An Isabelle Formalization of Co-rewrite Pairs for Non-reachability in Term Rewriting

Dohan Kim
University of Innsbruck
Austria
dohan.kim@uibk.ac.at

Teppei Saito
JAIST
Japan
saito@jaist.ac.jp

René Thiemann
University of Innsbruck
Austria
rene.thiemann@uibk.ac.at

Akihisa Yamada
AIST
Japan
akihisa.yamada@aist.go.jp

Abstract

In this paper, we present an Isabelle/HOL formalization of co-rewrite pairs for non-reachability analysis in term rewriting. In particular, we formalize polynomial interpretations over negative integers as well as the weighted path order (WPO) and its variant co-WPO. With this formalization, the verified certifier CeTA is now able to check such non-reachability proofs, including those for non-reachability problems of a database where existing tools fail to provide certified proofs.

CCS Concepts: • **Theory of computation** → **Logic and verification**; **Equational logic and rewriting**.

Keywords: term rewriting, non-reachability, co-rewrite pairs

ACM Reference Format:
Dohan Kim, Teppei Saito, René Thiemann, and Akihisa Yamada. 2025. An Isabelle Formalization of Co-rewrite Pairs for Non-reachability in Term Rewriting. In *Proceedings of the 14th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP '25), January 20–21, 2025, Denver, CO, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3703595.3705889

1 Introduction

Term rewriting is one of the most common methods for equational reasoning and has a wide variety of applications, functional programming [12], termination analysis of programs [25], computer algebra [13], and automated theorem proving [9], to mention a few. In particular, conditional term rewriting [11] has been widely used, since it is more expressive than unconditional term rewriting (see [4, Section 11.3] for a related discussion). It plays an important role in, for example, algebraic specification of abstract data types [23] and integration of functional and logic programming [8].

A term rewriting system (TRS) or a conditional term rewriting system (CTRS) $\mathcal{R}$ induces a notion of evaluation in the form of a binary relation $\to_{\mathcal{R}}$ that describes a single rewrite step. Now, given $\mathcal{R}$ and two terms s and t, the *reachability problem* $s \twoheadrightarrow t$ (w.r.t. $\mathcal{R}$) asks about the existence of a substitution σ with $s\sigma \to_{\mathcal{R}}^* t\sigma$, i.e., there is a way to instantiate s and t by a substitution σ so that the instance $s\sigma$ of s rewrites to the instance $t\sigma$ of t in finitely many steps. This reachability problem is said to be $\mathcal{R}$-*satisfiable* if there is such a substitution σ. If no such substitution exists, then this problem is said to be $\mathcal{R}$-*unsatisfiable* (or $\mathcal{R}$-*infeasible* [15]).

Example 1.1. Consider the TRS representation $\mathcal{R}$ of the Ackermann function:

$$\begin{aligned} A(0,n) &\to s(n) \\ A(s(m),0) &\to A(m,s(0)) \\ A(s(m),s(n)) &\to A(m,A(s(m),n)) \end{aligned}$$

The reachability problem $A(x,s(y)) \twoheadrightarrow A(z,y)$ is $\mathcal{R}$-unsatisfiable, which means that there is no substitution σ satisfying $A(x,s(y))\sigma \to_{\mathcal{R}}^* A(z,y)\sigma$.

Non-reachability has many applications in both unconditional and conditional rewriting, such as termination analysis by the dependency pair method [3], confluence analysis of conditional rewriting [26], non-joinability analysis [2], irreducibility of rewriting systems [15], root-stability [19], and so on. Due to its importance, since 2019, there has been a category for automatic infeasibility problem solvers in the International Confluence Competition [20]. (Infeasibility problems are a variant of non-reachability problems.)

We illustrate an application of non-reachability analysis in the area of termination analysis. The termination of a TRS $\mathcal{R}$ can be characterized by dependency pairs, and one important termination technique is the analysis of the *dependency graph* [3]. The dependency graph is a directed graph over rewrite rules, where there is an edge from $s_1 \to t_1$ to $s_2 \to t_2$ iff the reachability problem $t_1 \twoheadrightarrow s_2$ is $\mathcal{R}$-satisfiable. With the help of the dependency graph, termination analysis can be done in a modular way by proving termination of each strongly connected component (SCC) in the graph separately, similar to a call-graph analysis. The problem however is that, in general, satisfiability of reachability problems is an undecidable property, and therefore the edge-relation in the dependency graph is undecidable, too. To this end, termination tools often perform over-approximations of the dependency graph by using sufficient criteria to ensure $\mathcal{R}$-unsatisfiability of the reachability problems. The better the non-reachability analysis is, the smaller will be the resulting SCCs of the graph, resulting in a simpler task for those termination methods that analyze the termination behavior of each computed SCC.

Recently, Yamada has shown that term orderings are useful for proving non-reachability in unconditional and conditional rewriting [35]. In particular, *co-rewrite pairs*, which generalize *reduction pairs* [3], play a central role. Yamada also has shown that the weighted path order (WPO) [37] and co-WPO can be used to form a co-rewrite pair, where WPO subsumes some well-known orders, such as the lexicographic path order (LPO) [10] and the Knuth–Bendix order (KBO) [13].

In this paper, we present a formalization of Yamada's non-reachability proving methods based on co-rewrite pairs. This work could have been done in several proof assistants, based on existing libraries that formalize term rewriting ([1], [6]). We have chosen to use the proof assistant Isabelle/HOL [22] and integrated to the library IsaFoR, the *Is*abelle *Fo*malization of *R*ewriting [33], since we are most comfortable with this proof assistant and this library. The main formalization results are as follows:

- the result that co-rewrite pairs provide a simple condition of non-reachability in (conditional) rewriting;
- two methods to construct co-rewrite pairs, namely co-WPO and polynomial orders over negative integers.

In the source code of IsaFoR,[1] all of the files related to this work are located in the two directories thys/Orderings and thys/Nonreachability. For the latter directory, Conditional_Nonreach_TRS.thy is the only relevant file, while for the former directory, these theory files are of interest:

CoWPO.thy	CoWPO_Impl.thy
Poly_Order_Neg.thy	Term_Order_Impl.thy
WPO_Impl.thy	

Moreover, some part of this formalization is contained in the archive of formal proofs (AFP). In particular, our extensions of WPO have been integrated into the AFP-entry on WPO [30].

In the remainder of this paper we provide hyperlinks—marked by ☑—to an HTML rendering of our formalized proofs in Isabelle/HOL.

[1]All supplementary material of this paper is available at:
`http://cl-informatik.uibk.ac.at/experiments/CPP2025/`

Reflexivity $$\frac{}{s \to_{\mathcal{R}}^* s}$$

Transitivity: $$\frac{s \to_{\mathcal{R}} t \qquad t \to_{\mathcal{R}}^* u}{s \to_{\mathcal{R}}^* u}$$

Congruence:
$$\frac{s_i \to_{\mathcal{R}} s_i'}{f(s_1, \ldots, s_i, \ldots s_n) \to_{\mathcal{R}} f(s_1, \ldots, s_i', \ldots, s_n)}$$

Replacement: $$\frac{s_1\sigma \to_{\mathcal{R}}^* t_1\sigma \ \cdots \ s_n\sigma \to_{\mathcal{R}}^* t_n\sigma}{\ell\sigma \to_{\mathcal{R}} r\sigma}$$
if $(\ell \to r \Leftarrow s_1 \twoheadrightarrow t_1, \ldots, s_n \twoheadrightarrow t_n) \in \mathcal{R}$.

Above, $\mathcal{R}$ is a conditional term rewriting system yielding the one step rewrite relation $\to_{\mathcal{R}}$ as well as many step rewriting $\to_{\mathcal{R}}^*$.

Figure 1. Inference rules for (oriented) conditional term rewriting with a CTRS $\mathcal{R}$ (cf. [16]).

2 Preliminaries

We briefly recapitulate basic definitions in term rewriting. For more detailed account, readers may refer to the book [4] by Baader and Nipkow.

A *signature* is a set $\mathcal{F}$ of function symbols, where each $f \in \mathcal{F}$ is associated with its fixed arity. The set of terms built from $\mathcal{F}$ and a denumerable set $\mathcal{V}$ of variables is denoted by $\mathcal{T}(\mathcal{F}, \mathcal{V})$. Given a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ and a *substitution* $\sigma : \mathcal{V} \to \mathcal{T}(\mathcal{F}, \mathcal{V})$, $t\sigma$ denotes the term obtained from t by replacing every variable x by $\sigma(x)$. A *context* is a term $C \in \mathcal{T}(\mathcal{F}, \mathcal{V} \cup \{\square\})$, where a special variable $\square$ occurs exactly once. Given a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$, we denote by $C[t]$ the term obtained by replacing $\square$ in C by t.

