Completion for Logically Constrained Rewriting

Sarah Winkler

Department of Computer Science, University of Innsbruck, Austria
sarah.winkler@uibk.ac.at
 https://orcid.org/0000-0001-8114-3107

Aart Middeldorp

Department of Computer Science, University of Innsbruck, Austria
aart.middeldorp@uibk.ac.at
 https://orcid.org/0000-0001-7366-8464

──── **Abstract** ────

We propose an abstract completion procedure for logically constrained term rewrite systems (LCTRSs). This procedure can be instantiated to both standard Knuth-Bendix completion and ordered completion for LCTRSs, and we present a succinct and uniform correctness proof. A prototype implementation illustrates the viability of the new completion approach.

1 Introduction

Rewriting in the presence of side constraints captures simplification processes in various areas, such as expression rewriting in compilers, theorem provers, or SMT solvers [10, 15, 17]. The imposed side constraints can often be expressed as logical formulas. *Logically constrained rewrite systems* [13] formalize this rewriting mechanism, admitting side constraints over an arbitrary first-order logic. Though their application for practical analysis tasks relies on satisfiability checks in the respective logic, thanks to the advent of powerful SMT solvers in the last decade LCTRSs are valuable in a wide range of areas, including program verification [7].

Often simplification procedures aim for unique results. *Knuth-Bendix completion* [11] thus poses a natural means to obtain a presentation of the rewrite system which is confluent and terminating, such that unique results are guaranteed. In particular, such a presentation can be used to decide the validity problem. Standard completion may fail if unorientable equations are encountered. To address this drawback, *ordered completion* was proposed by Bachmair, Dershowitz, and Plaisted [3]. This variant of completion never fails, at the price of the resulting system being only ground complete.

In this paper we propose an abstract inference system for completion of LCTRSs. This deduction scheme can be instantiated to both standard and ordered completion procedures, depending on the success condition satisfied by a run. To this end, we also state and prove a critical pair lemma for LCTRSs. Correctness proofs of completion procedures traditionally relied on proof orders. In contrast, we give proofs that exploit the more recent notion of

peak decreasingness [9] to show (ground) confluence, which is the key part of the proof. This approach permits a succinct and uniform proof for finite runs of both standard and ordered completion. It also shows how peak decreasingness does not only apply to ordered completion, but also extends to the considerably more intricate setting of constrained rewriting.

To ensure termination of the resulting rewrite system, we propose the notion of a *constrained reduction order*, which generalizes the recursive path order presented in [13]. Since any terminating LCTRS gives rise to such an order, the proposed completion procedures can also be implemented using termination tools instead of a fixed reduction order, a known approach in the unconstrained setting [19]. We outline an implementation within the tool Ctrl [14] and give examples that illustrate the practicality of our method.

Overview.   The remainder of this paper is organized as follows. In Section 2 we summarize the relevant background. Section 3 is devoted to constrained reduction orders. To analyze peaks over LCTRSs, we prove a critical pair lemma in Section 4. Our inference system along with all proofs is presented in Section 5. Section 6 outlines our implementation in Ctrl before we conclude in Section 7. Due to space limitations, some proofs were moved to an appendix.

2 Preliminaries

We assume familiarity with the basic notions of term rewrite systems (TRSs) and completion [1, 2], but shortly recapitulate terminology and notation that we use in the remainder. In particular, we recall the notion of logically constrained rewriting as defined in [7, 13].

Terms.   We assume a sorted signature $\mathcal{F} = \mathcal{F}_{\text{terms}} \cup \mathcal{F}_{\text{theory}}$. The set $\mathcal{T}(\mathcal{F}, \mathcal{V})$ denotes the terms over this signature. We assume a mapping $\mathcal{I}$ which assigns to every sort ι occurring in $\mathcal{F}_{\text{theory}}$ a carrier set $\mathcal{I}(\iota)$, and an interpretation $\mathcal{J}$ that assigns to every symbol $f \in \mathcal{F}_{\text{theory}}$ of sort $\iota_1 \times \cdots \times \iota_n \to \kappa$ a function $f_{\mathcal{J}} \colon \mathcal{I}(\iota_1) \times \cdots \times \mathcal{I}(\iota_n) \to \mathcal{I}(\kappa)$. Moreover, for every sort ι occurring in $\mathcal{F}_{\text{theory}}$ we assume a set $\mathcal{V}\mathsf{al}_\iota \subseteq \mathcal{F}_{\text{theory}}$ of value symbols, such that all $c \in \mathcal{V}\mathsf{al}_\iota$ are constants of sort ι and $\mathcal{J}$ constitutes a bijective mapping between $\mathcal{V}\mathsf{al}_\iota$ and $\mathcal{I}(\iota)$. Thus there exists a constant symbol for every value in the carrier set. The interpretation $\mathcal{J}$ naturally extends to an interpretation of ground terms, mapping ground terms to values:

$$[f(t_1, \ldots, t_n)]_{\mathcal{J}} = f_{\mathcal{I}}([t_1]_{\mathcal{J}}, \ldots, [t_n]_{\mathcal{J}})$$

Thus every ground term has a unique value. We demand that theory symbols and term symbols overlap only on values, i.e., $\mathcal{F}_{\text{terms}} \cap \mathcal{F}_{\text{theory}} \subseteq \mathcal{V}\mathsf{al}$ holds. A term in $\mathcal{T}(\mathcal{F}_{\text{theory}}, \mathcal{V})$ is called a *logical* term. Moreover we assume existence of a sort bool such that $\mathcal{I}(\mathsf{bool}) = \mathbb{B} = \{\top, \bot\}$, $\mathcal{V}\mathsf{al}_{\mathsf{bool}} = \{\mathsf{true}, \mathsf{false}\}$, $[\mathsf{true}]_{\mathcal{J}} = \top$, and $[\mathsf{false}]_{\mathcal{J}} = \bot$ hold. Logical terms of sort bool are called *constraints*. A constraint φ is *valid* if $[\varphi\gamma]_{\mathcal{J}} = \top$ for all substitutions γ such that $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$.

Rewriting with Constraints.   A *constrained equation* is a triple $\ell \approx r\ [\varphi]$ where $\ell, r \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ are of the same sort and φ is a constraint. If $\varphi = \mathsf{true}$ then the constraint is often omitted, and the equation denoted as $\ell \approx r$. Sometimes $s \simeq t$ is used to abbreviate "$s \approx t$ or $t \approx s$". A *constrained rewrite rule* is a constrained equation such that $\mathsf{root}(\ell) \in \mathcal{F}_{\text{terms}} \setminus \mathcal{F}_{\text{theory}}$ holds and which is denoted $\ell \to r\ [\varphi]$. For a set of constrained equations $\mathcal{E}$, we write $\mathcal{E}^{-1}$ for $\{v \approx u\ [\varphi] \mid u \approx v\ [\varphi] \in \mathcal{E}\}$ and $\mathcal{E}^{\pm}$ to denote $\mathcal{E} \cup \mathcal{E}^{-1}$. A set of constrained rewrite rules is called a *logically constrained rewrite system* (LCTRS for short). We now define rewriting using constrained equations. To this end, a substitution σ is said to *respect* a constraint φ if $\varphi\sigma$ is valid and $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$.

▶ **Definition 1.** Let $\mathcal{E}$ be a set of constrained equations.

- A *calculation step* $s \to_{\mathsf{calc}} t$ satisfies $s = C[f(s_1, \ldots, s_n)]$ for some $f \in \mathcal{F}_{\text{theory}} \setminus \mathcal{V}\text{al}$, $t = C[u]$, $s_i \in \mathcal{V}\text{al}$ for all $1 \leqslant i \leqslant n$, and $u \in \mathcal{V}\text{al}$ is the value symbol of $[f(s_1, \ldots, s_n)]_{\mathcal{J}}$. In this case $f(x_1, \ldots, x_n) \to y \ [y = f(x_1, \ldots, x_n)]$ with $y \in \mathcal{V}$ is a *calculation rule*.
- A *rule step* $s \to_{\ell \approx r \ [\varphi]} t$ satisfies $s = C[\ell\sigma]$, $t = C[r\sigma]$, and σ respects φ.

We also write $\to_{\mathsf{rule}, \mathcal{E}}$ to refer to the relation $\{\to_\alpha\}_{\alpha \in \mathcal{E}}$, and denote $\to_{\mathsf{calc}} \cup \to_{\mathsf{rule}, \mathcal{E}}$ by $\to_{\mathcal{E}}$.

We sometimes write $\to_{p, \mathcal{E}}$ to indicate that the rewrite step takes place at position p. The subscript $\mathcal{E}$ is dropped if clear from the context. Moreover, the LCTRS $\mathcal{R}_{\mathsf{calc}}$ refers to the set of all calculation rules. In contrast to [13] we also use equations for rewriting, so we do not require that a rule step using $\ell \approx r \ [\varphi]$ with substitution σ satisfies $\sigma(x) \in \mathcal{V}\text{al}$ for all logical variables of the rule, i.e., also for $x \in \mathcal{V}\text{ar}(r) \setminus (\mathcal{V}\text{ar}(\ell) \cup \mathcal{V}\text{ar}(\varphi))$.

Note that $\to_{\mathsf{calc}}$ is terminating since calculation steps strictly reduce the number of non-value symbols in terms.

▶ **Example 2.** Consider the sort int (besides bool) and let $\mathcal{F}_{\text{theory}}$ consist of symbols $\cdot$, $+$, $-$, $\leqslant$, and $\geqslant$ as well as values n for all $n \in \mathbb{Z}$, with the usual interpretations on $\mathbb{Z}$. Let $\mathcal{F}_{\text{terms}} = \mathcal{V}\text{al} \cup \{\mathsf{fact}\}$. The LCTRS $\mathcal{R}$ consisting of the rules

$$\mathsf{fact}(x) \to 1 \quad [x \leqslant 0] \qquad\qquad \mathsf{fact}(x) \to \mathsf{fact}(x - 1) \cdot x \quad [x - 1 \geqslant 0]$$

admits the following rewrite steps:

$$
\begin{aligned}
\mathsf{fact}(2) &\to_{\mathsf{rule}} \mathsf{fact}(2 - 1) \cdot 2 && (\text{as } 2 - 1 \geqslant 0 \text{ is valid}) \\
&\to_{\mathsf{calc}} \mathsf{fact}(1) \cdot 2 &\to_{\mathsf{rule}} (\mathsf{fact}(1 - 1) \cdot 1) \cdot 2 \quad\ & (\text{as } 1 - 1 \geqslant 0 \text{ is valid}) \\
&\to_{\mathsf{calc}} (\mathsf{fact}(0) \cdot 1) \cdot 2 &\to_{\mathsf{rule}} (1 \cdot 1) \cdot 2 \quad\ & (\text{as } 0 \leqslant 0 \text{ is valid}) \\
&\to_{\mathsf{calc}}^+ 2
\end{aligned}
$$

An LCTRS $\mathcal{R}$ is *terminating* if $\to_{\mathcal{R}}$ is well-founded, and *confluent* if $_{\mathcal{R}}^* \leftarrow \cdot \to_{\mathcal{R}}^* \subseteq \to_{\mathcal{R}}^* \cdot {_{\mathcal{R}}^*}\leftarrow$. We use *peak decreasingness* [9] as confluence criterion. An abstract rewrite system $\mathcal{A} = \langle A, \{\to_\alpha\}_{\alpha \in I} \rangle$ is *peak decreasing* if there exists a well-founded order $>$ on I such that for all $\alpha, \beta \in I$ the inclusion $_\alpha\leftarrow \cdot \to_\beta \ \subseteq \xleftrightarrow[\vee\alpha\beta]{*}$ holds. Here $\vee\alpha\beta$ denotes the set $\{\gamma \mid \alpha > \gamma \text{ or } \beta > \gamma\}$.

