Electronic Notes in Theoretical Computer Science 59 No. 4 (2001)
URL: http://www.elsevier.nl/locate/entcs/volume59.html 17 pages

Knuth-Bendix Completion for Non-Symmetric Transitive Relations

Georg Struth [1]

Institut für Informatik
Albert-Ludwigs-Universität Freiburg
Georges-Köhler-Allee 52
D-79110 Freiburg i. Br., Germany

Abstract

We extend the Knuth-Bendix completion procedure from equational rewriting to rewriting with non-symmetric transitive relations and quasi-orderings. The main differences are the following: Specification of the general non-ground case seems beyond first-order logic. It is within that realm when terms are linear or functions non-monotonic. The procedure requires critical-pair computations and need not terminate even in the ground case. Simplification is not don't care non-deterministic, but search based. Applications include ordered resolution and ordered chaining calculi, development of rule-based declarative procedures and algorithms, program and reachability analysis (in rewriting logic) and propagation of inequality constraints.

1 Introduction

Rewriting usually focuses on equational logic and term normal forms. In this context it yields solutions to word problems in certain free and finitely presented algebras via canonical term rewrite systems. The Knuth-Bendix procedure constructs such canonical systems incrementally from equational specifications. Equational rewriting and completion have been successfully integrated into automatied first- and interactive higher-order deductive systems.

Our main contribution is the specification, proof of correctness and discussion of Knuth-Bendix completion procedures beyond equations for non-symmetric transitive relations and quasiorderings. Part of this extension requires only slight modifications of the equational case. But there are also important differences. Specification of the general non-ground case seems beyond first-order logic. It depends on context variables and involves the context unification problem [7,13], whose decidability is open. Non-symmetric completion

[1] struth@informatik.uni-freiburg.de

is within first-order logic, when terms are linear or functions non-monotonic. The procedure requires critical-pair computations and need not terminate even in the ground case. Simplification is not don't care non-deterministic, but search based.

Non-symmetric rewriting and completion are not only interesting as generalizations of the equational case. We believe that a main application lies in rule-based specification and declarative programming. Completion procedures themselves can be described abstractly and declaratively in a rule-based way [2]. More interestingly here, they provide generic metaprocedures for the development of declarative procedures and algorithms. In fact, the Knuth-Bendix procedure from universal algebra, the Gröbner base algorithms from computer algebra and the ordered resolution procedure from computational logic can be derived as instances or refinements of general completion procedures [14,6,17]. In particular, the resolution rule itself is a non-symmetric critical pair computation for distributive lattices. Non-symmetric completion is also implicit in ordered chaining calculi for transitive relations and quasiorderings [3,18]. Moreover, it is a main ingredient of some recent declarative congruence closure algorithms [10,5]. In all these cases, the declarative approach yields simple, uniform and readable specifications, since all procedures are developed from the general completion mechanism of critical pair computation and simplification. Moreover, since the main data structures are hidden in the metaprocedure, efficient algorithms arise in a rather abstract and straightforward way.

We sketch an example of using non-symmetric completion for a rule-based dynamic algorithm to collapse cycles in digraphs or to construct the associated partial ordering from a given quasiordering. This, and non-symmetric completion in general, is useful for the reachability analysis of state transition systems. Moreover, also the analysis of execution sequences in rewriting logic [12] seems interesting. There, rewriting is a model of (concurrent) computation. Like non-symmetric rewriting, it goes beyond properties like confluence or termination, but does not use any procedural control provided by syntactic reduction orderings.

The remainder of the text is organized as follows. Section 2 contains some preliminary definitions. Section 3 sketches the basics of rewriting with non-symmetric transitive relations and quasiorderings. Section 4 specifies a ground Knuth-Bendix completion procedure for transitive relations and proves its correctness. Section 5 further discusses the procedure. Section 6 extends the procedure to quasiorderings. Section 7 discusses its lifting to the non-ground case. Section 8 sketches a simple application. Section 9 contains a conclusion.

2 Preliminaries

Let $T_\Sigma(V)$ be a set of terms with signature Σ and variables in V. We write T_Σ, if $V = \emptyset$, that is all terms are *ground*. We identify terms with $\Sigma \cup V$-labeled trees with nodes or *positions* in the monoid $\mathbb{N}^*$. ϵ denotes the root and ni the i-th child of node n. A variable is *linear* (*non-linear*) in a term, if it labels exactly one node (at least two nodes). A *substitution* is a mapping $\sigma : V \to T_\Sigma(V)$. We identify it with its homomorphic extension to $T_\Sigma(V)$. For a term t and a substitution σ, a node is a *skeleton position* of $t\sigma$, if it has a label from Σ in t. It is a *variable instance position* of $t\sigma$, if one of its ancestors has a label from V in t. $s[t]_p$ ambiguously denotes both the replacement of the subterm $s|_p$ of s at position p by t and the occurrence of t as a subterm of s at position p.