A relation R over terms is *closed under substitutions* (resp. *contexts*) if $s \; R \; t$ implies $s\sigma \; R \; t\sigma$ for any substitution σ (resp. $C[s] \; R \; C[t]$ for any context C). Relations over terms that are closed under contexts and substitutions are called *rewrite relations*. Rewrite relations which are also preorders (i.e. reflexive and transitive rewrite relations) are called *rewrite preorders*.

A *conditional rewriting rule* $\ell \to r \Leftarrow \varphi$ consists of a rule $\ell \to r$, and a list φ of conditions, which are just pairs of terms. For brevity, we write $\ell \to r$ for the conditional rewriting rule $\ell \to r \Leftarrow []$ with no condition; Similarly, we write $s_1 \twoheadrightarrow t_1, \ldots, s_n \twoheadrightarrow t_n$ for a list of conditions $[\langle s_1, t_1 \rangle, \ldots, \langle s_n, t_n \rangle]$. A *conditional term rewriting system* (CTRS) $\mathcal{R}$ is a set of conditional term rewriting rules. A CTRS $\mathcal{R}$ induces the single-step rewrite relation $\to_{\mathcal{R}}$ and the many-step rewrite relation $\to_{\mathcal{R}}^*$ on terms, defined simultaneously by the inference rules in Fig. 1. (In this paper, we are only concerned with *oriented* conditional term rewriting.) An unconditional term rewriting system (simply called TRS) is a CTRS such that all rules have no conditions.

A *reachability atom* for a CTRS $\mathcal{R}$ is a pair of terms s and t, written $s \twoheadrightarrow t$. A *reachability problem* $s_1 \twoheadrightarrow t_1, \cdots, s_n \twoheadrightarrow t_n$ is a list of reachability atoms. We say that the reachability problem is *$\mathcal{R}$-satisfiable* if there is a substitution σ such that $s_i\sigma \to^*_{\mathcal{R}} t_i\sigma$ for all $i \in \{1, \dots, n\}$; otherwise it is said to be *$\mathcal{R}$-unsatisfiable* (or non-reachable). The notion of satisfiability naturally carries over to a reachability atom by regarding it as a reachability problem consisting of single atom. Note that the replacement rule of Fig. 1 says that we can apply a rule if the reachability problem specified by the condition part is satisfiable.

A pair $(\sqsupset, \sqsupseteq)$ of relations is an *order pair* if $\sqsupset$ is irreflexive, $\sqsupset \subseteq \sqsupseteq$[2], and $\sqsupseteq \circ \sqsupset \circ \sqsupseteq \subseteq \sqsupset$. The last inclusion is called *compatibility*. A *reduction pair* [3] is an order pair $(\sqsupset, \sqsupseteq)$ on terms that satisfies the following requirements: (i) $\sqsupset$ is well-founded, (ii) $\sqsupseteq$ is a rewrite preorder, and (iii) $\sqsupset$ is closed under substitutions.

A related $\mathcal{F}$-algebra $\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}}, >, \geq)$ (cf. [35]) consists of a non-empty *carrier set* A equipped with a pair $(>, \geq)$ of relations and a collection of *interpretations* $f_{\mathcal{A}} : A^n \to A$ for each n-ary function symbol $f \in \mathcal{F}$. For brevity, we may call it *$\mathcal{F}$-algebra $\mathcal{A}$* or simply *algebra $\mathcal{A}$* if $\mathcal{F}$ is clear from context. An *assignment* $\alpha : \mathcal{V} \to A$ is a function mapping variables to elements in the carrier A.

Evaluation of terms is defined as a function $[\alpha]_{\mathcal{A}}(\cdot)$ defined as follows: $[\alpha]_{\mathcal{A}}(t) = \alpha(t)$ if t is a variable; otherwise $[\alpha]_{\mathcal{A}}(f(t_1, \dots, t_n)) = f_{\mathcal{A}}([\alpha]_{\mathcal{A}}(t_1), \dots, [\alpha]_{\mathcal{A}}(t_n))$. The algebra $\mathcal{A}$ is *strictly monotone* (resp. *weakly monotone*) if every algebra operation $f_{\mathcal{A}}$ is strictly monotone (resp. weakly monotone) in all arguments, i.e., if $f \in \mathcal{F}$ has arity n and $i \in \{1, \dots, n\}$, then $f_{\mathcal{A}}(a_1, \dots, a_i, \dots, a_n) > f_{\mathcal{A}}(a_1, \dots, b, \dots, a_n)$ (resp. $f_{\mathcal{A}}(a_1, \dots, a_i, \dots, a_n) \geq f_{\mathcal{A}}(a_1, \dots, b, \dots, a_n)$) for all $a_1, \dots, a_n, b \in A$ with $a_i > b$ (resp. $a_i \geq b$). The relation $>_{\mathcal{A}}$ on terms is defined as follows: $s >_{\mathcal{A}} t$ if $[\alpha]_{\mathcal{A}}(s) > [\alpha]_{\mathcal{A}}(t)$ for all assignments α; the relation $\geq_{\mathcal{A}}$ is defined analogously. We may also write $\mathcal{A} \models s > t$ and $\mathcal{A} \models s \geq t$ for $s >_{\mathcal{A}} t$ and $s \geq_{\mathcal{A}} t$.

A *polynomial interpretation* [14] over the natural numbers $\mathbb{N}$ is an $\mathcal{F}$-algebra $\mathcal{N}$ such that the carrier is $\mathbb{N}$ equipped with the standard orders $(>, \geq)$ and every n-ary algebra operation $f_{\mathcal{N}}$ is associated with a polynomial $P_f(X_1, \dots, X_n)$ in n indeterminates $X_1, \dots, X_n$ with integer coefficients. The interpretation of f in $\mathcal{N}$ is P_f, i.e., $f_{\mathcal{N}}(a_1, \dots, a_n) := P_f(a_1, \dots, a_n)$. Note that it must satisfy the *well-definedness* [21]:

$$f_{\mathcal{N}}(x_1, \dots, x_n) \geq 0$$

for all $x_1, \dots, x_n \in \mathbb{N}$. The orderings $>_{\mathcal{N}}$ and $\geq_{\mathcal{N}}$ induced by a polynomial interpretation $\mathcal{N}$ over the natural numbers are called the *polynomial orderings*.

Proposition 2.1. *The pair $(>_{\mathcal{N}}, \geq_{\mathcal{N}})$ is a reduction pair for every weakly monotone polynomial interpretation $\mathcal{N}$ over natural numbers.*

3 Formalization of Co-rewrite Pairs

Co-rewrite pairs are a generalization of reduction pairs for proving non-reachability. In the following definition of co-rewrite pairs,[3] it is easy to see that every reduction pair $(\sqsupset, \sqsupseteq)$ is also a co-rewrite pair.

Definition 3.1 ([35]). We call a pair $(\sqsupset, \sqsupseteq)$ of relations over terms a *co-rewrite pair*, if $\sqsupseteq$ is a rewrite preorder, $\sqsupset$ is closed under substitutions, and $\sqsupseteq \cap \sqsubset = \emptyset$, where $\sqsubset$ denotes the converse relation of $\sqsupset$.

We use Isabelle's locale mechanism [5] to formalize co-rewrite pair, where a locale is a named context for a set of fixed parameters coming with assumptions. Locales are useful for modeling a hierarchical structure of the term relations used in this paper. In the following locale for co-rewrite pair, S (the **S**trict relation) corresponds to $\sqsupset$, and NS (the **N**on-**S**trict relation) corresponds to $\sqsupseteq$ in Definition 3.1.

```
locale co_rewrite_pair = rewrite_pair S NS
  for S NS ::   "('f, 'v) term rel" +
 assumes        refl_NS: "refl NS"
  and           trans_NS: "trans NS"
  and           disj_NS_S: "NS ∩ (S⁻¹) = {}"
 begin
 ...
```

In the above, the locale rewrite_pair is formalized as just a pair of relations that satisfy closure properties:

```
locale rewrite_pair =
  fixes S NS ::   "('f, 'v) trs"
 assumes          ctxt_NS: "ctxt.closed NS"
  and             subst_S: "subst.closed S"
  and             subst_NS: "subst.closed NS"
 begin
 ...
```

Note that the above definition does not impose compatibility $\sqsupseteq \circ \sqsupset \circ \sqsupseteq \subseteq \sqsupset$ or any order assumptions, unlike the original definition of *rewrite pairs* [35]. This generic locale is used in several other locales in IsaFoR. For example,

```
locale redpair = rewrite_pair S NS + SN_ars S
   for S NS :: "('f, 'v) term rel",
locale redpair_order = redpair S NS + pre_order_pair S NS
   for S NS :: "('f, 'v) term rel", and
locale compat_redpair_order = redpair_order S NS +
     compat_pair S NS
   for S NS :: "('f,'v) trs"
```

[2]The inclusion $\sqsupset \subseteq \sqsupseteq$ is normally not required for reduction pairs, but here it is required to keep (co-)WPO well-behaved.