▶ **Lemma 3** ([9]). *Every peak decreasing ARS is confluent.*

Rewriting Constrained Terms. Logically constrained rewriting aims to rewrite unconstrained terms with constrained rules. However, for the sake of analysis, rewriting *constrained terms* is useful. In particular, our completion procedure will maintain sets of constrained equations, and rewrite constrained terms. We recall the relevant notions [7, 13].

A *constrained term* is a pair $s \ [\varphi]$ of a term s and a constraint φ. Two constrained terms $s \ [\varphi]$ and $t \ [\psi]$ are *equivalent*, denoted by $s \ [\varphi] \sim t \ [\psi]$, if for every substitution γ respecting φ there is some substitution δ that respects ψ such that $s\gamma = t\delta$, and vice versa. For example, $\mathsf{fact}(x) \cdot x \ [x = 1 \wedge x < y] \sim \mathsf{fact}(1) \cdot y \ [y > 0 \wedge y < 2]$ holds, but these terms are not equivalent to $\mathsf{fact}(x) \cdot y \ [x = y]$ or $\mathsf{fact}(1) \ [\mathsf{true}]$.

▶ **Definition 4.** Let $\mathcal{E}$ be a set of constrained equations.

- A *calculation step* $s \ [\varphi] \to_{\mathsf{calc}} t \ [\varphi \wedge x = f(s_1, \ldots, s_n)]$ satisfies $s = C[f(s_1, \ldots, s_n)]$ for some $f \in \mathcal{F}_{\text{theory}} \setminus \mathcal{F}_{\text{terms}}$ and $t = C[x]$ such that $s_1, \ldots, s_n \in \mathcal{V}\text{ar}(\varphi) \cup \mathcal{V}\text{al}$ and x is a fresh variable.
- A constraint rewrite rule $\alpha : \ell \to r \ [\psi]$ admits a *rule step* $s \ [\varphi] \to_\alpha t \ [\varphi]$ if φ is satisfiable, $s = C[\ell\sigma]$, $t = C[r\sigma]$, $\sigma(x) \in \mathcal{V}\text{al} \cup \mathcal{V}\text{ar}(\varphi)$ for all $x \in \mathcal{V}\text{ar}(\psi)$, and $\varphi \Rightarrow \psi\sigma$ is valid.

Given an LCTRS $\mathcal{E}$, we again write $\to_{\mathsf{rule},\,\mathcal{E}}$ for $\{\to_\alpha\}_{\alpha\in\mathcal{E}}$. The main rewrite relation $\to_\mathcal{E}$ on constrained terms is defined as $\sim \cdot (\to_{\mathsf{calc}} \cup \to_{\mathsf{rule},\,\mathcal{E}}) \cdot \sim$.

▶ **Example 5.** Consider the LCTRS from Example 2, the constraint $\varphi = x \geqslant 1 \wedge y \geqslant 0$, and let z be a fresh variable. Then the following rewrite steps are possible:

$$\mathsf{fact}(x+y)\ [\varphi] \to_{\mathsf{rule}} \mathsf{fact}(x+y-1)\cdot(x+y)\ [\varphi]$$
$$\mathsf{fact}(x+y)\ [\varphi] \to_{\mathsf{calc}} \mathsf{fact}(z)\ [\varphi \wedge z = x+y]$$

The following key results relate rewriting on constrained terms to rewriting on unconstrained terms.

▶ **Lemma 6** ([13, Lemma 2]). *If $s\ [\varphi] \to_{\mathsf{rule}} t\ [\psi]$ and γ respects φ then $s\gamma \to_{\mathsf{rule}} t\gamma$. The two steps take place at the same positions.*

▶ **Lemma 7** ([7, Theorems 2.19 and 2.20]). *Suppose $s\ [\varphi] \to_{p,\mathcal{R}} t\ [\psi]$.*
1. *If γ respects φ then $s\gamma \to_{p,\mathcal{R}} t\delta$ for some substitution δ respecting ψ.*
2. *If δ respects ψ then $s\gamma \to_{p,\mathcal{R}} t\delta$ for some substitution γ respecting φ.*

Using a fresh binary term symbol $\langle \cdot, \cdot \rangle$, we show a slightly stronger version of Lemma 7.

▶ **Lemma 8.** *Suppose $\langle s, u \rangle\ [\varphi] \to_\mathcal{R} \langle t, v \rangle\ [\psi]$ at a position of the form $1p$.*
1. *If γ respects φ then $s\gamma \to_\mathcal{R} t\delta$ and $u\gamma = v\delta$ for some substitution δ respecting ψ.*
2. *If δ respects ψ then $s\gamma \to_\mathcal{R} t\delta$ and $u\gamma = v\delta$ for some substitution γ respecting φ.*

Proof. Note that whenever $\langle t_1, t_2 \rangle \sim w$ for terms t_1 and t_2 then w must be of the form $\langle w_1, w_2 \rangle$, by the definition of $\sim$ and because respectful substitutions introduce only values. We can therefore consider a step

$$\langle s, u \rangle\ [\varphi] \sim \langle s', u' \rangle\ [\varphi'] \xrightarrow[1p]{} \langle t', v' \rangle\ [\psi'] \sim \langle t, v \rangle\ [\psi] \tag{1}$$

1. Since γ respects φ there is some γ' respecting φ' such that $\langle s, u \rangle\gamma = \langle s', u' \rangle\gamma'$.
   First, if (1) involves a rule step then $\varphi' = \psi'$. Hence $\langle s', u' \rangle\gamma' \to_{\mathsf{rule},1p} \langle t', v' \rangle\gamma'$ by Lemma 6 and we have $u'\gamma' = v'\gamma'$. Because γ' respects ψ' and $\langle t', v' \rangle\ [\psi'] \sim \langle t, v \rangle\ [\psi]$ there is some δ respecting ψ such that $\langle t', v' \rangle\gamma' = \langle t, v \rangle\delta$. We thus have $s\gamma = s'\gamma' \to_\mathcal{R} t'\gamma' = t\delta$ and $u\gamma = u'\gamma' = v'\gamma' = v\delta$.
   Second, suppose (1) involves a calculation step. By the definition of $\to_{\mathsf{calc}}$ we have $\psi' = (\varphi' \wedge x = f(s_1, \ldots, s_n))$ for some $x \in \mathcal{V}$, $f \in \mathcal{F}_{\mathrm{theory}}$, and $s_1, \ldots, s_n \in \mathcal{V}\mathrm{ar}(\varphi') \cup \mathcal{V}\mathrm{al}$. So $f(s_1, \ldots, s_n)\gamma' \in \mathcal{T}(\mathcal{F}_{\mathrm{theory}})$ since γ' respects φ'. For w being the value symbol corresponding to $f(s_1, \ldots, s_n)\gamma'$, the substitution β given by $\beta(y) = w$ if $y = x$ and $\beta(y) = \gamma'(y)$ otherwise, respects ψ' and satisfies both $s'\gamma' \to_{\mathsf{calc}} t'\beta$ and $u'\gamma' = v'\beta$ because x is fresh. Because $\langle t', v' \rangle\ [\psi'] \sim \langle t, v \rangle\ [\psi]$ there is some δ respecting ψ such that $\langle t', v' \rangle\beta = \langle t, v \rangle\delta$. So $s\gamma = s'\gamma' \to_\mathcal{R} t'\beta = t\delta$ and $u\gamma = u'\gamma' = v'\beta = v\delta$.
2. Similar, see the appendix.   ◀

We conclude this section with an auxiliary result relating rewrite steps on constrained term pairs to steps on unconstrained term pairs.

▶ **Lemma 9.** *Suppose the LCTRS $\mathcal{R}$ admits a rewrite step $\langle s, u \rangle\ [\varphi] \to_\mathcal{R} \langle t, u \rangle\ [\psi]$. For all substitutions γ and δ and all contexts C,*
1. *if $C[s\gamma] \xleftrightarrow[s\approx u\ [\varphi]]{} C[u\gamma]$ then $C[s\gamma] \xrightarrow{\mathcal{R}} \cdot \xleftrightarrow[t\approx u\ [\psi]]{} C[u\gamma]$, and*
2. *if $C[t\delta] \xleftrightarrow[t\approx u\ [\psi]]{} C[u\delta]$ then $C[t\delta] \xleftarrow{\mathcal{R}} \cdot \xleftrightarrow[s\approx u\ [\varphi]]{} C[u\delta]$.*

Proof.

1. The substitution γ respects φ. By Lemma 8(1) there is some δ respecting ψ such that $s\gamma \to_{\mathcal{R}} t\delta$ and $u\gamma = u\delta$. Hence $C[s\gamma] \xrightarrow[\mathcal{R}]{} C[t\delta] \xleftrightarrow[t \approx u \ [\psi]]{} C[u\delta] = C[u\gamma]$.

2. The substitution δ respects ψ. By Lemma 8(2) there is some γ respecting φ such that $s\gamma \to_{\mathcal{R}} t\delta$ and $u\gamma = u\delta$. Hence $C[t\delta] \xleftarrow[\mathcal{R}]{} C[s\gamma] \xleftrightarrow[s \approx u \ [\varphi]]{} C[u\gamma] = C[u\delta]$. ◄

3 Constrained Reduction Orders

To ensure termination of the resulting system, completion procedures rely on reduction orders. In [13] the following definition of a recursive path order was given.

▶ **Definition 10.** Suppose $\mathcal{F}_{\text{theory}}$ contains a symbol $>_\iota$ for every sort ι occurring in $\mathcal{F}_{\text{theory}}$ such that $>_\iota$ is interpreted as a (partial) well-founded order $\sqsupset_\iota$ on $\mathcal{I}_\iota$. Moreover, let $>^p$ be a precedence on $\mathcal{F}_{\text{terms}} \setminus \mathcal{F}_{\text{theory}}$. For terms s and t and constraint φ

1. $s \geqslant^{\text{rpo}}_{[\varphi]} t$ if one of the following alternatives applies:
   a. $s, t \in \mathcal{T}(\mathcal{F}_{\text{theory}}, \mathcal{V}\text{ar}(\varphi))$ and $\varphi \Rightarrow (s = t \vee s >_{\text{sort}(s)} t)$ is valid,
   b. $s = f(s_1, \ldots, s_n)$ and $t = f(t_1, \ldots, t_n)$ with $f \notin \mathcal{F}_{\text{theory}}$ and $s_i \geqslant^{\text{rpo}}_{[\varphi]} t_i$ for all $1 \leqslant i \leqslant n$,
   c. $s >^{\text{rpo}}_{[\varphi]} t$, or $s = t$ and $s \in \mathcal{V}$;

2. $s >^{\text{rpo}}_{[\varphi]} t$ if one of the following alternatives applies:
   a. $s, t \in \mathcal{T}(\mathcal{F}_{\text{theory}}, \mathcal{V}\text{ar}(\varphi))$ and $\varphi \Rightarrow s >_{\text{sort}(s)} t$ is valid,
   b. $s = f(s_1, \ldots, s_n)$ for some $f \notin \mathcal{F}_{\text{theory}}$ and one of
      i. $s_i \geqslant^{\text{rpo}}_{[\varphi]} t$ for some $1 \leqslant i \leqslant n$,
      ii. $t = g(t_1, \ldots, t_m)$, either $g \in \mathcal{F}_{\text{theory}}$ or $f >^p g$, and $s >^{\text{rpo}}_{[\varphi]} t_j$ for all $1 \leqslant j \leqslant m$,
      iii. $t = f(t_1, \ldots, t_n)$, $s_i \geqslant^{\text{rpo}}_{[\varphi]} t_i$ for all $1 \leqslant i \leqslant n$ and $s_i >^{\text{rpo}}_{[\varphi]} t_i$ for some $1 \leqslant i \leqslant n$,
      iv. $t \in \mathcal{V}\text{ar}(\varphi)$.

We now generalize this notion.