A binary relation $<$ on a set A is a *quasiordering*, if it is reflexive and transitive, a *partial ordering*, if it is also antisymmetric and an *equivalence*, if it is reflexive, transitive and symmetric. In our context, the differences between quasiorderings and partial orderings are negligible. Every quasiordering $<$ naturally induces a partial ordering. The relation $\sim = < \cap >$ is an equivalence and the relation $\sqsubset$ on $A/\sim$ defined by $[x] \sqsubset [y]$ iff $x < y$ for all $x, y \in A$ is a partial ordering. When $<$ is only transitive, $\sim$ can be understood as a partial equivalence. $\sqsubset$ is a partial ordering that is only partially reflexive.

We investigate in particular transitive relations and quasiorderings on a set of terms $T_\Sigma(V)$ and associated term algebra A. The operation f^A denoted by the n-ary operation symbol f is *monotone* (in each argument), if $s_1 < t_1, \dots, s_n < t_n$ implies $f^A(s_1, \dots, s_n) < f^A(t_1, \dots, t_n)$, for all $s_1, \dots, s_n$ and $t_1, \dots, t_n$ in A. The relation $<$ is *compatible*, if every operation is monotone. This is the case iff $s < t$ implies $r[s]_p < r[t]_p$ for all $r, s, t \in T_\Sigma(V)$ and positions p. $<$ is *stable*, if $s < t$ implies $s\sigma < t\sigma$ for all $s, t \in T_\Sigma(X)$ and endomorphisms σ of A and *fully invariant* or a *rewrite relation*, if it is compatible and stable.

The above definitions for computing with fully invariant non-symmetric relations are also relevant to term rewriting and rewriting logic [12]. The only difference to rewriting logic is the use of an additional syntactic ordering on terms, usually a *reduction ordering*, that is a well-founded fully invariant quasiordering.

3 Non-Symmetric Rewriting

We presuppose knowledge of equational rewriting and completion. See [9,2,1] for further information.

In equational rewriting, a reduction ordering is used to turn an equational specification into a set of rewrite rules. When rewriting with a non-symmetric transitive relation, one can usually at most obtain a partition into a set of increasing rules and a set of decreasing rules. The theory of rewriting can be split into an abstract algebraic part dealing with semi-commutation properties of relations and a concrete part dealing with the (tree-) structure of terms. We first consider the abstract part. Let R^+ and R^* denote the transitive and reflexive transitive closure of a relation R. An *R-proof* of a pair (s, t) is a finite sequence of R-steps leading from s to t. Pairs (s_{i-1}, s_i) in a proof are called *proof steps.*

Lemma 3.1 *Let R and S be binary relations on some set A.*

(i) $S^*R^* \subseteq R^*S^*$ *iff* $(R \cup S)^* \subseteq R^*S^*$.

(ii) $SR \subseteq R^*S^*$ *implies* $(R \cup S)^* \subseteq R^*S^*$, *if* $(R \cup S^{-1})$ *is well-founded.*

Lemma 3.1 (i) expresses a generalized Church-Rosser property, lemma 3.1 (ii) a generalized Newman's lemma that is appropriate for quasiorderings. See [11,15,16] for proofs. Similar properties hold for transitive relations.

Lemma 3.2 *Let R and S be binary relations on some set A.*

(i) $SR^+ \subseteq R^+S^* \cup S^+$ *iff* $(R \cup S)^+ \subseteq R^+S^* \cup S^+$.

(ii) $SR \subseteq R^+S^* \cup S^+$ *implies* $(R \cup S)^+ \subseteq R^+S^* \cup S^+$, *if* $(R \cup S^{-1})$ *is well-founded.*

We call proofs in R^*S^* or $R^+S^* \cup S^+$, respectively, *rewrite proofs.* Those of the first kind are appropriate for quasiorderings, those of the second king for transitive relations. However, there is no concept of term normal form for non-symmetric rewriting. Lemma 3.1 and lemma 3.2 extend to non-symmetric rewriting modulo a relation E [11,15,16]. Here we only consider the two-relations theory.