[3]We change the order and direction of the two relations within a co-rewrite pair in comparison to [35]. This helps to illustrate the close relationship to reduction pairs.

where the locale SN_ars S demands that S is well-founded (or strongly normalizing), and the locale compat_redpair_order formulates the conditions of reduction pair (such as compatibility) as defined in the preliminaries.

In the formalization we easily express the relationship between reduction orders and co-rewrite pairs: Simply we state the relationship "every reduction pair is a co-rewrite pair" as subsumption between the corresponding locales.

sublocale *compat_redpair_order* $\subseteq$ *co_rewrite_pair*

Reduction pairs can be used to characterize termination of TRSs, and similarly, co-rewrite pairs can be used to characterize non-reachability:

Theorem 3.2 ([35]). *Given a TRS $\mathcal{R}$, the reachability problem $s \twoheadrightarrow t$ is $\mathcal{R}$-unsatisfiable if and only if there exists a co-rewrite pair $(\sqsupset, \sqsupseteq)$ such that $\mathcal{R} \subseteq \sqsupseteq$ and $s \sqsubset t$.* ✓

The above theorem shows how co-rewrite pairs are used to prove non-reachability in rewriting systems. It says that in order to show that $s \twoheadrightarrow t$ is $\mathcal{R}$-unsatisfiable for a given TRS $\mathcal{R}$, it is sufficient to find a co-rewrite pair $(\sqsupset, \sqsupseteq)$ such that $\mathcal{R} \subseteq \sqsupseteq$ and $s \sqsubset t$. Conversely, if $s \twoheadrightarrow t$ is $\mathcal{R}$-unsatisfiable, then there exists a co-rewrite pair $(\sqsupset, \sqsupseteq)$ such that $\mathcal{R} \subseteq \sqsupseteq$ and $s \sqsubset t$. In our formal statement of Theorem 3.2, the $\mathcal{R}$-unsatisfiability of $s \twoheadrightarrow t$ is simply formalized as $\neg(\exists\sigma.\ (s\cdot\sigma, t\cdot\sigma) \in (\mathit{rstep}\ \mathcal{R})^*)$. Here, rstep $\mathcal{R}$ is the formal version of the unconditional rewrite relation $\to_{\mathcal{R}}$ within the IsaFoR-library.

Co-rewrite pairs can also serve as a sufficient condition for proving non-reachability in conditional rewriting systems.

Theorem 3.3 ([35]). *Let $\mathcal{R}$ be a CTRS. The reachability problem $s \twoheadrightarrow t$ is $\mathcal{R}$-unsatisfiable if there exists a co-rewrite pair $(\sqsupset, \sqsupseteq)$ with the following conditions:*

- *$s \sqsubset t$*
- *for each $(\ell \to r \Leftarrow \varphi) \in \mathcal{R}$, the relation $\ell \sqsupseteq r$ holds or there is a condition $u \twoheadrightarrow v \in \varphi$ with $u \sqsubset v$* ✓

Our formalization of Theorem 3.3 is described as follows:

```
context co_rewrite_pair                                    ✓
begin
...
theorem conditional_nonreach:
 assumes  "∀l r cs. ((l,r),cs) ∈ R → ((l,r) ∈ NS ∨
    (∃u v.(u,v) ∈ set cs ∧ (u,v) ∈ S⁻¹))"
  and "(s,t) ∈ S⁻¹"
 shows  "¬(∃τ.(s · τ,t · τ) ∈ (cstep R)*)"
 ...
end
```

We open an Isabelle context to obtain access to the fixed relations S and NS of the locale co_rewrite_pair, i.e., to the relations $\sqsupset$ and $\sqsupseteq$ in Theorem 3.3. Consequently, S^{-1} in the formalization corresponds to $\sqsubset$ in the theorem statement. Furthermore, unsatisfiability of $s \twoheadrightarrow t$ w.r.t. a conditional rewriting system $\mathcal{R}$ is formalized as $\neg(\exists\tau.(s\cdot\tau, t\cdot\tau) \in (\mathit{cstep}\ \mathcal{R})^*)$, where *cstep* $\mathcal{R}$ is the formal version of the conditional rewrite relation $\to_{\mathcal{R}}$ within the IsaFoR-library. The rest of the formal statement directly corresponds to the textual description of the theorem.

4 Formalization of Polynomial Orders

Recall that every reduction pair is a co-rewrite pair. Since IsaFoR already contains several formalizations of reduction pairs, we immediately get access to all of these term orders for proving non-reachability. In particular, polynomial orders over the natural numbers [14] and path orders such as KBO and LPO can immediately be used as co-rewrite pairs.

For instance, the following lemma shows that polynomial orders induced by a polynomial interpretation over natural numbers (as a reduction pair) can be used to form a co-rewrite pair. Based on the existing formalization of polynomial interpretations over natural numbers in IsaFoR including Proposition 2.1, the formalization of the following lemma is obtained without much effort.

Lemma 4.1. *For every weakly monotone polynomial interpretation $\mathcal{N}$ over the natural numbers, $(>_{\mathcal{N}}, \geq_{\mathcal{N}})$ forms a co-rewrite pair.* ✓

However, it is impossible to handle the non-reachability problem of Example 1.1 by reduction pairs via Theorem 3.2. Consequently, polynomial interpretations over the natural numbers do not apply, cf. Proposition 2.1.

Example 4.2. Applying Theorem 3.2 on Example 1.1 yields the following problem: Find a co-rewrite pair $(\sqsupset, \sqsupseteq)$ such that all the following relations are satisfied.

$$\begin{aligned} A(0,n) &\sqsupseteq s(n) \\ A(s(m),0) &\sqsupseteq A(m,s(0)) \\ A(s(m),s(n)) &\sqsupseteq A(m,A(s(m),n)) \\ A(z,y) &\sqsupset A(x,s(y)) \end{aligned}$$

Now assume that there is a reduction pair that satisfies the constraints. Applying closure under substitutions to the last constraint repeatedly, we get the infinite sequence

$$A(z,y) \sqsupset A(z,s(y)) \sqsupset A(z,s(s(y))) \sqsupset \cdots.$$

This contradicts the well-foundedness property of a reduction pair.

To alleviate this problem, we formalize co-rewrite pairs that are not reduction pairs, namely co-rewrite pairs generated from polynomial interpretations over the negative integers $\mathbb{Z}_{\leq 0}$ (with zero) [35, Example 2]. Polynomial interpretations $\mathcal{Z}$ over $\mathbb{Z}_{\leq 0}$ are similar to those over natural numbers [14], i.e., they are algebras where each n-ary function symbol f is associated with a polynomial $P_f(X_1, \ldots, X_n)$ with n indeterminates $X_1, \ldots, X_n$. These polynomials must be well-defined ($P_f(x_1, \ldots, x_n) \in \mathbb{Z}_{\leq 0}$ for all $x_1, \ldots, x_n \in \mathbb{Z}_{\leq 0}$) and weakly monotone to form a co-rewrite pair. Importantly, well-foundedness is not necessary to be a co-rewrite pair.

Theorem 4.3. *The pair $(>_{\mathcal{Z}}, \geq_{\mathcal{Z}})$ forms a co-rewrite pair for every weakly monotone polynomial interpretation $\mathcal{Z}$ over $\mathbb{Z}_{\leq 0}$.* ☑

The following corollary is immediate from Theorem 3.3.

Corollary 4.4. *Given a polynomial interpretation $\mathcal{Z}$ over $\mathbb{Z}_{\leq 0}$, if $(\ell \to r \Leftarrow \varphi) \in \mathcal{R}$ implies $\ell \geq_{\mathcal{Z}} r$ or $u <_{\mathcal{Z}} v$ for some $u \twoheadrightarrow v \in \varphi$, and $s <_{\mathcal{Z}} t$, then $s \twoheadrightarrow t$ is $\mathcal{R}$-unsatisfiable.* ☑

Just with the change of the carrier, the leading example can easily be solved.

Example 4.5. Consider the non-reachability problem of Example 1.1. The following polynomial interpretations $\mathcal{Z}$ over $\mathbb{Z}_{\leq 0}$

$$A_{\mathcal{Z}}(x,y) = y - 1$$
$$s_{\mathcal{Z}}(x) = x - 1$$
$$0_{\mathcal{Z}} = 0$$

satisfies $\mathcal{R} \subseteq \geq_{\mathcal{Z}}$ and $A(z,y) >_{\mathcal{Z}} A(x, s(y))$. In more detail, the four term constraints result in the following four trivial inequalities on polynomials.

$$n - 1 \geq n - 1$$
$$0 - 1 \geq 0 - 1 - 1$$
$$n - 1 - 1 \geq n - 1 - 1$$
$$y - 1 > y - 1 - 1$$

Therefore the non-reachability is concluded by Corollary 4.4.