▶ **Definition 11.** A ternary relation $>_{[\cdot]}$ on $\mathcal{T}(\mathcal{F}, \mathcal{V}) \times \mathcal{T}_{\text{bool}}(\mathcal{F}_{\text{theory}}, \mathcal{V}) \times \mathcal{T}(\mathcal{F}, \mathcal{V})$ is a *constrained reduction order* if there exists a reduction order $>$ such that $s >_{[\varphi]} t$ if and only if $s\gamma > t\gamma$ for all substitutions γ that respect φ.

The relation $>^{\text{rpo}}_{[\cdot]}$ is a constrained reduction order according to this definition (Lemma 9 in the full version of [13]).

▶ **Example 12.**

1. Any reduction order $>$ gives rise to a constrained reduction order in which the constraints are simply ignored: setting $s >_{[\varphi]} t$ for all φ whenever $s > t$ vacuously satisfies the definition.

2. Let $n \sqsupset_{\text{int}} m$ if $m \geqslant 0$ and $n > m$. For the LCTRS from Example 2, rule (1) can be oriented using the recursive path order by condition 2.b.ii in Definition 10. For rule (2) and $\varphi = x - 1 \geqslant 0$, we have $x >^{\text{rpo}}_{[\varphi]} x - 1$ by condition 2.a since the implication $x - 1 \geqslant 0 \implies x > x - 1$ is valid. From this we obtain $\text{fact}(x) >^{\text{rpo}}_{[\varphi]} \text{fact}(x - 1)$ by condition 2.b.iii. Moreover, $\text{fact}(x) >^{\text{rpo}}_{[\varphi]} x$ by 2.b.i such that $\text{fact}(x) >^{\text{rpo}}_{[\varphi]} \text{fact}(x - 1) \cdot x$ follows from 2.b.ii. Note that the rules could not have been oriented by RPO when ignoring the constraints.

3. A well-founded $\mathcal{F}$-algebra $\mathcal{A}$ extending $\mathcal{I}$ with monotone (wrt $\sqsupseteq_{\mathsf{int}}$) interpretations for the function symbols in $\mathcal{F}_{\mathrm{terms}}$ gives rise to a constrained reduction order: $s >_{[\varphi]} t$ if and only if $[\alpha]_{\mathcal{A}}(s) > [\alpha]_{\mathcal{A}}(t)$ is satisfied for all substitutions α that respect φ. Moreover, if the carrier of $\mathcal{A}$ is $\mathcal{V}\mathsf{al}$ then compatibility is decidable whenever the language of constraints is. For instance, consider again Example 2 and let $\mathcal{A}$ extend $\mathcal{I}$ by the monotone interpretation $\mathsf{fact}_{\mathcal{A}}(x) = (x+1)! + 1$ if $x \geqslant 0$ and $\mathsf{fact}_{\mathcal{A}}(x) = 2$ otherwise. We have

$$\mathsf{fact}_{\mathcal{A}}(x) = 2 > 1 = 1_{\mathcal{A}} \qquad\qquad\qquad \text{for all } x \leqslant 0$$
$$\mathsf{fact}_{\mathcal{A}}(x) = (x+1)! + 1 > (x! + 1) \cdot x = \mathsf{fact}_{\mathcal{A}}(x -_{\mathcal{A}} 1) \cdot_{\mathcal{A}} x \qquad\qquad \text{for all } x \geqslant 1$$

▶ **Lemma 13.** *Let $>_{[\cdot]}$ be a constrained reduction order and let φ and ψ be constraints.*
1. *The relation $>_{[\varphi]}$ is transitive.*
2. *If $s >_{[\varphi]} t$ and σ respects φ then $s\sigma >_{[\varphi]} t\sigma$.*
3. *If $s >_{[\varphi]} t$ then $C[s] >_{[\varphi]} C[t]$.*
4. *If $\psi \Rightarrow \varphi$ is valid and $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{V}\mathsf{ar}(\psi)$ then $s >_{[\varphi]} t$ implies $s >_{[\psi]} t$.*

▶ **Lemma 14.** *If $s\ [\psi] \to^{\sigma}_{\alpha,p} t\ [\psi]$ using $\alpha\colon \ell \to r\ [\varphi]$ satisfies $\ell >_{[\varphi]} r$ then $s >_{[\psi]} t$.*

Proof. By assumption $\psi \Rightarrow \varphi\sigma$ is valid and $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\psi)$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$. Now suppose γ is a substitution which respects ψ, i.e., the constraint $\psi\gamma$ is valid and $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\psi)$. Then $\varphi\sigma\gamma$ is valid and $(\sigma\gamma)(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$, so $\sigma\gamma$ respects φ. From $\ell >_{[\varphi]} r$ we thus obtain $s|_p\gamma = \ell\sigma\gamma > r\sigma\gamma = t|_p\gamma$, and hence $s\gamma > t\gamma$ by Lemma 13(3). ◀

A *reduction pair* $(>, \geqslant)$ consists of a reduction order $>$ and a reduction preorder $\geqslant$ such that $\to_{\mathsf{calc}} \subseteq \geqslant$ and $\geqslant \cdot > \cdot \geqslant\ \subseteq\ >$.

▶ **Lemma 15.** *If there is a reduction pair $(>, \geqslant)$ such that $\ell >_{[\varphi]} r$ for all $\ell \to r\ [\varphi] \in \mathcal{R}$ then $\mathcal{R}$ is terminating.*

Proof. By Lemma 14, the inclusion $\to_{\mathsf{calc}} \subseteq \geqslant$, and the compatibility of $>$ and $\geqslant$. ◀

Kop and Nishida (Lemmata 6 and 8 in the full version of [13]) showed that a suitable reduction preorder exists for the recursive path ordering. Similar to the case of plain term rewrite systems, any terminating LCTRS induces a constrained reduction order.

▶ **Lemma 16.** *If $\mathcal{R}$ is a terminating LCTRS then the relation defined by $s >_{[\varphi]} t$ if and only if $s\ [\varphi] \to^{+}_{\mathsf{rule}} t\ [\varphi]$ is a constrained reduction order.*

Proof. Let $>$ be the relation such that $s > t$ if and only if $s\ [\mathsf{true}] \to^{+}_{\mathsf{rule}} t\ [\mathsf{true}]$. Since $\mathcal{R}$ is terminating $>$ is a reduction order. So by Lemma 6 all substitutions γ respecting φ satisfy

$$s >_{[\varphi]} t \iff s\ [\varphi] \to^{+}_{\mathsf{rule}} t\ [\varphi] \implies s\gamma \to^{+}_{\mathsf{rule}} t\gamma \iff s\gamma > t\gamma \qquad\qquad ◀$$

The following example shows that a constrained reduction order is not necessarily compatible with the equivalence relation $\sim$ on constrained terms in the sense that $s\ [\varphi] \sim s'\ [\psi]$ and $s >_{[\varphi]} t$ imply $s' >_{[\psi]} t$.

▶ **Example 17.** For instance, for the constrained reduction order $>^{\mathsf{rpo}}_{[\cdot]}$ we have $\mathsf{f}(x) >_{[x=0]} x$ and $\mathsf{f}(x)\ [x = 0] \sim \mathsf{f}(0)\ [\mathsf{true}]$ but $\mathsf{f}(0) >_{[\mathsf{true}]} x$ does not hold.

We conclude this section by comparing our concept of a constrained reduction pair to definitions from the literature. Our notion resembles the definition by Falke and Kapur [5, Definition 23] for the theory of Peano arithmetic ($\mathcal{PA}$). Whereas they demand $C[s]$ ($\geqslant \cap \leqslant$) $C[t]$ for all $s \leftrightarrow^*_{\mathcal{PA}} t$, we use the more relaxed condition $\rightarrow_{\mathsf{calc}} \subseteq \geqslant$.

Fuhs et al. [6] define the order pair ($\succ_{\mathcal{P}ol}, \succeq_{\mathcal{P}ol}$) as a reduction pair processor for integer term rewrite systems. Since the order pair is based on *max-polynomial* interpretations, which are not strictly monotone, the resulting order on terms in not a reduction order, and hence does not produce a constrained reduction order in our sense.

4 Critical Pair Lemma

In this section we establish the Critical Pair Lemma, which is a key result to obtain confluence of the result of our completion procedure described in the next section.

▶ **Definition 18.** An *overlap* of an LCTRS $\mathcal{R}$ is a triple $\langle \ell_1 \rightarrow r_1 \ [\varphi_1], p, \ell_2 \rightarrow r_2 \ [\varphi_2] \rangle$ satisfying the following properties:
- $\ell_1 \rightarrow r_1 \ [\varphi_1]$ and $\ell_2 \rightarrow r_2 \ [\varphi_2]$ are variable-disjoint variants of rewrite rules in $\mathcal{R} \cup \mathcal{R}_{\mathsf{calc}}$,
- $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2)$,
- ℓ_1 and $\ell_2|_p$ are unifiable with mgu σ and $\sigma(x) \in \mathcal{T}(\mathcal{F}_{\text{theory}}, \mathcal{V})$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi_1) \cup \mathcal{V}\mathsf{ar}(\varphi_2)$,
- $\varphi_1\sigma \wedge \varphi_2\sigma$ is satisfiable, and
- if $p = \epsilon$ then $\ell_1 \rightarrow r_1 \ [\varphi_1]$ and $\ell_2 \rightarrow r_2 \ [\varphi_2]$ are not variants, or $\mathcal{V}\mathsf{ar}(r_1) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_1)$.

In this case $\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma \ [\varphi_1\sigma \wedge \varphi_2\sigma]$ is a *constrained critical pair*. The set of all constrained critical pairs of $\mathcal{R}$ is denoted by $\mathsf{CP}(\mathcal{R})$.

Note that in the last condition also $\mathcal{V}\mathsf{ar}(r_2) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_2)$ since we may assume that the two rules are variants.

▶ **Example 19.** Consider the following rewrite rules:

(1) $\quad \mathsf{f}(x) \rightarrow \mathsf{g}(x,x) \quad [x \leqslant 0]$ $\qquad$ (3) $\quad \mathsf{g}(\mathsf{f}(x), y) \rightarrow \mathsf{g}(x, z) \quad [x > 0 \wedge z > x]$

(2) $\quad \mathsf{h}(\mathsf{f}(x)) \rightarrow \mathsf{h}(x) \qquad [x \geqslant 0]$ $\qquad$ (4) $\quad \mathsf{g}(x, x + y) \rightarrow \mathsf{f}(y) \qquad [x > 0 \wedge y > 0]$

The constrained critical pair $\mathsf{h}(\mathsf{g}(x,x)) \approx \mathsf{h}(x) \ [x \leqslant 0 \wedge x \geqslant 0]$ is obtained from the overlap $\langle (1), 1, (2) \rangle$. There is also an overlap $\langle (3), \epsilon, (3') \rangle$ between rule (3) and a renamed version (3′) of itself, which gives rise to the critical pair $\mathsf{g}(x, z) \approx \mathsf{g}(x, w) \ [x > 0 \wedge z > x \wedge w > x]$. Finally, the constrained critical pair $\mathsf{g}(x, z) \approx \mathsf{f}(y) \ [x > 0 \wedge y > 0 \wedge z = x + y]$ originates from the overlap $\langle \mathcal{R}_{\mathsf{calc}}, 2, (4) \rangle$. There is no constrained critical pair between rules (1) and (3) since the conjunction $x \leqslant 0 \wedge x > 0 \wedge z > x$ is not satisfiable. There is also no critical pair between (3) and (4) because any mgu σ of $\mathsf{g}(\mathsf{f}(x), y)$ and $\mathsf{g}(x', x' + y')$ assigns (a variant of) $\mathsf{f}(x)$ to x', violating the third condition in Definition 18.