The abstract results in lemma 3.1 and lemma 3.2 can be refined to critical pair lemmata at the term level. They introduce drastic differences to the equational case, as has first been noted in [11]. We assume two rewrite relations $\to_R$ and $\to_S$ induced by the sets R and S of rewrite rules: A term s rewrites to a term t in one step at position p, written as $s \to_R^p t$ or $s \to_S^p t$, if there is some rewrite rule $l \to_R r \in R$ or $l \to_S r \in S$ such that $s|_p = l\sigma$ for some position p and substitution σ and $t = s[r\sigma]_p$. We consider the sequential application of rewrite rules with respect to the term structure. The application of a rewrite rule $l \to_R r$ at position p of a term by substitution σ is a partial operation $(l \to_R r)_p^\sigma : T_\Sigma(V) \to T_\Sigma(V)$[2]. $(l \to_R r)_p^\sigma$ is obviously injective; thus its inverse exist. We write $(l \to_R r)_{p_1 \ldots p_n}^{\sigma_1 \ldots \sigma_n}$ to denote (simultaneous) application of the rule at disjoint positions (incomparable under the prefix ordering). We compose application of functions like for relations: $(fg)(x) = g(f(x))$.

Proposition 3.3 *Let R and S be two sets of rewrite rules.*

(i) Let p and q be disjoint positions. Then

$$(l_1 \to_R r_1)_p (l_2 \to_S r_2)_q = (l_2 \to_S r_2)_q (l_1 \to_R r_1)_p.$$

[2] We omit the substitution index, when the substitution is arbitrary.

(ii) *Let $p_1^l, \dots, p_m^l$ and $p_1^r, \dots, p_n^r$ denote all positions in the left-hand side l_1 and right-hand side r_1 of some rule $l_1 \to_R r_1$, where some variable x occurs. Let ρ and σ be substitutions. Then for all positions q,*

$$(l_1 \to_R r_1)_p^\rho (l_2 \to_S r_2)^\sigma_{(pp_1^r q)\dots(pp_n^r q)} = (l_2 \to_S r_2)^\sigma_{(pp_1^l q)\dots(pp_m^l q)} (l_1 \to_R r_1)_p^\rho.$$

The positions pp_iq are variable instance positions. The same substitution σ is applied to all instances of the rule $l_2 \to_S r_2$.

(iii) *There are rules $l_1 \to_R r_1$, $l_2 \to_S r_2$ and positions p and $q = pq'$, where q' is a skeleton position in r_1 such that*

$$(l_1 \to_R r_1)_p (l_2 \to_S r_2)_q \neq (l_2 \to_S r_2)_q (l_1 \to_R r_1)_p.$$

Similar facts hold for skeleton positions in l_1 etc.

See [15,16] for a proof.

Corollary 3.4 *Under the same conditions as in proposition 3.3 (ii), let x occur linearly in r_1, at position p_1. Then for all positions q,*

$$(l_1 \to_R r_1)_p^\rho (l_2 \to_S r_2)^\sigma_{(pp_1^r q)} = (l_2 \to_S r_2)^\sigma_{(pp_1^l q)\dots(pp_m^l q)} (l_1 \to_R r_1)_p^\rho.$$

Combining the term based results in proposition 3.3 and corollary 3.4 with the abstract ones in lemma 3.1 and lemma 3.2, it appears that semi-commutations of rewrite steps involving variable instance positions are only compatible with the abstract inclusions, iff the respective variable is linear. In other words, variable critical pairs for non-linear variables cannot be avoided. In order to handle variable critical pairs, we extend our concept of term to include contexts. We use a set V_1 of first-order variables $x_1, x_2, \dots$ and a set V_2 of context variables $X_1, X_2, \dots$. The inductive definition of terms is extended such that also $X(t)$ is a term, if t is a term and X a context variable. A context $C[\cdot]$ can then be inductively defined as an (extended) term containing exactly one hole $[\cdot]$. See [13] for a concise definition. $C[t]$ means that t is inserted into the context $C[\cdot]$. A ground context is a context without context variables. We also extend substitutions to replace context variables by contexts. The context unification problem deals with finding a ground substitution that makes a set of terms (with contexts) equal. Its decidability is open.

In analogy to the equational case, critical pairs are computed by the Knuth-Bendix procedure, when the inclusion in lemma 3.1 or lemma 3.2 is not guaranteed.