Example 4.6. Consider the CTRS $\mathcal{R}$ that consists of the following rules.

$$add(0,x) \to x$$
$$add(s(x),y) \to s(add(x,y))$$
$$add(x,0) \to x \Leftarrow add(x,s(y)) \twoheadrightarrow s(add(x,y))$$

To prove that $add(x,s(0)) \twoheadrightarrow x$ is unsatisfiable by Corollary 4.4, we use the following polynomial interpretation $(\mathcal{Z}, \geq)$ over $\mathbb{Z}_{\leq 0}$:

$$add_{\mathcal{Z}}(x,y) = x + y$$
$$s_{\mathcal{Z}}(x) = x - 1$$
$$0_{\mathcal{Z}} = 0$$

For this interpretation, the constraints $\ell \geq_{\mathcal{Z}} r$ for the three rules $\ell \to r \Leftarrow \varphi$ turn into obviously valid polynomial inequalities.

$$0 + x \geq x$$
$$x - 1 + y \geq (x + y) - 1$$
$$x + 0 \geq x$$

The constraint for $add(x,s(0)) \twoheadrightarrow x$ again turns out to be the trivial polynomial inequality $x + (0-1) < x$. So we conclude unsatisfiability.

The remaining part of this section is dedicated for illustrating our formalization. The main locale is as follows:

```
locale poly_order_neg = order_pair_neg +                         ☑
  fixes   I ::"('f, 'a) poly_inter"
   and    ...
  assumes neg_I ::"⋀ fn. fn ∈ F ⟹ zero_poly ≥pn I fn"
   and    mono_I ::"⋀ fn. poly_weak_neg_mono_all (I fn)"
   ...
begin
...
```

In the locale, F is the signature, and its elements fn are pairs of function symbols (of a type variable $'f$) and natural numbers (i.e., arities). The symbol I corresponds to a polynomial interpretation, where its type is defined as follows:

```
type_synonym ('f, 'a) poly_inter =
  "'f × nat ⇒ (nat, 'a) poly"
```

The type "$(nat, 'a)$ $poly$" in the above relies on the following existing type definitions (see [27]):

```
typedef (overloaded) 'v monom = ...
type_synonym ('v, 'a) poly ="('v monom × 'a) list"
```

Roughly speaking, the type "$('v, 'a)$ $poly$" is simply the type "$('v$ $monom \times 'a)$ $list$" because polynomials can be represented by a sum of monomials (with the type "$'v$ $monom$") multiplied by some coefficient (with the type "$'a$").

In the above locale poly_order_neg, the condition

$$\bigwedge fn.\ fn \in F \Longrightarrow zero\_poly \geq_{pn} I\ fn$$

corresponds to *well-definedness*, while

$$\bigwedge fn.\ poly\_weak\_neg\_mono\_all\ (I\ fn)$$

corresponds to *weak monotonicity* for polynomial interpretations over $\mathbb{Z}_{\leq 0}$.

Note that one could in principle also take $\mathbb{Z}$ as carrier, but it is known that for non-reachability proving it is not as good as using $\mathbb{Z}_{\leq 0}$ [35, Section 7], so we have decided to support $\mathbb{Z}_{\leq 0}$ first. In [35, Section 7] it is also remarked that having an edge point (namely 0) in the carrier is often beneficial. We in addition note that both on $\mathbb{N}$ and on $\mathbb{Z}_{\leq 0}$ one can successfully compare linear polynomials with different coefficients, but this is not possible on $\mathbb{Z}$. For instance, $2x \geq x$ on $\mathbb{N}$, $x \geq 2x$ on $\mathbb{Z}_{\leq 0}$, but $ax \geq bx$ is invalid on $\mathbb{Z}$ for all integers $a \neq b$.

5 Formalization of Weighted Path Order and Co-weighted Path Order

The *weighted path order* (WPO) [37] provides a way to extend interpretation-based orders such as polynomial orders. In this section, we formalize the results that WPO and its variant co-WPO form co-rewrite pairs and therefore can be used to prove non-reachability.

5.1 Weighted Path Order

The *weighted path order* unifies and extends several well-known term orders used in rewriting, such as the lexicographic path order (LPO) [10] and the Knuth-Bendix order (KBO) [13]. WPO has already been formalized in Isabelle/HOL [32], and we briefly repeat its definition and required notions.

A *partial status* π is a mapping that assigns to each n-ary symbol f a list $[\bar{i}_m] = [i_1, \dots, i_m]$ of distinct positions from $\{1, \dots, n\}$. We also view $\pi(f)$ as the set $\{\bar{i}_m\}$ for $\pi(f) = [\bar{i}_m]$. A related $\mathcal{F}$-algebra $\mathcal{A} = (A, \{f_{\mathcal{A}}\}_{f \in \mathcal{F}}, >, \geq)$ is weakly π-*simple* (or just π-simple) for a partial status π if $f_{\mathcal{A}}(a_1, \dots, a_n) \geq a_i$ for an arbitrary n-ary $f \in \mathcal{F}$, $a_1, \dots, a_n \in A$, and $i \in \pi(f)$.

Let π be a partial status and $\mathcal{A}$ be an $\mathcal{F}$-algebra. Let $(>, \geq)$ and $(\succ, \succeq)$ be pairs of relations on $\mathcal{A}$ and $\mathcal{F}$, respectively. The latter $(\succ, \succeq)$ is called *precedence*. The *weighted path order* $\mathrm{WPO}(\pi, \mathcal{A}, >, \geq, \succ, \succeq)$ is a pair $(\sqsupset_{\mathrm{WPO}}, \sqsupseteq_{\mathrm{WPO}})$ of relations on terms defined simultaneously as follows: $s \sqsupset_{\mathrm{WPO}} t$ iff

1. $\mathcal{A} \models s > t$ or
2. $\mathcal{A} \models s \geq t$ and
   a. $s = f(s_1, \dots, s_n)$ and $s_i \sqsupseteq_{\mathrm{WPO}} t$ for some $i \in \pi(f)$, or
   b. $s = f(s_1, \dots, s_n)$, $t = g(t_1, \dots, t_m)$, $s \sqsupset_{\mathrm{WPO}} t_j$ for every $j \in \pi(g)$, and
      i. $f \succ g$ or
      ii. $f \succeq g$ and $\pi_f(s_1, \dots, s_n) \sqsupset^{\mathrm{lex}}_{\mathrm{WPO}} \pi_g(t_1, \dots, t_m)$.

The relation $\sqsupseteq_{\mathrm{WPO}}$ is defined similarly by using $\sqsupseteq^{\mathrm{lex}}_{\mathrm{WPO}}$ instead of $\sqsupset^{\mathrm{lex}}_{\mathrm{WPO}}$ in (2b-ii) and adding $s = t \in \mathcal{V}$ as a new alternative case (2c). Here $\sqsupset^{\mathrm{lex}}_{\mathrm{WPO}}$ and $\sqsupseteq^{\mathrm{lex}}_{\mathrm{WPO}}$ compare lists of terms lexicographically using $\sqsupset_{\mathrm{WPO}}$ and $\sqsupseteq_{\mathrm{WPO}}$, respectively.

Theorem 5.1 ([35])**.** *Let π be a partial status and $\mathcal{A}$ be an $\mathcal{F}$-algebra. Suppose that $\mathcal{A}$ and $\mathcal{F}$ are equipped with order pairs $(>, \geq)$ and $(\succ, \succeq)$, respectively. If $\mathcal{A}$ is weakly monotone and weakly π-simple, then $(\sqsupset_{WPO}, \sqsupseteq_{WPO})$ forms a co-rewrite pair.* ☑

In the remainder of this paper, if a partial status π is not specifically given, then we assume that it is the total status.

Example 5.2. Let us consider a variant of the reachability problem of Example 1.1 where the reachability atom is replaced by $A(x, s(y)) \twoheadrightarrow A(s(x), y)$. Now we interpret all function symbols as max functions over natural numbers. The WPO induced by these interpretations and a precedence with $A \succ s$ satisfies $\mathcal{R} \subseteq \sqsupseteq_{\mathrm{WPO}}$ and $A(x, s(y)) \sqsubset_{\mathrm{WPO}} A(s(x), y)$, therefore the non-reachability is concluded by Theorem 3.2 and Theorem 5.1.

The formalization of Theorem 5.1 turned out to be more effort than expected. The problem was the previous design of IsaFoR regarding term relations. Since IsaFoR was initially designed for checking termination proofs, all strict term relations $\sqsupset$ always imposed well-foundedness as an assumption: without well-foundedness the relations could not be used for termination proving. This global assumption on well-foundedness also was imposed on algebras, etc. Hence, the existing interface of term relations had the shape of requiring several common properties unconditionally, e.g., well-foundedness, closure under substitutions and some other common properties. In addition, there have been a few flags that indicate whether further optional properties are satisfied, e.g., whether $\sqsupset$ is monotone, whether $\sqsupseteq$ has the subterm property, etc.