▶ **Lemma 20** (Constrained Critical Pair Lemma). *Let $\mathcal{R}$ be an LCTRS. If $t \ _{\mathcal{R}}\!\leftarrow s \rightarrow_{\mathcal{R}} u$ then $t \downarrow_{\mathcal{R}} u$ or $t \leftrightarrow_{\mathsf{CP}(\mathcal{R})} u$.*

Proof. We abbreviate $\ell_1 \rightarrow r_1 \ [\varphi_1]$ by α_1 and $\ell_2 \rightarrow r_2 \ [\varphi_2]$ by α_2, and consider a peak

$$t \xleftarrow[\alpha_1]{p_1, \sigma_1} s \xrightarrow[\alpha_2]{p_2, \sigma_2} u$$

where σ_1 and σ_2 denote the employed substitutions. We distinguish three cases.

(1) If $p_1 \parallel p_2$ then $t \xrightarrow[\alpha_2]{p_2, \sigma_2} s \xleftarrow[\alpha_1]{p_1, \sigma_1} u$ since the same substitutions can be used for the respective steps such that constraints are still respected.

Otherwise one position must be above the other one. Without loss of generality we assume that $p_1 \leqslant p_2$, so there must be a position p such that $p_2 = p_1 p$. We further assume that the rewrite rules α_1 and α_2 have no variables in common. Hence $\mathcal{D}om(\sigma_1) \cap \mathcal{D}om(\sigma_2) = \varnothing$ and the substitution $\sigma = \sigma_1 \cup \sigma_2$ is well-defined. We distinguish two further cases depending on whether the peak is an instance of an overlap.

(2) Suppose $\langle \alpha_1, p, \alpha_2 \rangle$ is an overlap. Let γ be a most general unifier of $\ell_2|_p$ and ℓ_1. We have $\ell_2\gamma[r_1\gamma]_p \approx r_2\gamma \; [\varphi_1\gamma \wedge \varphi_2\gamma] \in \mathsf{CP}(\mathcal{R})$. The substitution σ is a unifier of $\ell_2|_p$ and ℓ_1 because $(\ell_2|_p)\sigma = (\ell_2\sigma_2)|_p = \ell_1\sigma_1 = \ell_1\sigma$. Consequently there exists a substitution τ such that $\sigma = \gamma\tau$. Since validity of $\varphi_1\sigma = \varphi_1\gamma\tau$ and $\varphi_2\sigma = \varphi_2\gamma\tau$ implies validity of $(\varphi_1\gamma \wedge \varphi_2\gamma)\tau$ we have

$$\ell_2\sigma_2[r_1\sigma_1]_p = (\ell_2\gamma[r_1\gamma]_p)\tau \leftrightarrow_{\mathsf{CP}(\mathcal{R})} (r_2\gamma)\tau = r_2\sigma_2$$

and hence also $t \leftrightarrow_{\mathsf{CP}(\mathcal{R})} u$.

(3) Since $\varphi_1\sigma_1 \wedge \varphi_2\sigma_2 = \varphi_1\sigma \wedge \varphi_2\sigma = (\varphi_1\gamma \wedge \varphi_2\gamma)\tau$ is valid, the constraint $\varphi_1\gamma \wedge \varphi_2\gamma$ is satisfiable. So if $\langle \alpha_1, p, \alpha_2 \rangle$ is not an overlap then either $p = \epsilon$ and α_1 and α_2 are variants with $\mathcal{V}ar(r_1) \subseteq \mathcal{V}ar(\ell_1)$, or $p \notin \mathcal{P}os_{\mathcal{F}}(\ell_2)$. In the first case also $\mathcal{V}ar(r_2) \subseteq \mathcal{V}ar(\ell_2)$ must hold, which implies $r_1\sigma_1 = r_2\sigma_2$ and $t = u$. In the second case, there must be positions q_1 and q_2 such that $p = q_1 q_2$ and q_1 is a variable position in ℓ_2. Let x be the variable at $\ell_2|_{q_1}$, so $\sigma_2(x)|_{q_2} = \ell_1\sigma_1$. We define the substitution σ_2' as

$$\sigma_2'(y) = \begin{cases} \sigma_2(y)[r_1\sigma_1]_{q_2} & \text{if } y = x \\ \sigma_2(y) & \text{if } y \neq x \end{cases}$$

Clearly the step $\sigma_2(x) \rightarrow_{q_2, \alpha_1, \sigma_1} \sigma_2'(x)$ is valid. Hence $r_2\sigma_2 \rightarrow^* r_2\sigma_2'$ and $\ell_2\sigma_2[r_1\sigma_1]_p = \ell_2\sigma_2[\sigma_2'(x)]_{q_1} \rightarrow^* \ell_2\sigma_2'$. The substitution σ_2' also respects φ_2. This can be seen as follows. Since σ_2 respects φ_2 we have $\sigma_2(y) \in \mathcal{V}al$ for all $y \in \mathcal{V}ar(\varphi_2)$. So $x \notin \mathcal{V}ar(\varphi_2)$ as $x\sigma_2 \trianglerighteq \ell_1$ and left-hand sides of rules cannot be values. Therefore $\sigma_2(y) = \sigma_2'(y)$ holds for all $y \in \mathcal{V}ar(\alpha_2)$, and we have $\varphi_2\sigma_2 = \varphi_2\sigma_2'$. In summary, there exists a joining sequence

$$\ell_2\sigma_2[r_1\sigma_1]_p \xrightarrow[\ell_1 \rightarrow r_1 \; [\varphi_1]]{\sigma_1}{}^* \ell_2\sigma_2' \xrightarrow[\ell_2 \rightarrow r_2 \; [\varphi_2]]{\epsilon, \sigma_2'} r_2\sigma_2' \; {}^* \xleftarrow[\ell_1 \rightarrow r_1 \; [\varphi_1]]{\sigma_1} r_2\sigma_2 \qquad \blacktriangleleft$$

Extended Critical Pairs. For ordered rewriting in a constrained setting, we consider a reduction pair $(>, \geqslant)$ that gives rise to a constrained reduction order $>_{[\cdot]}$ and define

$$\mathcal{E}^> = \{ u\gamma \rightarrow v\gamma \; [\varphi\gamma] \mid u \approx v \; [\varphi] \in \mathcal{E}^{\pm} \text{ and } u\gamma >_{[\varphi\gamma]} v\gamma \}$$

for any set of constrained equations $\mathcal{E}$. Moreover, $\mathsf{CP}^>(\mathcal{E})$ denotes all constrained critical pairs originating from an overlap $\langle \ell_1 \approx r_1 \; [\varphi_1], p, \ell_2 \approx r_2 \; [\varphi_2] \rangle$ with most general unifier σ such that $\ell_1 \rightarrow r_1 \; [\varphi_1], \ell_2 \rightarrow r_2 \; [\varphi_2] \in \mathcal{E}^{\pm}$ and neither $r_1\sigma >_{[\varphi_1\sigma]} \ell_1\sigma$ nor $r_2\sigma >_{[\varphi_2\sigma]} \ell_2\sigma$. A reduction order is called *complete* for a set of constrained equations $\mathcal{E}$ if $s \leftrightarrow^*_{\mathcal{E}} t$ implies $s > t$, $s < t$, or $s = t$ for all ground terms s and t. The proof of the next result is in the appendix.

▶ **Lemma 21.** *If $>$ is complete for $\mathcal{R} \cup \mathcal{E}$ and $\mathcal{R} \subseteq >_{[\cdot]}$ then the inclusion $\xleftrightarrow[\mathsf{CP}(\mathcal{R} \cup \mathcal{E}^>)]{} \subseteq \xleftrightarrow[\mathsf{CP}^>(\mathcal{R} \cup \mathcal{E})]{} \cup \downarrow_{\mathcal{R} \cup \mathcal{E}^>}$ holds on ground terms.*

5   Abstract Completion

In this section we define an abstract inference system that can be instantiated to both standard and ordered completion. We consider a fixed reduction pair $(>, \geqslant)$ that gives rise to a constrained reduction order $>_{[\cdot]}$.

■ **Table 1** The inference rules of CKB.

deduce	$$\dfrac{\mathcal{E}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\ [\varphi]\}, \mathcal{R}}$$	if $\langle s, u \rangle\ [\varphi]\ {}_{\mathcal{R} \cup \mathcal{E}^\pm}\!\leftarrow \langle u, u \rangle\ [\varphi] \rightarrow_{\mathcal{R} \cup \mathcal{E}^\pm} \langle u, t \rangle\ [\varphi]$
compose	$$\dfrac{\mathcal{E}, \mathcal{R} \uplus \{s \rightarrow t\ [\varphi]\}}{\mathcal{E}, \mathcal{R} \cup \{s \rightarrow u\ [\psi]\}}$$	if $\langle s, t \rangle\ [\varphi] \rightarrow_{\mathcal{R} \cup \mathcal{E}^>} \langle s, u \rangle\ [\psi]$ and $s >_{[\varphi]} u$
orient	$$\dfrac{\mathcal{E} \uplus \{s \simeq t\ [\varphi]\}, \mathcal{R}}{\mathcal{E}, \mathcal{R} \cup \{s \rightarrow t\ [\varphi]\}}$$	if $s >_{[\varphi]} t$ and $\mathsf{root}(s) \in \mathcal{F}_{\text{terms}} \setminus \mathcal{F}_{\text{theory}}$
simplify	$$\dfrac{\mathcal{E} \uplus \{s \simeq t\ [\varphi]\}, \mathcal{R}}{\mathcal{E} \cup \{u \approx t\ [\psi]\}, \mathcal{R}}$$	if $\langle s, t \rangle\ [\varphi] \rightarrow_{\mathcal{R} \cup \mathcal{E}^>} \langle u, t \rangle\ [\psi]$
delete	$$\dfrac{\mathcal{E} \uplus \{s \approx t\ [\varphi]\}, \mathcal{R}}{\mathcal{E}, \mathcal{R}}$$	if $s\gamma = t\gamma$ for all γ respecting φ
collapse	$$\dfrac{\mathcal{E}, \mathcal{R} \uplus \{t \rightarrow s\ [\varphi]\}}{\mathcal{E} \cup \{u \approx s\ [\psi]\}, \mathcal{R}}$$	if $\langle t, s \rangle\ [\varphi] \rightarrow_{\mathcal{R} \cup \mathcal{E}^>} \langle u, s \rangle\ [\psi]$
split$_\mathcal{E}$	$$\dfrac{\mathcal{E} \uplus \{s \simeq t\ [\varphi]\}, \mathcal{R}}{\mathcal{E} \cup \{s \approx t\ [\varphi \wedge \neg\psi],\ s \approx t\ [\varphi \wedge \psi]\}, \mathcal{R}}$$	if $\mathcal{V}\mathsf{ar}(\psi) \subseteq \mathcal{V}\mathsf{ar}(\varphi)$
split$_\mathcal{R}$	$$\dfrac{\mathcal{E}, \mathcal{R} \uplus \{s \rightarrow t\ [\varphi]\}}{\mathcal{E}, \mathcal{R} \cup \{s \rightarrow t\ [\varphi \wedge \neg\psi],\ s \rightarrow t\ [\varphi \wedge \psi]\}}$$	if $\mathcal{V}\mathsf{ar}(\psi) \subseteq \mathcal{V}\mathsf{ar}(\varphi)$

▶ **Definition 22.** The inference system CKB of constrained (Knuth-Bendix) completion operates on pairs $(\mathcal{E}, \mathcal{R})$ consisting of constrained equations $\mathcal{E}$ and constrained rules $\mathcal{R}$, and consists of the inference rules of Table 1.