Example 3.5 Take the sets of rewrite rules $R = \{x + x \to_R x\}$ and $S = \{a \to_S b\}$. The SR-sequence $b + a \to_S b + b \to_R b$ cannot be replaced by a rewrite proof. Only a "backward" step, leading to $a+a \to_S b+a \to_S b+b \to_R b$ would yield a rewrite proof $a + a \to_R a \to_S b$. The situation is shown in the following diagram.

$$\begin{tikzcd}
 & & b+b \arrow[dr, "R"] & \\
 & b+a \arrow[ur, "S"] & & b \\
a+a \arrow[ur, dashed, "S"] \arrow[dr, "R"'] & & & \\
 & a \arrow[uurr, "S"'] & &
\end{tikzcd}$$

Let $l_1 \to_R r_1$ and $l_2 \to_S r_2$ be two rewrite rules. A *skeleton critical pair* is a pair of the form

$$(l_2\sigma, r_2\sigma[r_1\sigma]_p) \quad \text{or} \quad (l_1\sigma[l_2\sigma]_p, r_1\sigma),$$

where p is a skeleton position of r_2 and l_1, respectively, and where σ a most general unifier of $r_2|_p$ and l_1 or $l_1|_p$ and r_2, respectively. A *variable critical pair* is a pair of the form

$$(l_2\sigma, r_2\sigma[X(r_1)]_p) \quad \text{or} \quad (l_1\sigma[X(l_2)]_p, r_1\sigma),$$

where p is a variable instance position of a variable x, which in the first case occurs non-linear in r_2 and in the second case non-linear in l_1 and where σ maps x to $X(l_1)$ or to $X(r_2)$, respectively. The variable critical pair arising from example 3.5, for instance, is $(X(a) + X(b), X(b))$.

Without critical pairs and when R and S^{-1} are well-founded, there is the following search procedure (semi-decision procedure) for the reachability problem for a transitive relation. $s < t$, if either (s, t) is in the diagonal of $(R \cup S)^+$, or the dags spanned by R and S^{-1} from s and t, respectively, have a common vertex. By well-foundedness, all branches in the dags have finite length. If moreover the dag is finitely branching, we obtain a decision procedure. The decision procedure for quasiorderings is similar. At the term level, one usually considers reduction orderings containing the rules in R and S^{-1}. A pair (R, S) of rewrite rules enforcing such a procedure is called a *normal system*. It has by definition the property that all critical pairs can be joined by a rewrite proof, R and S^{-1} are well-founded and no rule from R or S can be deleted.

4 Ground Completion for Transitive Relations

Knuth-Bendix completion for transitive relations has many similarities to the equational case, but there are also important and interesting differences.

As usual, a completion procedure is a program that implements a state transition system together with a syntactic reduction ordering $\prec$ on terms, (oriented) inequalities and proofs. States are tuples of sets of inequalities or rewrite rules. The transition relation is presented as a set of transition rules of two kinds. First, there are deductive inference rules that add certain consequences to a state. Second, there are simplification rules that combine deduction steps with deletions implementing an (approximative) notion of redundancy: An (oriented) inequality is *redundant*, if it can be replaced by a smaller proof in $(I \cup R \cup S)$, using the proof ordering and measure $\prec$.

In case of a transitive relation, states are of the form

$$(I, \Delta, R, S),$$

where I and Δ are sets of inequalities and R and S sets of (decreasing and increasing) rewrite rules. In the initial state, R and S are empty, I is irreflexive and Δ contains at most inequalities of the form $t < t$. The set of transition rules is denoted by C_t. A *run* of the procedure is a (finite or infinite) sequence $q_0, q_1, q_2, \ldots$ of states such that q_0 is an initial state and between all consecutive states there is a transition that applies some rule in C_t. We also define *limit states* $(I_\infty, \Delta_\infty, R_\infty, S_\infty)$, where

$$I_\infty = \bigcup_{i=0}^{\infty}(\bigcap_{j=i}^{\infty} I_j), \quad \Delta_\infty = \bigcup_{i=0}^{\infty} \Delta_i, \quad R_\infty = \bigcup_{i=0}^{\infty}(\bigcap_{j=i}^{\infty} R_j), \quad S_\infty = \bigcup_{i=0}^{\infty}(\bigcap_{j=i}^{\infty} S_j).$$

We say that a run *fails*, if $I_\infty \neq \emptyset$. It *succeeds*, if it does not fail and R_∞ and S_∞ yield a normal system. A run is *fair*, if every enabled transition is eventually executed. A completion procedure is *correct*, if every run that does not fail succeeds and $R_\infty \cup S_\infty$ is equivalent to I_0.