Thus, with this interface there was no way to formalize Theorem 5.1 in its full generality, because there was an implicit restriction on well-founded algebras in the interface, and this restriction is not present in Theorem 5.1. As an example, it was possible in IsaFoR to obtain Theorem 5.1 with polynomial interpretations over $\mathbb{N}$ as an algebra, but it was not possible to use the theorem with polynomial interpretations over $\mathbb{Z}_{\leq 0}$ from the previous section.

In order to solve this problem, we did a major refactoring of IsaFoR's interface for term relations, so that strong normalization is no longer an unconditional prerequisite. In fact, we now integrated a detailed query interface about term relations. For every class of orders one can query 11 different properties about the term relations. Example properties are well-foundedness, reflexivity, closure of $\sqsupset$ under substitutions, closure of $\sqsupset$ under contexts, compatibility of $\sqsupseteq$ with some argument filter (a variant of a partial status), etc., cf. the definitions of `rel_impl` and `rel_impl_prop` in theory `Term_Order_Impl`. ☑

On the one hand, this extension of the interface enabled us to formalize Theorem 5.1 in its general form. On the other hand, we had to adjust all the existing classes of orders in IsaFoR to adhere to the new interface. For most of these orders it was not a big effort, as we just had to set the correct flags for each of the orderings. However, for all term relations that are parameterized by some existing term order, we needed to invest more effort and sometimes formalize new proofs.

Let us briefly illustrate the required changes w.r.t. WPO. The previous formalization of WPO just assumed that the algebra gives rise to a reduction pair, and then it was shown that the resulting WPO is again a reduction pair. The new formalization is more fine-grained. In particular it has now been formalized that the strict WPO-relation is irreflexive, without the assumption on well-foundedness of the algebra, cf. `lemma wpo_irrefl` ☑ in `locale wpo_with_assms` ☑, where the latter locale does not enforce strong normalization of the strict relation. This is a completely novel formalization that was not present earlier. We also refactored some other proofs about properties of WPO that previously referred to well-foundedness and now use a different way of reasoning. Note that these changes have been performed in the AFP-entry on WPO [30].

5.2 Co-weighted Path Order

It is known that one can refine WPO for non-reachability, resulting in the relation co-WPO [35]. Co-WPO is defined as WPO induced by the negated relations, i.e., WPO$(\pi, \mathcal{A}, \nleq, \nless, \npreceq, \nprec)$. More specifically, the pair $(\sqsupset_{\overline{\mathrm{WPO}}}, \sqsupseteq_{\overline{\mathrm{WPO}}})$ of relations on terms is defined as follows: $s \sqsupset_{\overline{\mathrm{WPO}}} t$ iff

1. $\mathcal{A} \models s \nleq t$ or
2. $\mathcal{A} \models s \nless t$ and
   a. $s = f(s_1, \ldots, s_n)$ and $s_i \sqsupseteq_{\overline{\mathrm{WPO}}} t$ for some $i \in \pi(f)$, or
   b. $s = f(s_1, \ldots, s_n)$, $t = g(t_1, \ldots, t_m)$, $s \sqsupset_{\overline{\mathrm{WPO}}} t_j$ for every $j \in \pi(g)$, and
      i. $f \npreceq g$ or
      ii. $f \nprec g$ and $\pi_f(s_1, \ldots, s_n) \sqsupset^{\mathrm{lex}}_{\overline{\mathrm{WPO}}} \pi_g(t_1, \ldots, t_m)$.

Also, $\sqsupseteq_{\overline{\mathrm{WPO}}}$ is defined similarly by using $\sqsupseteq^{\mathrm{lex}}_{\overline{\mathrm{WPO}}}$ instead of $\sqsupset^{\mathrm{lex}}_{\overline{\mathrm{WPO}}}$ in (2b-ii) and adding c. $s = t \in \mathcal{V}$ as the third alternative in case 2.

The idea of co-WPO is to use the relation $\sqsupset_{\overline{\mathrm{WPO}}}$ obtained from WPO$(\pi, \mathcal{A}, \nleq, \nless, \npreceq, \nprec)$ with the preorder $\sqsupseteq_{\mathrm{WPO}}$ from WPO$(\pi, \mathcal{A}, >, \geq, \succ, \succeq)$. Note that if the underlying orders are total, $\sqsupset_{\overline{\mathrm{WPO}}}$ is identical to $\sqsupset_{\mathrm{WPO}}$.

Proposition 5.3. *Let π be a partial status and $\mathcal{A}$ be an $\mathcal{F}$-algebra. Suppose that $\mathcal{A}$ and $\mathcal{F}$ are equipped with order pairs $(>, \geq)$ and $(\succ, \succeq)$, respectively. If $\mathcal{A}$ is a π-simple and weakly monotone algebra, then $(\sqsupset_{\overline{\mathrm{WPO}}}, \sqsupseteq_{WPO})$ is a co-rewrite pair.*

Example 5.4 ([35, Example 7])**.** Consider the CTRS $\mathcal{R} = \{a \to b \Leftarrow b \twoheadrightarrow a\}$ and the reachability problem $a \twoheadrightarrow b$, which is unsatisfiable. In order to prove the latter, we use an algebra $\mathcal{A}$ that evaluates a, b to two unrelated elements and a non-total precedence such that $a \npreceq b$ and $b \npreceq a$. For such an algebra and precedence we obtain both $b \sqsubset_{\overline{\mathrm{WPO}}} a$ for the rule of $\mathcal{R}$ and $a \sqsubset_{\overline{\mathrm{WPO}}} b$ for the reachability atom. Therefore, we can conclude the unsatisfiability from Theorem 3.3 and Proposition 5.3.

Our formalization of co-WPO is based on the generalized formalization of WPO as it was discussed in Section 5.1. Note that locale cowpo just fixes the different parameters that are occurring in Proposition 5.3. The later locales describe co-WPO and co-WPO with the required assumptions, respectively.

```
locale cowpo =
  fixes  n  ::nat
   and   π  ::"'f status"
   and   c  ::"'f × nat ⇒  order_tag"
   and   nleA nltA  ::"('f, 'v) term rel"
   and   S NS  ::"('f, 'v) term rel"
   and   nle_prc nlt_prc::"'f × nat ⇒' f × nat ⇒  bool"
   and   gt_prc ge_prc  ::"'f × nat ⇒' f × nat ⇒  bool"
begin
sublocale essential :
  wpo n S NS "(λ f g. (gt_prc f g, ge_prc f g))" ··· .
sublocale co :
  wpo n nleA nltA "(λ f g. (nle_prc f g, nlt_prc f g))" ··· .
...
```

Above, in order to construct co-rewrite pairs using WPO and co-WPO, we add the sublocale *essential* for specifying a formal version of $\sqsupseteq_{\mathrm{WPO}}$ via the wpo locale. Here, the strict and non-strict relation of the algebra are provided as parameters S and NS, so (S, NS) in Isabelle corresponds to the pair of relations $(>, \geq)$ in the algebra. Similarly, the expression "$(\lambda f g.\ (gt\_prc\ f\ g,\ ge\_prc\ f\ g))$" in the formalization encodes the precedence $(\succ, \succeq)$. In the same way, we add the sublocale *co* in order to specify a formal version of the co-WPO relation $\sqsupset_{\overline{\mathrm{WPO}}}$. Here, we use the relations containing the letter *n* for *negation*, e.g., *nltA* abbreviates "not-less-than in the algebra," i.e., the relation $\nless$.

In the next locale, cowpo_with_assms, we enforce various properties on the parameters that have been specified in the locale wpo.

```
locale cowpo_with_assms = order_pair' + cowpo + ··· +
 assumes  subst_S :   "(s,t) ∈ S ⟹ (s · σ,t · σ) ∈ S"
   and    subst_NS :  "(s,t) ∈ NS ⟹ (s · σ,t · σ) ∈ NS"
   and    nleA :  "nleA = {(s,t). ∀σ. (t · σ,s · σ) ∉ NS}"
   and    nltA :  "nltA = {(s,t). ∀σ. (t · σ,s · σ) ∉ S}"
   and    nle_prc_comp :  "nle_prc f g  = (¬ ge_prc g f)"
   and    nlt_prc_comp :  "nlt_prc f g  = (¬ gt_prc g f)"
begin
 ...
```

In this locale the standard properties of the relations from the algebras are enforced, e.g., closure under substitution. Additionally, it is demanded that the parameters of the four negated relations *nleA*, *nltA*, *nle_prc*, and *nlt_prc* are indeed defined as negated relations. With these assumptions in place, we formalize the existing paper proofs. Here, we reuse the available results about WPO whenever possible, and also establish the required connection between $\sqsupset_{\overline{\mathrm{WPO}}}$ and $\sqsupseteq_{\mathrm{WPO}}$ within Isabelle. Eventually we arrive at the formal version of Proposition 5.3.