Below we provide some comments on the inference rules. The rewrite steps in e.g. simplify rewrite a *pair* $\langle s, t \rangle$ rather than a single term s. As observed in [7] it does not suffice to assume a rewrite step $s\ [\varphi] \rightarrow_\mathcal{R} u\ [\psi]$. For example, we have

$$\mathsf{f}(x + 0)\ [x > y] \sim \mathsf{f}(x + 0)\ [\mathsf{true}] \rightarrow_{\mathsf{calc}} \mathsf{f}(z)\ [z = x + 0] \sim \mathsf{f}(x)\ [\mathsf{true}] \sim \mathsf{f}(x)\ [x < y]$$

but it is not desirable to replace $y \approx \mathsf{f}(x + 0)\ [x > y]$ by $y \approx \mathsf{f}(x)\ [x < y]$. The same holds for compose, collapse, and deduce. The deduce rule is quite general in that it allows to add arbitrary equations that emerge from a peak between two $\mathcal{R} \cup \mathcal{E}$ steps. Below we will show that only (extended) critical pairs are necessary; hence as usual with completion procedures, an implementation will likely limit the application of deduce to these equations. Moreover, ordered rewriting is permitted in all rules that perform rewrite steps. Though this is uncommon in (unconstrained) standard completion, it gives more freedom to implementations and allows us to present only one set of inference rules for both settings. Furthermore, it is uncommon to perform a term comparison in compose. However, since additional variables may be introduced by $\rightarrow_{\mathsf{calc}}$ steps and $\sim$, $s >_{[\psi]} u$ need not hold. For instance, consider $>_{[\cdot]}$ defined as $s >_{[\cdot]} t$ if $s >^{\mathsf{lpo}} t$ holds, for some fixed reduction order $>^{\mathsf{lpo}}$ with precedence $\mathsf{f} > \mathsf{g}$. Given a rule $\mathsf{f}(x + y) \rightarrow \mathsf{g}(x + y)\ [\mathsf{true}]$, we have $\mathsf{f}(x + y) >_{[\mathsf{true}]} \mathsf{g}(x + y)$. We further have $\mathsf{g}(x + y) \rightarrow_{\mathsf{calc}} \mathsf{g}(z)\ [z = x + y]$ but $\mathsf{f}(x + y) >_{[z = x + y]} \mathsf{g}(z)$ does not hold.[1] Finally, the split rules are inspired by [8], they allow for a case distinction.

An inference step from equations and rules $(\mathcal{E}, \mathcal{R})$ to $(\mathcal{E}', \mathcal{R}')$ using one of the inference rules of Definition 22 is denoted by $(\mathcal{E}, \mathcal{R}) \vdash (\mathcal{E}', \mathcal{R}')$. We illustrate CKB on a concrete example, before presenting some basic properties related to inference steps.

---

[1] Note that if a pure $\rightarrow_{\mathsf{rule}}$ step is performed then $s >_{[\psi]} u$ is guaranteed by Lemma 14.

▶ **Example 23.** Consider the theory of integer arithmetic, RPO $>$ with precedence $\mathsf{h} > \mathsf{f} > \mathsf{g}$, and the following set of input equations:

(1) $\mathsf{f}(x,y) \approx \mathsf{f}(z,y) + 1 \; [x \geqslant 1 \wedge z = x - 1]$ (3) $\mathsf{f}(x,0) \approx \mathsf{g}(1,x) \; [x \leqslant 1]$

(2) $\mathsf{g}(0,y) \approx y \; [y \leqslant 0]$ (4) $\mathsf{h}(x) \approx \mathsf{f}(x,0) + 1$

We apply **orient** to equations (1) and (2) to obtain rewrite rules

(1) $\mathsf{f}(x,y) \to \mathsf{f}(z,y) + 1 \; [x \geqslant 1 \wedge z = x - 1]$ (2) $\mathsf{g}(0,y) \to y \; [y \leqslant 0]$

In rule (1) the variable z occurs on the right but not on the left-hand side, hence we **deduce** the critical pair

$$\mathsf{f}(w,y) + 1 \approx \mathsf{f}(z,y) + 1 \; [x \geqslant 1 \wedge z = x - 1 \wedge w = x - 1]$$

which can, however, be dropped using **delete**. (Note that a syntactic equality check of the equated terms would not suffice at this point.) Next we **orient** equation (3) from left to right. At this point the critical pair

$$\mathsf{g}(1,x) \approx \mathsf{f}(z,0) + 1 \; [x \geqslant 1 \wedge z = x - 1 \wedge x \leqslant 1]$$

results from the overlap $\langle(3), \epsilon, (1)\rangle$. But we can instead **deduce** the simpler, unconstrained equation (5) $\mathsf{g}(1,1) \approx \mathsf{f}(0,0) + 1$ as follows:

$$\mathsf{g}(1,1) \; [\mathsf{true}] \sim \mathsf{g}(1,x) \; [x = 1] \xleftarrow[\text{rule}]{(3)} \mathsf{f}(x,0) \; [x = 1] \sim \mathsf{f}(1,0) \; [\mathsf{true}] \sim \mathsf{f}(x,0) \; [x = 1 \wedge z = 0]$$

$$\xrightarrow[\text{rule}]{(1)} \mathsf{f}(z,0) + 1 \; [x = 1 \wedge z = 0] \sim \mathsf{f}(0,0) + 1 \; [\mathsf{true}]$$

When applying **simplify** with rule (3), equation (5) gets replaced by (6) $\mathsf{g}(1,1) \approx \mathsf{g}(1,0) + 1$. An application of **orient** produces the corresponding rule (6) $\mathsf{g}(1,1) \to \mathsf{g}(1,0) + 1$. A case split on $[x \leqslant 1]$ using $\mathsf{split}_\mathcal{E}$ followed by orientations replaces equation (4) by the rules (7) $\mathsf{h}(x) \to \mathsf{f}(x,0) + 1 \; [x \leqslant 1]$ and (8) $\mathsf{h}(x) \to \mathsf{f}(x,0) + 1 \; [\neg(x \leqslant 1)]$. Now **compose** using rule (3) can replace rule (7) by (9) $\mathsf{h}(x) \to \mathsf{g}(1,x) + 1 \; [x \leqslant 1]$. Since this is a pure rule step, $\mathsf{h}(x) >_{[x \leqslant 1]} \mathsf{g}(1,x) + 1$ holds by Lemma 14. Another application of **compose** applying rule (1) to rule (8) results in (10) $\mathsf{h}(x) \to \mathsf{f}(z,0) + 1 + 1 \; [x \geqslant 1 \wedge z = x - 1]$. We can apply **compose** again to (10), performing a calculation step to obtain (11) $\mathsf{h}(x) \to \mathsf{f}(x - 1, 0) + 2 \; [x \geqslant 1]$ (using $\mathsf{f}(z,y) + 2 \; [x \geqslant 1 \wedge z = x - 1] \sim \mathsf{f}(x - 1, y) + 2 \; [x \geqslant 1]$). Note that in both **compose** steps the orientation using $>_{[\cdot]}$ is preserved. At this point no equations are left, and all constrained critical pairs among the current set of rules

(1) $\mathsf{f}(x,y) \to \mathsf{f}(z,y) + 1 \; [x \geqslant 1 \wedge z = x - 1]$ (6) $\mathsf{g}(1,1) \to \mathsf{g}(1,0) + 1$

(2) $\mathsf{g}(0,y) \to y \; [x \leqslant 0]$ (9) $\mathsf{h}(x) \to \mathsf{g}(1,x) + 1 \; [x \leqslant 1]$

(3) $\mathsf{f}(x,0) \to \mathsf{g}(1,x) \; [x \leqslant 1]$ (11) $\mathsf{h}(x) \to \mathsf{f}(x - 1, 0) + 2 \; [x \geqslant 1]$

have been considered. Thus the system is complete according to Theorem 33 below.

▶ **Lemma 24.** *If $(\mathcal{E}, \mathcal{R}) \vdash (\mathcal{E}', \mathcal{R}')$ and the LCTRS $\mathcal{R}$ satisfies $\mathcal{R} \subseteq >_{[\cdot]}$ then also $\mathcal{R}'$ is an LCTRS such that $\mathcal{R}' \subseteq >_{[\cdot]}$.*

Proof. We show that any step $(\mathcal{E}, \mathcal{R}) \vdash (\mathcal{E}', \mathcal{R}')$ satisfies $\mathcal{R}' \setminus \mathcal{R} \subseteq >_{[\cdot]}$. If $(\mathcal{E}, \mathcal{R}) \vdash (\mathcal{E}', \mathcal{R}')$ applies **orient** or **compose** then this holds by definition. If $\mathsf{split}_\mathcal{R}$ was applied then $s >_{[\varphi \wedge \neg\psi]} t$ and $s >_{[\varphi \wedge \psi]} t$ follow from $s >_{[\varphi]} t$ by Lemma 13(4). The side condition $\mathsf{root}(s) \in \mathcal{F}_\text{terms} \setminus \mathcal{F}_\text{theory}$ in the **orient** rule ensures that $\mathcal{R}'$ is an LCTRS whenever $\mathcal{R}$ is. ◀

We next show that the conversion relation associated with $(\mathcal{E}, \mathcal{R})$ is not changed by inference steps. Note that from now on we consider conversions between *unconstrained* terms, as opposed to the rewrite steps that are performed when applying inference rules.

▶ **Lemma 25.** *If* $(\mathcal{E}, \mathcal{R}) \vdash (\mathcal{E}', \mathcal{R}')$ *then* $\xrightarrow[\mathcal{E} \cup \mathcal{R}]{} \setminus \xleftrightarrow[\mathcal{E}' \cup \mathcal{R}']{} \subseteq \xrightarrow[\mathcal{R}' \cup \mathcal{E}'>]{=} \cdot \xleftrightarrow[\mathcal{E}']{=} \cdot \xleftarrow[\mathcal{R}' \cup \mathcal{E}'>]{=}.$

Proof. We perform a case distinction on the applied inference rule.

- If compose was applied then $\langle s, t \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E}>} \langle s, u \rangle \ [\psi]$ implies $\langle t, s \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E}>} \langle u, s \rangle \ [\psi]$. If there is a step $C[s\sigma] \to C[t\sigma]$ using $s \to t \ [\varphi]$ then σ respects φ. Applying Lemma 9(1) to $C[t\sigma] \leftarrow C[s\sigma]$ yields $C[t\sigma] \to_{\mathcal{R}} \cdot \leftrightarrow_{u \approx s \ [\psi]} C[s\sigma]$ and thus $C[s\sigma] \leftrightarrow_{s \approx u \ [\psi]} \cdot_{\mathcal{R}} \leftarrow C[t\sigma]$. As $s \to u \ [\psi] \in \mathcal{R}'$ and $\mathcal{R} \subseteq \mathcal{R}'$ we have $C[s\sigma] \to_{\mathcal{R}'} \cdot_{\mathcal{R}'} \leftarrow C[t\sigma]$.
- In the case orient was applied to $s \simeq t \ [\varphi] \in \mathcal{E}$ then for a step $C[s\sigma] \leftrightarrow C[t\sigma]$ we have $C[s\sigma] \to_{s \to t \ [\varphi]} C[t\sigma]$ or $C[t\sigma]_{s \to t \ [\varphi]} \leftarrow C[s\sigma]$. The claim holds because $s \to t \ [\varphi] \in \mathcal{R}'$.
- Suppose simplify was applied with $\langle s, t \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E}>} \langle u, t \rangle \ [\psi]$. Given a step $C[s\sigma] \leftrightarrow C[t\sigma]$ using $s \approx t \ [\varphi]$, σ respects φ. By Lemma 9(1) we have $C[s\sigma] \to_{\mathcal{R}} \cdot \leftrightarrow_{u \approx t \ [\psi]} C[t\sigma]$. The claim holds because $u \approx t \ [\psi] \in \mathcal{E}'$.
- If delete was applied and there is a step $C[s\sigma] \leftrightarrow C[t\sigma]$ using the deleted equation $s \approx t \ [\varphi]$, then $\varphi\sigma$ is valid, so $s\sigma = t\sigma$ must hold.
- Suppose collapse was applied with $\langle t, s \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E}>} \langle u, s \rangle \ [\psi]$. If $C[t\sigma] \leftrightarrow C[s\sigma]$ then $C[t\sigma] \to_{\mathcal{R}} \cdot \leftrightarrow_{u \approx t \ [\psi]} C[s\sigma]$ by Lemma 9(1). As $u \approx s \ [\psi] \in \mathcal{E}'$ and $\mathcal{R} \setminus \{t \to s \ [\varphi]\} \subseteq \mathcal{R}'$ we have $C[s\sigma] \leftrightarrow_{\mathcal{E}'} \cdot_{\mathcal{R}'} \leftarrow C[t\sigma]$.
- In the case of $\mathsf{split}_{\mathcal{E}}$ and a step $C[s\sigma] \leftrightarrow C[t\sigma]$ using $s \approx t \ [\varphi]$, the substitution σ respects φ. Since $\mathcal{V}\mathsf{ar}(\psi) \subseteq \mathcal{V}\mathsf{ar}(\varphi)$, σ must respect either ψ or $\neg\psi$.
- Similarly, in the case of $\mathsf{split}_{\mathcal{R}}$ and a step $C[s\sigma] \to C[t\sigma]$ using $s \to t \ [\varphi]$, the substitution σ respects φ. Since $\mathcal{V}\mathsf{ar}(\psi) \subseteq \mathcal{V}\mathsf{ar}(\varphi)$, σ must respect either ψ or $\neg\psi$ such that one of the two new rules can be applied. ◀