The deduction and simplification rules in C_t are defined as follows.

$$\frac{(I, \Delta, R, S)}{I \cup \{s < t\}, \Delta, R, S)}, \qquad \text{(Deduce)}$$

if (s, t) is a skeleton critical pair (variable instance critical pairs do not exist in the ground case). This rule can also be written as a pair of inference rules on inequalities.

$$\frac{l_2 \rightarrow_S r_2 \qquad l_1[r_2]_p \rightarrow_R r_1}{l_1[l_2]_p < r_1}, \qquad \frac{l_2 \rightarrow_S r_2[l_1]_p \qquad l_1 \rightarrow_R r_1}{l_2 < r_2[r_1]_p}.$$

$$\frac{(I \cup \{s < t\}, \Delta, R, S)}{(I, \Delta, R \cup \{s \rightarrow_R t\}, S)}, \qquad \frac{(I \cup \{t < s\}, \Delta, R, S)}{(I, \Delta, R, S \cup \{t \rightarrow_S s\})}, \qquad \text{(Orient)}$$

if $s \succ t$.

$$\frac{(I \cup \{s < t\}, \Delta, R, S)}{(I, \Delta, R, S)}, \quad \frac{(I, \Delta, R \cup \{s \rightarrow_R t\}, S)}{(I, \Delta, R, S)}, \quad \frac{(I, \Delta, R, S \cup \{s \rightarrow_S t\})}{(I, \Delta, R, S)}, \qquad \text{(Simplify)}$$

if $s < t$, $s \rightarrow_R t$ or $s \rightarrow_S t$ is redundant.

$$\frac{(I \cup \{s < s\}, \Delta, R, S)}{(I, \Delta \cup \{s < s\}, R, S)}, \qquad \text{(Diagonal)}$$

Let M be the set of finite multisets over T_Σ. Let $S = M \times T_\Sigma \times T_\Sigma$. Let $\prec$ be some reduction ordering on T_Σ and $\lhd$ the (strict) subterm ordering[3]. We compare tuples in S by a proof ordering $\prec'$ which is the lexicographic combination of the multiset extension of $\prec$ (also denoted by $\prec$) for the first component, $\lhd$ for the second component and $\prec$ for the third component. A proof step measure is a mapping $\mu : T_\Sigma \times T_\Sigma \to S$ defined for a proof step $w = (s_{i-1}, s_i)$ by[4]

$$\mu : w \mapsto \begin{cases} (\{s_{i-1}, S_i\}, \_, \_), & \text{if } s_{i-1} \to_I s_i, \text{ (\_ denotes an arbitrary term)}, \\ (\{s_{i-1}\}, l, s_i), & \text{if } s_{i-1} \to_R s_i, \text{ by rule } l \to r, \\ (\{s_i\}, \_, \_), & \text{if } s_{i-1} \to_\Delta s_i, \\ (\{s_i\}, l, s_{i-1}), & \text{if } s_{i-1} \to_S s_i, \text{ by rule } l \to r. \end{cases}$$

Proofs are measured as multisets of proof steps, taking the multiset extension $\prec''$ of $\prec'$. Both orderings inherit wellfoundedness from $\prec$. We will denote them all by $\prec$ if no confusion may arise.

We can now prove the following theorem analogously to the equational case.

Theorem 4.1 *(i) The rules of C_t except* Deduce *are sound simplification rules with respect to the proof ordering $\prec$.*

(ii) The proof ordering $\prec$ induces termination of the proof transformation relation corresponding to the rules of C_t.

(iii) C_t is correct, if it is fair. The limit state of a fair successful non-symmetric ground completion process is a normal system.

Proof We only give a sketch. A complete proof can be found in [16].

(ad i) the Simplify rules are simplification rules by definition, since a redundant rule is deleted. For the Orient rules we argue as follows. Consider the inequality $s < t$ with $t \prec s$. Then the left-hand rule can be derived as

$$\cfrac{\cfrac{(I \cup \{s < t\}, \Delta, R, S)}{(I \cup \{s < t\}, \Delta, R \cup \{s \longrightarrow_R t\}, S)}}{(I, \Delta, R \cup \{s \longrightarrow_R t\}, S)}$$

The first step is a deduction inference, since addition of a rule to an already present inequality is sound, provided the inequality can be oriented. The second step is a deletion step, since $\mu(s < t) \succ \mu(s \longrightarrow_R t)$ by the first component.

(ad ii) For Orient and Simplify, this follows immediately from (i). For Deduce, a similar argument can be applied, also using the critical pair lemma

[3] In the non-ground case, we use the (strict) encompassment ordering $\lessdot$ instead. s is encompassed by t, if some subterm of t is an instance of s.

