
    https://doi.org/10.1007/10704567_3
22. Lankford, D.: Canonical algebraic simplification in computational logic. Tech. Rep.
    ATP-25, University of Texas (1975)
23. Lucas, S.: MU-TERM: A tool for proving termination of context-sensitive rewrit-
    ing. In: van Oostrom, V. (ed.) RTA 2004. LNCS, vol. 3091, pp. 200–209. Springer
    (2004). https://doi.org/10.1007/978-3-540-25979-4_14
24. Lucas, S., Gutiérrez, R.: Automatic synthesis of logical models for order-
    sorted first-order theories. J. Autom. Reason. **60**(4), 465–501 (2018).
    https://doi.org/10.1007/s10817-017-9419-3
25. Manna, Z., Ness, S.: On the termination of Markov algorithms. In: the 3rd Hawaii
    International Conference on System Science. pp. 789–792 (1970)
26. de Moura, L.M., Bjørner, N.: Z3: an efficient SMT solver. In: Ramakrishnan, C.R.,
    Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 337–340. Springer (2008).
    https://doi.org/10.1007/978-3-540-78800-3_24
27. Sternagel, C., Thiemann, R.: The certification problem format. In: Benzmüller,
    C., Paleo, B.W. (eds.) UITP 2014. EPTCS, vol. 167, pp. 61–72 (2014).
    https://doi.org/10.4204/EPTCS.167.8
28. Sternagel, C., Thiemann, R.: Formalizing monotone algebras for certification of
    termination and complexity proofs. In: Dowek, G. (ed.) RTA-TLCA 2014. LNCS,
    vol. 8560, pp. 441–455. Springer (2014). https://doi.org/10.1007/978-3-319-08918-
    8_30
29. Stump, A., Sutcliffe, G., Tinelli, C.: StarExec: A cross-community infrastructure
    for logic solving. In: Demri, S., Kapur, D., Weidenbach, C. (eds.) IJCAR. LNCS,
    vol. 8562, pp. 367–373. Springer (2014). https://doi.org/10.1007/978-3-319-08587-
    6_28
30. Thiemann, R., Schöpf, J., Sternagel, C., Yamada, A.: Certifying the Weighted
    Path Order (Invited Talk). In: Ariola, Z.M. (ed.) FSCD 2020. LIPIcs, vol. 167, pp.
    4:1–4:20. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany
    (2020). https://doi.org/10.4230/LIPIcs.FSCD.2020.4
31. Thiemann, R., Sternagel, C.: Certification of termination proofs using CeTA. In:
    Berghofer, S., Nipkow, T., Urban, C., Wenzel, M. (eds.) TPHOLs 2009. LNCS,

vol. 5674, pp. 452–468. Springer (2009). https://doi.org/10.1007/978-3-642-03359-9_31

32. The termination problem data base, `http://termination-portal.org/wiki/TPDB`

33. Watson, T., Goguen, J., Thatcher, J., Wagner, E.: An initial algebra approach to the specification, correctness, and implementation of abstract data types. In: Current Trends in Programming Methodology. Prentice Hall (1976)

34. Yamada, A., Kusakari, K., Sakabe, T.: Nagoya Termination Tool. In: Dowek, G. (ed.) RTA-TLCA 2014. LNCS, vol. 8560, pp. 466–475. Springer (2014). https://doi.org/10.1007/978-3-319-08918-8_32

35. Yamada, A., Kusakari, K., Sakabe, T.: A unified order for termination proving. Sci. Comput. Program. **111**, 110–134 (2015). https://doi.org/10.1016/j.scico.2014.07.009

36. Zantema, H.: Termination of term rewriting: interpretation and type elimination. J. Symb. Comput. **17**(1), 23–50 (1994). https://doi.org/10.1006/jsco.1994.1003

37. Zantema, H.: The termination hierarchy for term rewriting. Appl. Algebr. Eng. Comm. Compt. **12**(1/2), 3–19 (2001). https://doi.org/10.1007/s002000100061