Let us now consider a potential extension of (co-)WPO. The existing WPO formalization [32] (for termination) allows *multiset* comparisons instead of lexicographic comparisons, as it is possible in recursive path orders. Namely, for every function symbol, one can select between comparing the arguments lexicographically or via multiset comparison.

Definition 5.5. [31, Definition 2.2] Let $\sqsupset$ and $\sqsupseteq$ be relations on a set A. For multisets X, Y consisting elements of A, we write $X \sqsupseteq^{\mathrm{mul}} Y$ if there are some (disjoint) partitions $X = \{x_1, \ldots, x_n\} \uplus X'$ and $Y = \{y_1, \ldots, y_n\} \uplus Y'$ such that $x_i \sqsupseteq y_i$ for all $0 \leq i \leq n$ and moreover for all $y \in Y'$ there is some $x \in X'$ such that $x \sqsupset y$. We also write $X \sqsupset^{\mathrm{mul}} Y$ if in addition X' is not empty.

Of course, we also tried to integrate multiset comparisons into co-WPO, to obtain a more powerful order for non-reachability. However, during the formalization it turned out that it is unsound to allow multiset comparisons in co-WPO: we got stuck in proving certain cases of this modified co-WPO, and these cases led us to a counter-example.

Example 5.6. Assume that co-WPO uses multiset comparisons instead of lexicographic comparisons. Consider $\mathcal{F} = \{f, a, b, c\}$, where $\mathcal{A}$ is a trivial algebra such that $>_{\mathcal{A}} = \emptyset$ and $\geq_{\mathcal{A}} = \mathcal{T}(\mathcal{F},\mathcal{V}) \times \mathcal{T}(\mathcal{F},\mathcal{V})$. The precedence is defined by $a \succ b$ and $\succeq$ is the reflexive closure of $\succ$. Let $s = f(a,c)$ and $t = f(b,c)$. We first conclude $s \sqsupseteq_{\mathrm{WPO}} t$, since $\{a,c\} \sqsupseteq_{\mathrm{WPO}}^{\mathrm{mul}} \{b,c\}$ as $a \sqsupset_{\mathrm{WPO}} b$. But we further conclude $c \sqsupset_{\overline{\mathrm{WPO}}} a$ (since $c \not\lesssim a$) and $b \sqsupset_{\overline{\mathrm{WPO}}} c$ (since $b \not\lesssim c$) and therefore, $\{b,c\} = \{c,b\} \sqsupset_{\overline{\mathrm{WPO}}}^{\mathrm{mul}} \{a,c\}$. This finally shows $t = f(b,c) \sqsupset_{\overline{\mathrm{WPO}}} f(a,c) = s$ and hence $s \sqsubset_{\overline{\mathrm{WPO}}} t$, i.e., the modified version of $(\sqsupset_{\overline{\mathrm{WPO}}}, \sqsupseteq_{\mathrm{WPO}})$ with multiset comparisons is not a co-rewrite pair, since it violates the condition $\sqsupseteq_{\mathrm{WPO}} \cap \sqsubset_{\overline{\mathrm{WPO}}} = \emptyset$.

The problem in the example is that the relevant property of *co-compatibility* is not preserved by a multiset comparison, in contrast to a lexicographic comparison [35, Lemma 5]. (Two pairs of relations, say $(\sqsupset_P, \sqsupseteq_P)$ and $(\sqsupset_Q, \sqsupseteq_Q)$, are co-compatible if $\sqsupseteq_P \cap \sqsubset_Q = \sqsupseteq_Q \cap \sqsubset_P = \emptyset$.) The upcoming example shows that co-compatibility is lost, even if $\sqsupset_P$ and $\sqsupset_Q$ are transitive.

Example 5.7. Let $D = \{a,b,c,d\}$. Define the relations $\sqsupset_P$ and $\sqsupset_Q$ on D as follows: $a \sqsupset_P c$, $b \sqsupset_P d$, $c \sqsupset_Q b$, and $d \sqsupset_Q a$. Let $\sqsupseteq_P$ and $\sqsupseteq_Q$ be the reflexive closures of $\sqsupset_P$ and $\sqsupset_Q$, respectively. We can confirm that $(\sqsupset_P, \sqsupseteq_P)$ and $(\sqsupset_Q, \sqsupseteq_Q)$ are co-compatible. On the other hand, $\{a,b\} \sqsupseteq_P^{\mathrm{mul}} \{c,d\}$ but also $\{a,b\} \sqsubset_Q^{\mathrm{mul}} \{c,d\}$. So $(\sqsupset_P^{\mathrm{mul}}, \sqsupseteq_P^{\mathrm{mul}})$ and $(\sqsupset_Q^{\mathrm{mul}}, \sqsupseteq_Q^{\mathrm{mul}})$ are not co-compatible.

6 Certification

After having formalized the theory of co-rewrite pairs, the next step is to utilize this formalization to develop a verified checker that can certify untrusted proofs of non-reachability. To this end, we extend CeTA [33], a checker that is able to certify various kinds of proofs specified in the certification problem format (CPF) [28]. (CeTA is a verified checker, in the sense that the code is automatically generated from the formalization IsaFoR.) Prior to our presented developments, CeTA and CPF cover termination proofs and confluence proofs for TRSs. Also non-reachability proofs are already supported, but not those based on co-rewrite pairs.

For polynomial interpretations over $\mathbb{N}$, nearly all the work has actually been done previously: we just need to utilize the lemma that every reduction pair is a co-rewrite pair and then rely upon the existing formalization of reduction pairs in CeTA. Here, under a given interpretation $\mathcal{N}$, order constraints $\ell \sqsupseteq_{\mathcal{N}} r$ are converted into polynomial constraints $[\alpha]_{\mathcal{N}}(\ell) \geq [\alpha]_{\mathcal{N}}(r)$ and are then passed to the polynomial library of CeTA [29]. That library implements "check"-functions, i.e., sufficient criteria to validate polynomial inequalities over $\mathbb{N}$.

In order to add support for polynomial interpretations on $\mathbb{Z}_{\leq 0}$, we convert the arising polynomial inequalities over $\mathbb{Z}_{\leq 0}$ into inequalities where the variables range over $\mathbb{N}$. For instance, the following Isabelle code implements a "check"-function to ensure whether a polynomial is greater or equal to another, w.r.t. a polynomial interpretation on $\mathbb{Z}_{\leq 0}$.

```
definition check_poly_neg_ge where
  "check_poly_neg_ge p q = (let p' = poly_convert p;
    q' = poly_convert q in check_poly_ge p' q')"
```

Above, check_poly_ge is the existing "check"-function for polynomial inequalities on $\mathbb{N}$, and poly_convert is simply defined for converting a polynomial p into another polynomial p' using the substitution $v \mapsto -v$ for each variable v in p.

```
definition poly_convert where
  "poly_convert p = poly_subst (λv.[(var_monom v, −1)]) p"
```

In essence, in order to ensure $p(x_1,\ldots,x_n) \geq q(x_1,\ldots,x_n)$ for all $x_1,\ldots,x_n \in \mathbb{Z}_{\leq 0}$, we test whether $p(-x_1,\ldots,-x_n) \geq q(-x_1,\ldots,-x_n)$ for all $x_1,\ldots,x_n \in \mathbb{N}$ by the existing polynomial library on $\mathbb{N}$.

Similar to polynomial interpretations, we also reuse the existing "check"-function for WPO in CeTA [32]. However, instead of using the naive recursive implementation of WPO from the original formalization, we avoid the exponential runtime by using a more efficient memoized implementation of WPO [34].

Concerning co-WPO, we can nearly fully reuse the existing WPO implementation, as co-WPO is just an instance of WPO with other parameters. However, here a complication arises as there are *negative* occurrences of relations in the definition of co-WPO. For instance, the first rule of co-WPO requires $\mathcal{A} \models s \not\lesssim t$. Since the weakly monotone algebras in CeTA do not offer such negated checks, we instead use the positive inverted relation ($\mathcal{A} \models s > t$). This switch is an equivalence transformation for polynomial interpretations over $\mathbb{N}$ and $\mathbb{Z}_{\leq 0}$, but it is just an under-approximation in the general case, since the order in the weakly monotone algebra is not necessarily total.