[4] Abusing notation we also write $\mu(s < t)$ or $\mu(s \to_R t)$.

and essentially the first component of the measure. Therefore all inference rules are antimonotonic according to $\prec$.

(ad iii) By induction on the run. If some proof is not a rewrite proof, then it must contain a SR-proof or an unordered step that is reducible. Otherwise, if some proof contains a step that is later deleted, then it can be replaced by a smaller proof, by induction a rewrite proof. □

5 Discussion

There are three main differences between equational and non-symmetric completion. First, even in the ground case, Deduce is not subsumed by Simplify. Second, the Simplify rules are based on search and cannot, as in the equational case, be refined as one-step rules. Third, even in the ground case, the procedure need not terminate. We now further discuss all three phenomena.

Let us first consider the proof patterns associated with the equational Deduce rule at the left-hand side and the Collapse-rule at the right-hand side (c.f. [2]) of the following diagram.

$$\begin{tikzcd}
 & l_1[l_2] \arrow[dl, "\mathrm{R}"'] \arrow[dr, "\mathrm{R}"] & & \Longrightarrow & & l_1[l_2] \arrow[dl, "\mathrm{R}"'] \arrow[dr, "\mathrm{R}"] & \\
r_1 & & l_1[r_2] & & r_1 \arrow[rr, leftrightarrow, "\mathrm{E}"'] & & l_1[r_2]
\end{tikzcd}
\qquad
\begin{tikzcd}
 & l_1[l_2] \arrow[dl, "R"'] \arrow[dr, "R"] & & \Longrightarrow & & l_1[l_2] \arrow[dr, "R"] & \\
r_1 & & l_1[r_2] & & r_1 \arrow[rr, leftrightarrow, "E"'] & & l_1[r_2]
\end{tikzcd}$$

Collapse has an additional proviso with respect to encompassment, which trivially holds in the ground case. Then, equational Deduce reduces to Collapse and becomes a simplification step. In the non-ground case this does not hold. Take, for instance, the conclusion of the Deduce inference of $f(f(x)) \longrightarrow a$ with itself. It is $f(a) \approx a$. Here, the encompassment condition is not satisfied and the step is not a simplification by Collapse. In the non-symmetric case, the proof patterns corresponding to Deduce and Simplify are

$$\begin{tikzcd}
 & l_1[r_2] \arrow[dr, "R"] & & \Longrightarrow & & l_1[r_2] \arrow[dr, "R"] & \\
l_1[l_2] \arrow[ur, "S"] & & r_1 & & l_1[l_2] \arrow[ur, "S"] \arrow[rr, "I"'] & & r_1
\end{tikzcd}
\qquad
\begin{tikzcd}
s \arrow[r, "R"'] & t & \Longrightarrow & s \arrow[r, bend right, "(I \cup R \cup S)^+"'] & t
\end{tikzcd}$$

and their duals. The first transformation cannot be reduced to the second one. Due to lack of symmetry, the arrow of S in the diagram for Deduce has the converse direction of those in $(I \cup R \cup S)^+$. This observation is interesting for ordered resolution as a specialization of non-symmetric Knuth-Bendix completion to distributive lattices [17]. The ground resolution rule is a variant of ground Deduce, hence not a simplification rule.

Let us now consider the proof patterns of the equational simplification rules Simplify, Compose and Collapse in the following diagrams.

$$\begin{tikzcd}
s[l\sigma] \arrow[dr, "R"'] \arrow[rr, leftrightarrow, "E"] & & t & \Longrightarrow & s[l\sigma] \arrow[dr, "R"'] & & t \arrow[dl, leftrightarrow, "E"] \\
& s[r\sigma] & & & & s[r\sigma] &
\end{tikzcd}
\qquad
\begin{tikzcd}
l_1 \arrow[dr, "R"'] & & \Longrightarrow & l_1 \arrow[ddr, "R"'] & \\
& r_1[l_2\sigma] \arrow[d, "R"] & & & r_1[l_2\sigma] \arrow[d, "R"] \\
& r_1[r_2\sigma] & & & r_1[r_2\sigma]
\end{tikzcd}$$

$$\begin{tikzcd}
& l_1[l_2\sigma] \arrow[dl, "R"'] \arrow[dr, "R"] & & \Longrightarrow & l_1[l_2\sigma] \arrow[dr, "R"] & \\
r_1 & & l_1[r_2\sigma] & & r_1 \arrow[r, leftrightarrow, "E"'] & l_1[r_2\sigma]
\end{tikzcd}$$

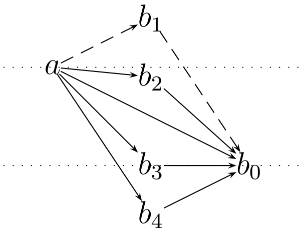
Obviously, all replacements of E and R by I, R and S are unsound; they change the theory. Of course this is due to lack of symmetry. Therefore, in non-symmetric completion, search for redundant proofs cannot be further refined like in the equational case. This is another interesting similarity to ordered resolution, where elimination of redundant rules is search-based. It is again due to the correspondence between resolution and completion for distributive lattices.