▶ **Lemma 26.** *If* $(\mathcal{E}, \mathcal{R}) \vdash (\mathcal{E}', \mathcal{R}')$ *then* $\xleftrightarrow[\mathcal{E}' \cup \mathcal{R}']{} \subseteq \xleftrightarrow[\mathcal{E} \cup \mathcal{R}]{*}.$

The proof of Lemma 26 is similar to that of Lemma 25 and can be found in the appendix. In combination the last two lemmata show the invariance of the conversion relation.

▶ **Corollary 27.** *If* $(\mathcal{E}, \mathcal{R}) \vdash^* (\mathcal{E}', \mathcal{R}')$ *then the relations* $\xleftrightarrow[\mathcal{E} \cup \mathcal{R}]{*}$ *and* $\xleftrightarrow[\mathcal{E}' \cup \mathcal{R}']{*}$ *coincide.*

We now follow the proof approach in [9] by showing that whenever a term multiset dominates a conversion in $(\mathcal{E}, \mathcal{R})$ then this property is preserved by CKB steps. To this end, we compare terms with the (well-founded) relation $\succ$ defined as the lexicographic combination of $>$ and $\to_{\mathsf{calc}}^+$, using $\geqslant$ as preorder. The relation $\succeq$ denotes its reflexive closure. We label rewrite steps by multisets of terms and write $s \xrightarrow[\mathcal{R}]{S} t$ if $s \to_{\mathcal{R}} t$ and there exist terms $s', t' \in S$ such that $s' \succeq s$ and $t' \succeq t$.

▶ **Lemma 28.** *If* $\mathcal{R} \subseteq >_{[\cdot]}$ *then* $\to_{\mathcal{R}} \subseteq \succ$.

Proof. For a rule step $s \to_{\mathsf{rule}, \mathcal{R}} t$ we have $s > t$ by Lemma 14 and hence $s \succ t$. A step $s \to_{\mathsf{calc}} t$ satisfies $s \geqslant t$ by assumption and $s \succ t$ by the definition of $\succ$. ◀

▶ **Lemma 29.** *If* $(\mathcal{E}, \mathcal{R}) \vdash (\mathcal{E}', \mathcal{R}')$ *and* $\mathcal{R} \subseteq >_{[\cdot]}$ *then* $\xleftrightarrow[\mathcal{E} \cup \mathcal{R}]{S}^* \subseteq \xleftrightarrow[\mathcal{E}' \cup \mathcal{R}']{S}^*.$

Proof. We consider a single step

$$C[t\sigma] \xleftarrow[t \approx u \ [\varphi]]{S} C[u\sigma]$$

such that $t \approx u \ [\varphi] \in \mathcal{E} \cup \mathcal{R}$ and σ respects φ, and show $C[t\sigma] \xleftrightarrow[\mathcal{E}' \cup \mathcal{R}']{S}^* C[u\sigma]$. The statement of the lemma follows then by induction on the length of the conversion. According to Lemma 25 there exist terms v and w that satisfy

$$C[t\sigma] \xrightarrow[\mathcal{R}']{=} v \xrightarrow[\mathcal{E}' \cup \mathcal{R}']{=} w \xleftarrow[\mathcal{R}']{=} C[u\sigma]$$

There must be terms t' and u' in S such that $t' \succeq C[t\sigma]$ and $u' \succeq C[u\sigma]$. From the assumption $\mathcal{R} \subseteq >_{[\cdot]}$ we obtain $C[t\sigma] \succeq v$ and $C[u\sigma] \succeq w$ by Lemmata 24 and 28 and thus $t' \succeq v$ and $u' \succeq w$. Hence all (non-empty) steps between $C[t\sigma]$ and $C[u\sigma]$ can be labeled by S such that $C[t\sigma] \xleftrightarrow[\mathcal{E}' \cup \mathcal{R}']{S}^* C[u\sigma]$. ◀

We now consider a *run*, that is, a finite sequence of the form

$$\Gamma \colon (\mathcal{E}_0, \mathcal{R}_0) \vdash (\mathcal{E}_1, \mathcal{R}_1) \vdash (\mathcal{E}_2, \mathcal{R}_2) \vdash \cdots \vdash (\mathcal{E}_n, \mathcal{R}_n)$$

where $\mathcal{R}_0 \subseteq >_{[\cdot]}$ is assumed.[2] Simple induction proofs using Lemma 24, Corollary 27, and Lemma 29 extend the respective results to the final system $\mathcal{E}_n \cup \mathcal{R}_n$ of the run.

▶ **Corollary 30.** *The inclusion* $\mathcal{R}_n \subseteq >_{[\cdot]}$ *holds.*

▶ **Corollary 31.** *The relations* $\xleftrightarrow[\mathcal{E}_0 \cup \mathcal{R}_0]{}^*$ *and* $\xleftrightarrow[\mathcal{E}_n \cup \mathcal{R}_n]{}^*$ *coincide.*

▶ **Corollary 32.** *The inclusion* $\xleftrightarrow[\mathcal{E}_i \cup \mathcal{R}_i]{S}^* \subseteq \xleftrightarrow[\mathcal{E}_n \cup \mathcal{R}_n]{S}^*$ *holds.*

5.1 Standard Completion

The run Γ is *successful* if $\mathcal{E}_n = \varnothing$ and the inclusion $\mathsf{CP}(\mathcal{R}_n) \subseteq \bigcup_{i=0}^{n} \xleftrightarrow[\mathcal{E}_i]{}$ holds.

▶ **Theorem 33.** *If* Γ *is successful then* $\mathcal{R}_n$ *is a complete presentation of* $\mathcal{E}_0 \cup \mathcal{R}_0$.

Proof. From Corollary 30 we obtain $\mathcal{R}_n \subseteq >_{[\cdot]}$ and thus $\mathcal{R}_n$ is terminating by Lemma 15. In order to establish confluence, consider a peak

$$t \xleftarrow[\mathcal{R}_n]{S_1} s \xrightarrow[\mathcal{R}_n]{S_2} u$$

From $\mathcal{R}_n \subseteq >_{[\cdot]}$ and Lemma 28 we obtain $s \succ t$ and $s \succ u$. Using Lemma 20 and the definition of success, two cases are distinguished.

- If $t \downarrow_{\mathcal{R}_n} u$ then all steps in this joining sequence can be labeled with $\{t, u\}$, again using Lemma 28.
- Suppose $t \leftrightarrow_{\mathcal{E}_i} u$ for some $i \geqslant 0$. We can label this step with $\{t, u\}$ and thus obtain $t \xleftrightarrow[\mathcal{R}_n]{\{t,u\}}^* u$ from Corollary 32 since $\mathcal{E}_n$ is empty.

In both cases there is a conversion $t \xleftrightarrow[\mathcal{R}_n]{\{t,u\}}^* u$. Since $s \succ t$ and $s \succ u$ imply $S_1 \succ_{\mathsf{mul}} \{t, u\}$ and $S_2 \succ_{\mathsf{mul}} \{t, u\}$, $\mathcal{R}_n$ is peak decreasing and hence confluent by Lemma 3. ◀

---

[2] Rather than requiring the usual $\mathcal{R}_0 = \varnothing$, the more general condition $\mathcal{R}_0 \subseteq >_{[\cdot]}$ is useful when the orientation of the input equations needs to be preserved.

5.2 Ordered Completion

In this section we assume that the reduction order $>$ is complete for $\mathcal{E}_0 \cup \mathcal{R}_0$, so by Corollary 31 complete for $\mathcal{E}_n \cup \mathcal{R}_n$. We use the following modified notion of success: Γ is successful if

$$\mathsf{CP}^>(\mathcal{R}_n \cup \mathcal{E}_n) \subseteq \bigcup_{i=0}^{n} \overset{\longleftrightarrow}{\mathcal{E}_i}$$

holds. We write $\mathcal{S}_n$ for the LCTRS $\mathcal{R}_n \cup \mathcal{E}_n^>$.

▶ **Theorem 34.** *If Γ is successful then $\mathcal{S}_n$ is a ground complete presentation of $\mathcal{E}_0 \cup \mathcal{R}_0$.*

Proof. The LCTRS $\mathcal{S}_n$ is contained in $>_{[\cdot]}$ by Corollary 30 and the definition of $\mathcal{E}_n^>$, hence $\mathcal{S}_n$ is terminating by Lemma 15. For showing ground confluence we consider a ground peak

$$t \xleftarrow[\mathcal{S}_n]{S_1} s \xrightarrow[\mathcal{S}_n]{S_2} u$$

From the inclusion $\mathcal{S}_n \subseteq >_{[\cdot]}$ and Lemma 28 we obtain both $s \succ t$ and $s \succ u$. By Lemma 20 $t \downarrow_{\mathcal{S}_n} u$ or $t \leftrightarrow_{\mathsf{CP}(\mathcal{S}_n)} u$ hold. The latter in turn implies $t \downarrow_{\mathcal{S}_n} u$ or $t \leftrightarrow_{\mathsf{CP}^>(\mathcal{R}_n \cup \mathcal{E}_n)} u$ by Lemma 21. Taking the definition of success into account there are two possibilities.

- If $t \downarrow_{\mathcal{S}_n} u$ then all steps in this joining sequence can be labeled by $\{t, u\}$, using Lemma 28.
- If $t \leftrightarrow_{\mathcal{E}_i} u$ for some $i \geqslant 0$ then this step can be labeled with $\{t, u\}$ and therefore $t \xleftrightarrow{\{t,u\}}^*_{\mathcal{R}_n \cup \mathcal{E}_n} u$ is obtained from Corollary 32. Then there also exists such a conversion between t and u where all intermediate terms are ground. Since the reduction order $>$ is assumed to be complete, $v \rightarrow_{\mathcal{S}_n} w$ or $w \,_{\mathcal{S}_n}\!\!\leftarrow v$ for every step $v \leftrightarrow_{\mathcal{E}_n} w$ in this conversion. Hence $t \xleftrightarrow{\{t,u\}}^*_{\mathcal{S}_n} u$ follows.