In order to test our extensions of CeTA, we have implemented a prototype non-reachability tool that can generate certificates in CPF and conducted experiment using non-reachability problems from the ARI-COPS database.[4] (In the database, the problems are called infeasibility problems.) In the current version of CeTA, the following proposition (a variant of Theorem 3.2) is the only way to use co-rewrite pairs for non-reachability:

Proposition 6.1. *Let $\mathcal{R}$ be a CTRS and $s_1 \twoheadrightarrow t_1, \cdots, s_n \twoheadrightarrow t_n$ a reachability problem. The problem is unsatisfiable if there is a co-rewrite pair $(\sqsupset, \sqsupseteq)$ such that $\mathcal{R}' \subseteq \sqsupseteq$ and $s_i \sqsubset t_i$ for some $i \in \{1,\ldots,n\}$, where $\mathcal{R}'$ is the TRS obtained from $\mathcal{R}$ by discarding all condition parts of the rules, namely*

$$\{\ell \to r \mid \ell \to r \Leftarrow \varphi \in \mathcal{R}\}.$$

[4] https://ari-cops.uibk.ac.at

Table 1. Comparison using the 146 non-reachability problems in ARI-COPS with 60 seconds time limit.

	p-nat	p-neg	wpol	nonreach	Moca
proved	21	24	22	43	49
certified	21	24	22	31	32

We illustrate the overall process from proof search to certification. Given a non-reachability problem, the prototype tool reduces it to a reachability problem by Proposition 6.1. If it succeeds to find a witness of non-reachability (namely a co-rewrite pair), CeTA confirms if the witness for Proposition 6.1 indeed gives a non-reachability proof to the original problem.

In the experiment,[5] our prototype tool uses Proposition 6.1 with three kinds of co-rewrite pairs (as indicated in Table 1), namely:

1. p-nat: polynomial orders on $\mathbb{N}$ (Lemma 4.1)
2. p-neg: polynomial orders on $\mathbb{Z}_{\leq 0}$ (Theorem 4.3)
3. wpol: WPO with polynomial on $\mathbb{N}$ as algebras (Theorem 5.1)

In all these co-rewrite pairs, only polynomials of the form $f_{\mathcal{A}}(x_1, \dots, x_n) = f_0 + f_1 x_1 + \cdots + f_n x_n$ with $f_1, \dots, f_n \in \{0, 1\}$ are considered. In addition, for WPO, only strict total precedences are considered, i.e., there are no different function symbols f, g with $f \succeq g$ and $g \succeq f$. Under these restrictions, for all the three methods, it is decidable whether, given (finite) order constraints, there is a co-rewrite pair that satisfies the constraints. The point is the restriction $f_1, \dots, f_n \in \{0, 1\}$ on coefficients of polynomials. Thanks to this, we can encode constraints into linear arithmetic with if-then-else expressions and use SMT solvers such as Z3 [7] that have decision procedures for this.

In Table 1, the experimental data shows that, within the time limit of 60 seconds, the methods 1, 2 and 3 of our prototype tool (in combination with Z3) gives certified proofs to 21, 24 and 22 problems, respectively, among all of the 146 non-reachability problems in ARI-COPS. There already exist other non-reachability tools (nonreach [17, 18] and Moca [24]) that can generate certified proofs for non-reachability in CPF. On the same problem set and within the same time limit, nonreach reports non-reachability of 43 problems, and among them 31 are certified by CeTA, while the remaining 12 proofs are rejected; Moca finds 49 problems infeasible and succeeds to synthesize valid certificates for 32 of them. Their methods are based on syntactical analysis (such as term-cap and equational completion), and these are complementary to our semantical ordering-based approach. In fact, our tool has found first certified proofs ever to eight problems in ARI-COPS, to which neither nonreach nor Moca are able to give certified proofs. Below is an example from the eight problems where the ordering approach shines.

[5]See http://cl-informatik.uibk.ac.at/experiments/CPP2025/exp/ for details.

Example 6.2. Consider the following TRS $\mathcal{R}$ of the Ackermann function, together with the predicate $\mathit{isNat}(t)$ that decides whether an input term t is a natural number:

$$\begin{aligned} A(0, n) &\to s(n) \\ A(s(m), 0) &\to A(m, s(0)) \\ A(s(m), s(n)) &\to A(m, A(s(m), n)) \\ \mathit{isNat}(0) &\to \mathit{true} \\ \mathit{isNat}(s(n)) &\to \mathit{isNat}(n) \\ \mathit{isNat}(\mathit{true}) &\to \mathit{false} \\ \mathit{isNat}(\mathit{false}) &\to \mathit{false} \end{aligned}$$

The problem ARI-COPS#1613 asks to prove that the atom

$$\mathit{isNat}(A(s^7(0), s^7(0))) \twoheadrightarrow \mathit{false}$$

is unsatisfiable, where $s^7(0)$ denotes seven times application of s to 0, i.e., $s(s(s(s(s(s(s(0)))))))$. A naive syntactical approach is to rewrite $A(s^7(0), s^7(0))$ to the normal form, which cannot be done in a reasonable amount of time since the Ackermann function grows very rapidly. Actually, because Moca is based on equational completion, it indeed tries to normalize the term and exceeds the time limit of the experiment. Unfortunately, nonreach also gives up on this problem. On the other hand, our prototype tool finds the following simple proof of non-reachability by the polynomial interpretation $\mathcal{N}$ over natural numbers:

$$\begin{aligned} 0_{\mathcal{N}} &= 0 \\ s_{\mathcal{N}}(x) &= x \\ \mathit{true}_{\mathcal{N}} &= 1 \\ \mathit{false}_{\mathcal{N}} &= 2 \\ A_{\mathcal{N}}(x, y) &= x \\ \mathit{isNat}_{\mathcal{N}}(x) &= x + 1 \end{aligned}$$

It is easy to confirm $\mathcal{R} \subseteq \geqslant_{\mathcal{N}}$ and also

$$\mathit{false} >_{\mathcal{N}} \mathit{isNat}(A(s^7(0), s^7(0)))$$

so we can conclude the non-reachability.

7 Summary and Future Work

In this paper, we have formalized co-WPO and polynomials on the domain $\mathbb{Z}_{\leq 0}$ for non-reachability as a part of IsaFoR. As a by-product, CeTA is now able to certify non-reachability proofs by co-rewrite pairs. We conclude the paper with discussion about future work.

One of the possibilities to strengthen the support for certified non-reachability proofs lies in the addition of further classes of co-rewrite pairs. Here, in particular *tuple-interpretations* [36] are of interest: these co-rewrite pairs are already implemented in automated non-reachability tools, but are not yet supported by any certifier for non-reachability.

Finally, one might also consider to use IsaFoR's new extensive interface for term orderings for related certification tasks. For instance, *discrimination pairs* [2] are another kind of generalization of reduction pairs, and they can be used for certifying non-confluence proofs.

Acknowledgements

We thank Nao Hirokawa for valuable discussion. This research was supported by JST SPRING Grant Number JPMJSP2102, JSPS-FWF Grant Number JPJSBP120222001, and the Austrian Science Fund (FWF) project I 5943.

References

[1] Ariane Alves Almeida and Mauricio Ayala-Rincón. 2020. Formalizing the dependency pair criterion for innermost termination. *Sci. Comput. Program.* 195 (2020), 102474. https://doi.org/10.1016/J.SCICO.2020.102474

[2] Takahito Aoto. 2013. Disproving Confluence of Term Rewriting Systems by Interpretation and Ordering. In *Frontiers of Combining Systems - 9th International Symposium, FroCoS 2013, Nancy, France, September 18-20, 2013. Proceedings (Lecture Notes in Computer Science, Vol. 8152)*, Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt (Eds.). Springer, 311–326. https://doi.org/10.1007/978-3-642-40885-4_22

[3] Thomas Arts and Jürgen Giesl. 2000. Termination of term rewriting using dependency pairs. *Theor. Comput. Sci.* 236, 1-2 (2000), 133–178. https://doi.org/10.1016/S0304-3975(99)00207-8

[4] Franz Baader and Tobias Nipkow. 1998. *Term rewriting and all that.* Cambridge University Press, Cambridge, England.

[5] Clemens Ballarin. 2010. Tutorial to Locales and Locale Interpretation. http://isabelle.in.tum.de/doc/locales.pdf.

[6] Frédéric Blanqui and Adam Koprowski. 2011. 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 (2011), 827–859. https://doi.org/10.1017/S0960129511000120

[7] Leonardo Mendonça de Moura and Nikolaj S. Bjørner. 2008. Z3: An Efficient SMT Solver. In *Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2008, Budapest, Hungary, March 29-April 6, 2008. Proceedings (Lecture Notes in Computer Science, Vol. 4963)*, C. R. Ramakrishnan and Jakob Rehof (Eds.). Springer, 337–340. https://doi.org/10.1007/978-3-540-78800-3_24

[8] Michael Hanus. 1994. The Integration of Functions into Logic Programming: From Theory to Practice. *J. Log. Program.* 19/20 (1994), 583–628. https://doi.org/10.1016/0743-1066(94)90034-5

[9] Jieh Hsiang, Hélène Kirchner, Pierre Lescanne, and Michaël Rusinowitch. 1992. The Term Rewriting Approach to Automated Theorem Proving. *J. Log. Program.* 14, 1&2 (1992), 71–99. https://doi.org/10.1016/0743-1066(92)90047-7

[10] Sam Kamin and Jean-Jacques Levy. 1980. *Two generalizations of the recursive path ordering.* Technical Report. Department of Computer Science, University of Illinois at Urbana-Champaign.