Let us now consider termination. We have just seen that the Deduce rule is indispensable even in the ground case. The following example shows non-termination of ground non-symmetric completion.

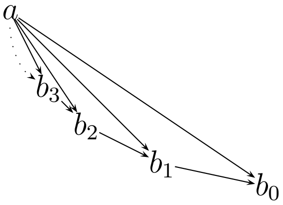
Example 5.1 Let $b \succ f \succ a$ be a precedence for constants a, b and function f of arity one that is extended to a reduction ordering containing the subterm ordering. The relations $f(b) < b$ and $f(a) < b$ are oriented as $f(b) \longrightarrow_R b$ and $f(a) \longrightarrow_S b$. From the first rule, inequalities $f^n(a) < b$ can be computed by Deduce for arbitrary n. The second rule is irredundant at all stages of the process and can never be deleted.

Returning to our analogy with ordered ground resolution, we see that termination of ground resolution cannot be explained from non-symmetric completion. There, it is mainly caused by the fact that (translated to lattice theory) join and meet are idempotent operations and thereby the number of non-equivalent ground lattice terms in a distributive lattice can be bounded.

Non-symmetric completion has the following drawback. The simplification mechanisms (in combination with the reduction ordering) are too weak to cover some natural cases of redundancy that arise in many applications (c.f. [16]). Consider the inequality $a < b_0$ in the following diagram.



According to the proof measure of non-symmetric completion, only $a < b_4 < b_0$ simplifies $a < b_0$, whereas even the chain $a < b_3 < b_0$ does not. However it appears only natural to count all chains except $a < b_1 < b_0$ as simplifications, since they do not contain a SR-proof and yield a finer presentation of the coarse step $a < b_0$. With the standard proof measures, however, there seems no simple way to strengthen the concept of simplification. Consider the sequence $a < b_1 < b_0$ in in the following diagram and assume that $a < b_1$ is again simplifiable by a sequence $a < b_2 < b_1$ such that $a \succ b_2 \succ b_1$. Iterating this argument yields an infinite sequence $b_1 \prec b_2 \prec b_3 \prec \ldots$ with upper bound a which does not contradict well-foundedness of $\prec$.



Of course with simplification chains in the sense of non-symmetric rewriting this case cannot occur. By König's lemma, the case depicted in the diagram—an infinitely branching node—is the only way such a divergence may arise. However, for regular term rewrite systems, where no fresh variables may occur in the smaller term of a rule, there are no infinite branches, since rewriting can only occur at finitely many positions in a term and when there are no fresh variables also the possibilities to instantiate are finitely bounded. As the above example shows, situations like this can however arise by critical pair computations.

The non-symmetric simplification facilities can be augmented by allowing the replacement of a rule by a finer rearrangement consisting solely of rules which belong to the same set as the rule to be discarded. For a regular non-symmetric term rewrite system, a rule $l \longrightarrow_R r$ can be *refined*, if there exists a proof $l \longrightarrow_R^+ r$ of length ≥ 2. The definition for S is dual. Proofs refining a rule are called *refinements* or *finer proofs*. This can be modeled by the deletion rules

$$\frac{(I, \Delta, R \cup \{s \rightarrow_R t\}, S)}{(I, \Delta, R, S)}, \qquad \frac{(I, \Delta, R, S \cup \{s \rightarrow_S t\})}{(I, \Delta, R, S)}, \qquad \text{(Refine)}$$

if $s \rightarrow_R t$ or $s \rightarrow_S t$, respectively, can be refined.

Refinements of rewrite rules do not decrease the proof measure, they can also destroy termination of the completion process. However they preserve correctness, since this property only depends on the elimination of critical pairs. In a completion process consisting only of critical pair computations there are more critical pairs without than with refinements.

On the other hand, an R rule (or S rule) could be deleted by a refinement that is eventually used as the first step in a simplification. Then the same simplification could be impossible after the refinement. In example 5.1, for instance, the rule $f(b) \to_R b$ could eventually be simplified in presence of additional rules and the procedure could terminate. If the initial R-rule of the simplifying proof is refined, this simplification could be blocked, the rule $f(b) \to_R b$ could persist and the procedure might diverge. Refine rules should be used carefully, one might also use rules that can be refined for simplification purposes, but not for critical pair computations.