So in both cases we obtain a conversion $t \xleftrightarrow{\{t,u\}}^*_{\mathcal{S}_n} u$. From $s \succ t$ and $s \succ u$ we obtain $S_1 \succ_{\mathsf{mul}} \{t, u\}$ and $S_2 \succ_{\mathsf{mul}} \{t, u\}$. Hence $\mathcal{S}_n$ is peak decreasing on ground terms with respect to $\succ_{\mathsf{mul}}$ and therefore ground confluent by Lemma 3. ◀

6 Implementation and Applications

We implemented the inference system CKB presented in Section 5 on top of the Ctrl tool [14] which now supports both standard Knuth-Bendix completion and ordered completion for LCTRSs. To establish termination of the resulting system, either RPO (Definition 10) with a user-specified precedence or the termination proving facilities already present in Ctrl can be used (which are rather powerful due to a DP framework for LCTRSs [12]). For the latter mode, we adapted the approach of [19] to the LCTRS setting. If desirable, the orientation of the input equations can be preserved (provided that the termination proving capabilities of Ctrl suffice, obviously). In the Ctrl infrastructure, the underlying theory can be specified by the user in a theory specification file. For common theories such as integers, bit vectors, strings, and matrices these specification files are already present. As SMT solvers we used Ctrl's internal solver and Z3 [4].

▶ **Example 35.** As one of many optimizations on the intermediate representation, LLVM provides the Instcombine pass to simplify expressions, comprising over 1000 simplification rules. In [15,16] about 500 of these rules were expressed in the domain-specific Alive language, which closely resembles constrained rewrite systems. We transformed this rule set into an

LCTRS, resulting in rules of the following shape:[3]

$$\mathsf{add}(x, x) \to \mathsf{shift\_left}(x, \#\mathsf{x01}) \tag{1}$$

$$\mathsf{add}(\mathsf{add}(\mathsf{xor}(\mathsf{or}(x, c_1), y), \#\mathsf{x01}), w) \to \mathsf{sub}(w, \mathsf{and}(x, c_2)) \ [c_1 =\sim c_2] \tag{2}$$

$$\mathsf{add}(\mathsf{xor}(x, c), z) \to \mathsf{sub}(c + z, x) \ [\mathsf{isPowerOf2}(c + \#\mathsf{x01}) \wedge \dots] \tag{3}$$

Here the set of values comprises all bit vectors of a fixed length (e.g., 8). The set of function symbols $\mathcal{F}_{\mathrm{theory}}$ adds logical, arithmetic, and shift operations on bit vectors, which allow to express auxiliary predicates like isPowerOf2. On the other hand $\mathcal{F}_{\mathrm{terms}}$ consists of symbols such as add, sub, and xor which refer to the bit vector operations in programs that get replaced in the Instcombine pass. For example, rule (1) replaces addition of two equal numbers by a left-shift by one bit. Rule (2) simplifies two consecutive additions with bitwise operations in their arguments to a subtraction, but is only applicable if the constant c_1 is the bitwise negation of the constant c_2. Also rule (3) implements some bit twiddling, checking whether a constant is a power of 2 (among other constraints). Since the optimization set is community maintained and constantly in flux, unintended interaction and overlapping patterns are not uncommon, despite the expectation of the community that there be a "canonical" form for every expression. For example rule (2) admits a critical pair with rule (1).

Completing the entire system is beyond the reach for our tool, but Ctrl completes for instance 11 optimizations for expressions rooted by an addition to a system of 15 rules, thereby eliminating some sources of nonconfluence. We maintained the orientation of the input rules, and could use the termination prover present in Ctrl.

The next example illustrates that LCTRS completion can prevail over completion of standard rewrite systems even in the absence of constraints: using a "background theory" such as integer arithmetic may admit a much more succinct presentation.

▶ **Example 36.** The tool AQL[4] performs data integration, i.e., the transformation of data from one database scheme to another, by means of a category-theoretic approach which takes advantage of (ordered) completion [18]. More precisely, it attempts to get a complete presentation for a set of ground equations describing data in the first database, plus non-ground equations describing the transformation to the second schema. A (ground) complete system can then be used to build the initial term model describing the data in the second schema, by enumerating terms and rewriting them to normal form until a fixed point is reached. The following example is taken from AQL's problem suite:

$$\mathsf{workAt}(\mathsf{mng}(e)) \approx \mathsf{workAt}(e) \quad \mathsf{workAt}(\mathsf{sec}(x)) \approx \mathsf{dep}(x) \quad \mathsf{age}_c(e) \approx \mathsf{age}(e) + \mathsf{age}(\mathsf{mng}(e))$$

$$\mathsf{first}(\mathsf{a}) \approx \texttt{"Alice"} \qquad \mathsf{first}(\mathsf{b}) \approx \texttt{"Bob"} \qquad \mathsf{first}(\mathsf{c}) \approx \texttt{"Carl"}$$

$$\mathsf{mng}(\mathsf{a}) \approx \mathsf{b} \qquad \mathsf{mng}(\mathsf{b}) \approx \mathsf{b} \qquad \mathsf{sec}(\mathsf{b}) \approx \mathsf{a}$$

$$\mathsf{workAt}(\mathsf{a}) \approx \mathsf{m} \qquad \mathsf{dname}(\mathsf{s}) \approx \texttt{"CS"} \quad \mathsf{dname}(\mathsf{m}) \approx \texttt{"Math"}$$

Here the set of values consists of integers and strings, and $\mathcal{F}_{\mathrm{theory}}$ contains integer arithmetic. The signature $\mathcal{F}_{\mathrm{terms}}$ contains symbols dep, workAt, mng, and sec to represent a database schema that describes departments and employees working therein, with relations to designate managers and secretaries. There are equations defining general relations between relations as in the first row, and many ground equations describing the actual data (the constants a,

---

[3] Full details can be found at `http://cl-informatik.uibk.ac.at/users/swinkler/lctrs_completion/`.
[4] `http://categoricaldata.net/aql.html`

b, c refer to entries in a database table). Not all equations of the latter type are shown due to reasons of space. Ctrl can easily complete this system of 22 equations to an LCTRS of 25 rules within less than a second. Input problems for AQL often relate to standard data types like integers and strings, thus it is a key advantage if, e.g., numbers and arithmetic are already present in the theory and do not need to be axiomatized explicitly.

We conclude this section with an example on ordered completion.

▶ **Example 37.** In its ordered completion mode, Ctrl can verify ground completeness of the following system describing sorting the elements in an unordered tree:

$$[\,] \,@\, xs \to xs \qquad\qquad (x : xs) \,@\, ys \to x : (xs \,@\, ys)$$
$$\mathsf{add}(x, [\,]) \to [x] \qquad\qquad \mathsf{add}(x, y : ys) \to x : (y : ys) \;[x < y]$$
$$\mathsf{add}(x, y : ys) \to y : \mathsf{add}(x, ys)) \;[x \geqslant y] \qquad\qquad \mathsf{sort}([\,]) \to [\,]$$
$$\mathsf{sort}(x : xs) \to \mathsf{add}(x, \mathsf{sort}(xs)) \qquad\qquad \mathsf{flatten}(\mathsf{L}(x)) \to [x]$$
$$\mathsf{flatten}(\mathsf{N}(x, y)) \to \mathsf{flatten}(x) \,@\, \mathsf{flatten}(y)$$
$$\mathsf{tsort}(t) \to \mathsf{sort}(\mathsf{flatten}(t)) \qquad\qquad \mathsf{N}(x, y) = \mathsf{N}(y, x)$$

where the logical constraints are expressed over the theory of integer arithmetic. Obviously, standard completion fails on this example because of the commutativity equation. For this example we used constrained RPO such that the resulting system is complete with respect to a ground-total reduction order.

7 Conclusion

In this paper we presented an abstract completion inference system for both standard and ordered completion of LCTRSs. We provide a new and succinct correctness proof. Our prototype implementation shows the potential of completion for this powerful rewriting concept also on practical examples.

A completion procedure for a different kind of constrained rewrite systems was already proposed in [8]. However, the work presented in this paper differs from this older approach in several crucial aspects. It is known [13] that LCTRSs can express systems where the version of constrained systems from [8] admits no finite presentation. Moreover, our inference system covers not only standard but also ordered completion, and we give full and novel proofs which are very concise due to the use of peak decreasingness. We also employ constrained instead of standard reduction orders, which gives a lot more flexibility to implementations and allows in particular to perform completion with termination tools. Finally, the implementation mentioned in [8] was restricted to integers and is no longer available.

For future research a variety of directions is conceivable. There are several opportunities to enhance efficiency and effectiveness of the tool, such as stronger termination techniques and critical pair criteria [2]. Theoretical results on infinite runs can shed light on the interesting case of systems generated in the limit. Finally, formalization of LCTRSs and respective completion procedures in a proof assistant would enhance reliability.

References

1 F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998. `doi:10.1017/CBO9781139172752`.

2 L. Bachmair. *Canonical Equational Proofs.* Birkhäuser, 1991.

3 L. Bachmair, N. Dershowitz, and D.A. Plaisted. Completion without failure. In H. Aït Kaci and M. Nivat, editors, *Resolution of Equations in Algebraic Structures*, volume 2: Rewriting Techniques of *Progress in Theoretical Computer Science*, pages 1–30. Academic Press, 1989.

4 L. de Moura and N. Bjørner. Z3: An efficient SMT solver. In *Proc. 14th TACAS*, volume 4963 of *LNCS*, pages 337–340, 2008. `doi:10.1007/978-3-540-78800-3_24`.

5 S. Falke and D. Kapur. Dependency pairs for rewriting with built-in numbers and semantic data structures. In *Proc. 19th RTA*, volume 5117 of *LNCS*, pages 94–109, 2008. `doi:10.1007/978-3-540-70590-1_7`.

6 C. Fuhs, J. Giesl, M. Plücker, P. Schneider-Kamp, and S. Falke. Proving termination of integer term rewriting. In *Proc. 20th RTA*, volume 5595 of *LNCS*, pages 32–47, 2009. `doi:10.1007/978-3-642-02348-4_3`.

7 C. Fuhs, C. Kop, and N. Nishida. Verifying procedural programs via constrained rewriting induction. *ACM TOCL*, 18(2):14:1–14:50, 2017. `doi:10.1145/3060143`.

8 Y. Furuichi, N. Nishida, M. Sakai, K. Kusakari, and T. Sakabe. Approach to procedural-program verification based on implicit induction of constrained term rewriting systems. *IPSJ Transactions on Programming*, 1(2):100–121, 2008. In Japanese.

9 N. Hirokawa, A. Middeldorp, and C. Sternagel. A new and formalized proof of abstract completion. In *Proc. 5th ITP*, volume 8558 of *LNCS*, pages 292–307, 2014. `doi:10.1007/978-3-319-08970-6_19`.

10 K. Hoder, Z. Khasidashvili, K. Korovin, and A. Voronkov. Preprocessing techniques for first-order clausification. In *Proc. 12th FMCAD*, pages 44–51, 2012.

11 D.E. Knuth and P. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970. `doi:10.1016/B978-0-08-012975-4`.

12 C. Kop. Termination of LCTRSs. In *Proc. 13th WST*, pages 59–63, 2013.

13 C. Kop and N. Nishida. Term rewriting with logical constraints. In *Proc. 9th FroCoS*, volume 8152 of *LNAI*, pages 343–358, 2013. Full version available at `https://www.cs.ru.nl/~cynthiakop/frocos13.pdf`. `doi:10.1007/978-3-642-40885-4_24`.

14 C. Kop and N. Nishida. Constrained Term Rewriting tooL. In *Proc. 20th LPAR*, volume 9450 of *LNAI*, pages 549–557, 2015. `doi:10.1007/978-3-662-48899-7_38`.

15 N. Lopes, D. Menendez, S. Nagarakatte, and J. Regehr. Provably correct peephole optimizations with Alive. In *Proc. 36th PLDI*, pages 22–32, 2015. `doi:10.1145/2737924.2737965`.

16 N. Lopes, D. Menendez, S. Nagarakatte, and J. Regehr. Practical verification of peephole optimizations with Alive. *Communications of the ACM*, 61(2):84–91, 2018. `doi:10.1145/3166064`.

17 A. Nadel. Bit-vector rewriting with automatic rule generation. In *Proc. 16th CAV*, pages 663–679, 2014. `doi:10.1007/978-3-319-08867-9_44`.