[11] Stéphane Kaplan. 1987. Simplifying Conditional Term Rewriting Systems: Unification, Termination and Confluence. *J. Symb. Comput.* 4, 3 (1987), 295–334. https://doi.org/10.1016/S0747-7171(87)80010-X

[12] Yutaka Kikuchi and Takuya Katayama. 1991. An Application of Term Rewriting Systems for Functional Programming. In *Distributed Environments: Software Paradigms and Workstations*, Yutaka Ohno and Toshiko Matsuda (Eds.). Springer Japan, 79–106. https://doi.org/10.1007/978-4-431-68144-1_7

[13] D. E. Knuth and P. B. Bendix. 1983. Simple Word Problems in Universal Algebras. In *Automation of Reasoning: 2: Classical Papers on Computational Logic 1967–1970*, Jörg H. Siekmann and Graham Wrightson (Eds.). Springer Berlin Heidelberg, 342–376.

[14] Dallas Lankford. 1979. *On Proving Term Rewrite Systems are Noetherian.* Technical Report MTP-3. Louisiana Technical University, Ruston, LA, USA.

[15] Salvador Lucas and Raúl Gutiérrez. 2018. Use of logical models for proving infeasibility in term rewriting. *Inf. Process. Lett.* 136 (2018), 90–95. https://doi.org/10.1016/J.IPL.2018.04.002

[16] Salvador Lucas, José Meseguer, and Raúl Gutiérrez. 2018. The 2D Dependency Pair Framework for conditional rewrite systems. Part I: Definition and basic processors. *J. Comput. Syst. Sci.* 96 (2018), 74–106. https://doi.org/10.1016/J.JCSS.2018.04.002

[17] Florian Meßner. 2023. nonreach 1.2. In *Proceedings of the 12nd International Workshop on Confluence*, Cyrille Chenavier and Sarah Winkler (Eds.). 75. http://cl-informatik.uibk.ac.at/iwc/iwc2023.pdf

[18] Florian Meßner and Christian Sternagel. 2019. nonreach - A Tool for Nonreachability Analysis. In *Tools and Algorithms for the Construction and Analysis of Systems - 25th International Conference, TACAS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11427)*, Tomás Vojnar and Lijun Zhang (Eds.). Springer, 337–343. https://doi.org/10.1007/978-3-030-17462-0_19

[19] Aart Middeldorp. 1997. Call by Need Computations to Root-Stable Form. In *Conference Record of POPL'97: The 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Papers Presented at the Symposium, Paris, France, 15-17 January 1997*, Peter Lee, Fritz Henglein, and Neil D. Jones (Eds.). ACM Press, 94–105. https://doi.org/10.1145/263699.263711

[20] Aart Middeldorp, Julian Nagele, and Kiraku Shintani. 2021. CoCo 2019: report on the eighth confluence competition. *Int. J. Softw. Tools Technol. Transf.* 23, 6 (2021), 905–916. https://doi.org/10.1007/S10009-021-00620-4

[21] Friedrich Neurauter, Aart Middeldorp, and Harald Zankl. 2010. Monotonicity Criteria for Polynomial Interpretations over the Naturals. In *Automated Reasoning, 5th International Joint Conference, IJCAR 2010, Edinburgh, UK, July 16-19, 2010. Proceedings (Lecture Notes in Computer Science, Vol. 6173)*, Jürgen Giesl and Reiner Hähnle (Eds.). Springer, 502–517. https://doi.org/10.1007/978-3-642-14203-1_42

[22] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. 2002. *Isabelle/HOL - A Proof Assistant for Higher-Order Logic.* Lecture Notes in Computer Science, Vol. 2283. Springer.

[23] Enno Ohlebusch. 2002. *Advanced topics in term rewriting.* Springer Science & Business Media, Berlin/Heidelberg, Germany.

[24] Yusuke Oi, Nao Hirokawa, and Teppei Saito. 2024. Moca 0.3: A First-Order Theorem Prover for Horn Clauses. In *Proceedings of the 13rd International Workshop on Confluence*, Mauricio Ayala-Rincón and Sarah Winkler (Eds.). 69–70. http://cl-informatik.uibk.ac.at/iwc/iwc2024.pdf

[25] Peter Schneider-Kamp, Jürgen Giesl, Alexander Serebrenik, and René Thiemann. 2006. Automated Termination Analysis for Logic Programs by Term Rewriting. In *Logic-Based Program Synthesis and Transformation, 16th International Symposium, LOPSTR 2006, Venice, Italy, July 12-14, 2006, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 4407)*, Germán Puebla (Ed.). Springer, 177–193. https://doi.org/10.1007/978-3-540-71410-1_13

[26] Christian Sternagel and Thomas Sternagel. 2017. Certifying Confluence of Quasi-Decreasing Strongly Deterministic Conditional Term Rewrite Systems. In *Automated Deduction - CADE 26 - 26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6-11, 2017, Proceedings (Lecture Notes in Computer Science, Vol. 10395)*, Leonardo de Moura (Ed.). Springer, 413–431. https://doi.org/10.1007/978-3-319-63046-5_26

[27] Christian Sternagel and René Thiemann. 2010. Executable Multivariate Polynomials. *Arch. Formal Proofs* 2010 (2010).

[28] Christian Sternagel and René Thiemann. 2014. The Certification Problem Format. In *Proceedings Eleventh Workshop on User Interfaces for Theorem Provers, UITP 2014, Vienna, Austria, 17th July 2014 (EPTCS, Vol. 167)*, Christoph Benzmüller and Bruno Woltzenlogel Paleo (Eds.). 61–72. https://doi.org/10.4204/EPTCS.167.8

[29] Christian Sternagel and René Thiemann. 2014. Formalizing Monotone Algebras for Certification of Termination and Complexity Proofs. In *Rewriting and Typed Lambda Calculi - Joint International Conference, RTA-TLCA 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings (Lecture Notes in Computer Science, Vol. 8560)*, Gilles Dowek (Ed.). Springer, 441–455. https://doi.org/10.1007/978-3-319-08918-8_30

[30] Christian Sternagel, René Thiemann, and Akihisa Yamada. 2021. A Formalization of Weighted Path Orders and Recursive Path Orders. *Archive of Formal Proofs* (September 2021). https://isa-afp.org/entries/Weighted_Path_Order.html, Formal proof development.

[31] René Thiemann, Guillaume Allais, and Julian Nagele. 2012. On the Formalization of Termination Techniques based on Multiset Orderings. In *23rd International Conference on Rewriting Techniques and Applications, RTA 2012, Nagoya, Japan, May 28-June 2, 2012. Proceedings (LIPIcs, Vol. 15)*, Ashish Tiwari (Ed.). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 339–354. https://doi.org/10.4230/LIPIcs.RTA.2012.339

[32] René Thiemann, Jonas Schöpf, Christian Sternagel, and Akihisa Yamada. 2020. Certifying the Weighted Path Order (Invited Talk). In *5th International Conference on Formal Structures for Computation and Deduction, FSCD 2020, June 29-July 6, 2020, Paris, France (Virtual Conference) (LIPIcs, Vol. 167)*, Zena M. Ariola (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 4:1–4:20. https://doi.org/10.4230/LIPICS.FSCD.2020.4

[33] René Thiemann and Christian Sternagel. 2009. Certification of Termination Proofs Using CeTA. In *Theorem Proving in Higher Order Logics, 22nd International Conference, TPHOLs 2009, Munich, Germany, August 17-20, 2009. Proceedings (Lecture Notes in Computer Science, Vol. 5674)*, Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel (Eds.). Springer, 452–468. https://doi.org/10.1007/978-3-642-03359-9_31

[34] René Thiemann and Elias Wenninger. 2023. A Verified Efficient Implementation of the Weighted Path Order. *Archive of Formal Proofs* (June 2023). https://isa-afp.org/entries/Efficient_Weighted_Path_Order.html, Formal proof development.

[35] Akihisa Yamada. 2022. Term Orderings for Non-reachability of (Conditional) Rewriting. In *Automated Reasoning - 11th International Joint Conference, IJCAR 2022, Haifa, Israel, August 8-10, 2022, Proceedings (Lecture Notes in Computer Science, Vol. 13385)*, Jasmin Blanchette, Laura Kovács, and Dirk Pattinson (Eds.). Springer, 248–267. https://doi.org/10.1007/978-3-031-10769-6_15

[36] Akihisa Yamada. 2022. Tuple Interpretations for Termination of Term Rewriting. *J. Autom. Reason.* 66, 4 (2022), 667–688. https://doi.org/10.1007/S10817-022-09640-4

[37] Akihisa Yamada, Keiichirou Kusakari, and Toshiki Sakabe. 2015. A unified ordering for termination proving. *Sci. Comput. Program.* 111 (2015), 110–134. https://doi.org/10.1016/J.SCICO.2014.07.009

Received 2024-09-17; accepted 2024-11-19