18 P. Schultz and R. Wisnesky. Algebraic data integration. *JFP*, 27(e24):51 pages, 2017. `doi:10.1017/S0956796817000168`.

19 I. Wehrman, A. Stump, and E.M. Westbrook. Slothrop: Knuth-Bendix completion with a modern termination checker. In *Proc. 17th RTA*, volume 4098 of *LNCS*, pages 287–296, 2006. `doi:10.1007/11805618_22`.

A Proofs

The following fact becomes useful in the sequel.

▶ **Lemma 38.** *If $\varphi \Rightarrow \psi$ is valid, $\mathcal{V}\mathsf{ar}(\psi) \subseteq \mathcal{V}\mathsf{ar}(\varphi)$, and γ respects φ then γ respects ψ.*

Proof of Lemma 8(2). Since δ respects ψ there is some δ' respecting ψ' such that $\langle t, v\rangle\delta = \langle t', v'\rangle\delta'$.

First, (1) involves a rule step then $\varphi' = \psi'$, $\langle s', u'\rangle\delta' \to_{\mathsf{rule}, 1p} \langle t', v'\rangle\delta'$, and $u'\delta' = v'\delta'$ by Lemma 6. Because $\langle s, u\rangle\ [\varphi] \sim \langle s', u'\rangle\ [\varphi']$ there is some γ such that $\langle s, u\rangle\gamma = \langle s', u'\rangle\delta'$. We thus have $s\gamma = s'\delta' \to_{\mathcal{R}} t'\delta' = t\delta$ and $u\gamma = u'\delta' = v'\delta' = v\delta$.

Next suppose (1) involves a calculation step. By the definition of $\to_{\mathsf{calc}}$ we have $\psi' = (\varphi' \wedge x = f(s_1, \ldots, s_n))$ for some $x \in \mathcal{V}$, $f \in \mathcal{F}_{\mathsf{theory}}$, and $s_1, \ldots, s_n \in \mathcal{V}\mathsf{ar}(\varphi') \cup \mathcal{V}\mathsf{al}$. Since x is fresh we have $s'\delta' \to_{\mathsf{calc}} t'\delta'$ and $u'\delta' = v'\delta'$. From $\langle s, u\rangle\ [\varphi] \sim \langle s', u'\rangle\ [\varphi']$ we obtain a substitution γ respecting φ such that $\langle s, u\rangle\gamma = \langle s', u'\rangle\delta'$. Therefore $s\gamma = s'\delta' \to_{\mathcal{R}} t'\delta' = t\delta$ and $u\gamma = u'\delta' = v'\delta' = v\delta$. ◄

Proof of Lemma 13.

1. For any γ that respects φ we have $s\gamma > t\gamma$ and $t\gamma > u\gamma$, hence $s\gamma > u\gamma$ follows from transitivity of $>$.
2. Suppose γ satisfies φ. We have to show that $(s\sigma)\gamma > (t\sigma)\gamma$ holds. By assumption σ respects φ, hence so does $\sigma\gamma$. From $s >_{[\varphi]} t$ we therefore obtain $s(\sigma\gamma) > t(\sigma\gamma)$, so also $(s\sigma)\gamma > (t\sigma)\gamma$ holds.
3. This follows from closure under contexts of $>$.
4. Any substitution γ that respects ψ also respects φ by Lemma 38, hence $s\gamma > t\gamma$ because of $s >_{[\varphi]} t$. ◄

Proof of Lemma 21. Suppose a step $C[s\tau] \leftrightarrow_{\mathsf{CP}(\mathcal{R} \cup \mathcal{E}^>)} C[t\tau]$ between ground terms $C[s\tau]$ and $C[t\tau]$ uses a critical pair $s \approx t\ [\chi]$ in $\mathsf{CP}(\mathcal{R} \cup \mathcal{E}^>)$ originating from an overlap $\langle \ell_1 \to r_1\ [\varphi_1], p, \ell_2 \to r_2\ [\varphi_2]\rangle$ with most general unifier σ. Hence we have $\ell_1 \to r_1\ [\varphi_1]$, $\ell_2 \to r_2\ [\varphi_2] \in \mathcal{R} \cup \mathcal{E}^> \cup \mathcal{R}_{\mathsf{calc}}$, $s = \ell_2\sigma[r_1\sigma]_p$, $t = r_2\sigma$, and $\chi = \varphi_1\sigma \wedge \varphi_2\sigma$. Then there are $u_1 \approx v_1\ [\psi_1]$ and $u_2 \approx v_2\ [\psi_2]$ in $\mathcal{R} \cup \mathcal{E} \cup \mathcal{R}_{\mathsf{calc}}$ and a substitution γ such that $\ell_1 \to r_1\ [\varphi_1] = (u_1 \approx v_1\ [\psi_1])\gamma$ and $\ell_2 \to r_2\ [\varphi_2] = (u_2 \approx v_2\ [\psi_2])\gamma$ (assuming that equations and rules in $\mathcal{R} \cup \mathcal{E} \cup \mathcal{R}_{\mathsf{calc}}$ are renamed apart).

We distinguish two cases. First, suppose $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(u_2)$. We have $u_2\gamma\sigma = u_2\gamma\sigma[u_1\gamma\sigma]$, so $u_2|_p$ and u_1 must be unifiable. Let ρ be their most general unifier, so there is some substitution δ such that $\gamma\sigma = \rho\delta$. Since $\chi\tau = (\varphi_1 \wedge \varphi_2)\sigma\tau = (\psi_1 \wedge \psi_2)\gamma\sigma\tau$ is valid, $(\psi_1 \wedge \psi_2)\gamma$ is satisfiable. So there is an overlap $\langle u_1 \approx v_1\ [\psi_1], p, u_2 \approx v_2\ [\psi_2]\rangle$. Moreover, since τ respects χ, the substitution $\sigma\tau$ respects $\varphi_1 = \psi_1\gamma$. If $u_1 \approx v_1\ [\psi_1] \in \mathcal{R} \cup \mathcal{E}^>$ then we have $u_1\gamma >_{[\psi_1\gamma]} v_1\gamma$ and hence $u_1\gamma\sigma\tau > v_1\gamma\sigma\tau$; if $u_1 \approx v_1\ [\psi_1] \in \mathcal{R}_{\mathsf{calc}}$ then $u_1\gamma\sigma\tau \geqslant v_1\gamma\sigma\tau$ by the properties of a reduction pair. In either case $v_1\gamma >_{[\psi_1\gamma]} u_1\gamma$ cannot hold because $>$ is well-founded. Similarly, $v_2\gamma >_{[\psi_2\gamma]} u_2\gamma$ cannot hold. So the overlap gives rise to a constrained extended critical pair $u_2\rho[v_1\rho]_p \approx v_2\rho\ [\psi_1\gamma \wedge \psi_2\gamma]$, and we have a step

$$C[s\tau] = C[u_2[v_1]_p\gamma\sigma\tau] = C[u_2\rho[v_1\rho]_p\delta\tau] \xleftarrow{\quad\mathsf{CP}^>(\mathcal{R} \cup \mathcal{E})\quad} C[v_2\rho\delta\tau] = C[v_2\gamma\sigma\tau] = C[t\tau]$$

Second, if $p \notin \mathcal{P}\mathsf{os}_{\mathcal{F}}(u_2)$ then the peak $u_2\gamma[v_1\gamma]_p \leftarrow u_2\gamma[u_1\gamma]_p = u_2\gamma \to v_2\gamma$ forms a variable overlap between $u_1 \approx v_1\ [\psi_1]$ and $u_2 \approx v_2\ [\psi_2]$. There are positions p' and q such that $u_2|_{p'}$ is some variable x and $\gamma(x)|_q = u_1\gamma$. Note that $u_2 \approx v_2\ [\psi_2]$ cannot be a calculation rule since $\gamma(x) \notin \mathcal{V}\mathsf{al}$. Let γ' be the substitution defined by $\gamma'(x) = \gamma(x)[v_1\gamma]_q$ and $\gamma'(y) = \gamma(y)$

for all $y \neq x$. Then x cannot occur in ψ_2 since $\gamma(y) \in \mathcal{V}\mathsf{al}$ for all $y \in \mathcal{V}\mathsf{ar}(\psi_2)$, but left-hand sides are headed by non-theory symbols. Therefore γ' respects ψ_2 and thus we obtain

$$u_2\gamma[v_1\gamma]_p \xrightarrow[\ell_1 \to r_1 \ [\varphi_1]]{*} u_2\gamma' \xleftrightarrow[u_2 \approx v_2 \ [\psi_2]]{} v_2\gamma' \xleftarrow[\ell_1 \to r_1 \ [\varphi_1]]{*} v_2\gamma$$

Since $u_2\gamma$ and $v_2\gamma$ are ground, $u_2 \approx v_2 \ [\psi_2] \in \mathcal{R} \cup \mathcal{E}^\pm$ and $>$ is complete for $\mathcal{R} \cup \mathcal{E}$, either $u_2\gamma' \to_{\mathcal{R} \cup \mathcal{E} >} v_2\gamma'$ or $v_2\gamma' \to_{\mathcal{R} \cup \mathcal{E} >} u_2\gamma'$ must hold. ◀

Proof of Lemma 26. We perform a case distinction on the applied inference rule.

- Suppose deduce was applied and there is a step $C[s\sigma] \leftrightarrow C[t\sigma]$ using $s \approx t \ [\varphi] \in \mathcal{E}'$. The substitution σ must respect φ. From $\langle u, u \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E}^\pm} \langle s, u \rangle \ [\varphi]$ and Lemma 8(2) we obtain a substitution γ_1 such that $u\gamma_1 \to_{\mathcal{R} \cup \mathcal{E}^\pm} s\sigma$ and $u\gamma_1 = u\sigma$, and from $\langle u, u \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E}^\pm} \langle u, t \rangle \ [\varphi]$ a substitution γ_2 such that $u\gamma_2 \to_{\mathcal{R} \cup \mathcal{E}^\pm} t\sigma$ and $u\gamma_2 = u\sigma$. We have

  $$C[s\sigma] \ _{\mathcal{R} \cup \mathcal{E}^\pm}\!\leftarrow C[u\gamma_1] = C[u\sigma] = C[u\gamma_2] \to_{\mathcal{R} \cup \mathcal{E}^\pm} C[t\sigma]$$

  and thus $C[s\sigma] \leftrightarrow^*_{\mathcal{E} \cup \mathcal{R}} C[t\sigma]$.
- If compose was applied then $\langle s, t \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E} >} \langle s, u \rangle \ [\psi]$ and hence $\langle t, s \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E} >} \langle u, s \rangle \ [\psi]$. If there is a step $C[u\sigma] \leftarrow C[s\sigma]$ then $C[s\sigma] \leftrightarrow^2_{\mathcal{E} \cup \mathcal{R}} C[u\sigma]$ by Lemma 9(2).
- Next, suppose simplify was applied so we have $\langle s, t \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E} >} \langle u, t \rangle \ [\psi]$. For a step $C[s\sigma] \leftrightarrow C[u\sigma]$ using $s \approx u \ [\psi]$ we thus have $C[s\sigma] \leftrightarrow^2_{\mathcal{E} \cup \mathcal{R}} C[u\sigma]$ by Lemma 9(2).
- If collapse was applied we have $\langle t, s \rangle \ [\varphi] \to_{\mathcal{R} \cup \mathcal{E} >} \langle u, s \rangle \ [\psi]$. If there is a step $C[u\sigma] \leftrightarrow C[s\sigma]$ using $u \approx s \ [\psi]$ then $C[s\sigma] \leftrightarrow^2_{\mathcal{E} \cup \mathcal{R}} C[u\sigma]$ follows again from Lemma 9(2).
- The case of orient is trivial, and in the case of delete there is nothing to show. Also the split cases are easy since substitutions respecting $\varphi \wedge \psi$ (or $\varphi \wedge \neg\psi$) also respect φ. ◀