6 Ground Completion for Quasiorderings

When $<$ is a quasiordering, the procedure of non-symmetric ground completion must be adapted, since in presence of reflexivity, the diagonal need not explicitly be considered. Also the theory of rewriting is slightly different, as section 3 shows. Here, we only point out some differences to completion with a non-symmetric transitive relation. See [16] for a more formal treatment.

The states of the completion process are now triples

$$(I, R, S),$$

since the diagonal is not needed. A Delete rule

$$\frac{(I \cup \{s < s\}, R, S)}{(I, R, S)}$$

that discards the reflexive part of the relation replaces the Diagonal rule. The remaining state transition rules and concepts are like for transitive relations, but operate on triples. We denote the set of transitions rules by C_q.

The differences between completion processes for transitive relations and quasiorderings are so small that theorem 4.1 holds with only minor changes. The additional Delete steps are no obstacle. The complexity measure is even slightly simpler than that for transitive relations. section. Given a reduction ordering $\prec$, the proof ordering $\prec$ is now defined as follows. We associate a measure

$$\mu : w \mapsto \begin{cases} (\{s_{i-1}, S_i\}, \_, \_), & \text{if } s_{i-1} \to_I s_i, \\ (\{s_{i-1}\}, l, s_i), & \text{if } s_{i-1} \to_R s_i, \text{ by rule } l \to r, \\ (\{s_i\}, l, s_{i-1}), & \text{if } s_{i-1} \to_S s_i, \text{ by rule } l \to r. \end{cases}$$

with a proof step (s_{i-1}, s_i) We compare tuples—except for Δ-steps—as for transitive relations. We obtain the following statement in analogy to theorem 4.1.

Theorem 6.1 *(i) The rules of C_q except* Deduce *are sound simplification rules with respect to the proof ordering $\prec$.*

(ii) The ordering $\prec$ induces termination of the proof transformation relation corresponding to the rules of C_q.

(iii) C_q is correct, if it is fair. The limit state of a fair non-failing non-symmetric ground completion process is a normal system.

Proofs largely follow those of theorem 4.1.

7 Non-Ground Non-Symmetric Completion

Section 3 shows that the lifting of the ground completion procedures to the non-ground case has a severe obstacle. Second-order context variables have been introduced to represent the variable critical pairs that arise for non-linear variables. There are three cases of interest. First the non-compatible case (no function is monotonic), second the linear case and third the general case.

The non-compatible case is interesting as the basis for ordered chaining calculi for transitive relations and quasiorderings [3,18].

The completion procedure of the linear case is obvious. Linearity is an invariant of the process. The transition rules are a combination of the equational non-ground and the the non-symmetric ground case. The Deduce rules, for instance, have the form

$$\frac{l_2 \rightarrow_S r_2 \qquad l_1[r_2']_p \rightarrow_R r_1}{l_1\sigma[l_2\sigma]_p < r_1\sigma}, \qquad \frac{l_2 \rightarrow_S r_2[l_1']_p \qquad l_1 \rightarrow_R r_1}{l_2\sigma < r_2\sigma[r_1\sigma]_p}.$$

Thereby σ is a most general unifier of r_2 and r_2' and l_1 and l_1', respectively. A particularly interesting instance is ground non-symmetric Knuth-Bendix completion modulo AC, since the AC rules, which are the only non-ground rules, are linear. Here one uses an AC-compatible reduction ordering [4,8], AC-unification and extended rules. For an AC-operation symbol f, a rule $f(r,s) \rightarrow_R t$ or $r \rightarrow_S f(t,s)$ is extended to $f(f(r,s),x) \rightarrow_R f(t,x)$ or $f(r,x) \rightarrow_S f(f(t,s),x)$, where x is a fresh extension variable. Hence also extensions preserve linearity. Now, two extension rules are added to the completion procedure. The rules

$$\frac{(I,R,S)}{(I,R\cup\{s \rightarrow_R t\},S)}, \qquad \frac{(I,R,S)}{(I,R,S\cup\{s \rightarrow_S t\})}, \qquad \text{(Extend)}$$

if $s \rightarrow_R t$ ($s \rightarrow_S t$) extends a non-extended rule in R (S), are the extension rules for ground completion for a quasiordering. See [17] for further details.

In the non-linear case, also Deduce rules for variable critical pairs and the extended signature for contexts have to be used. These Deduce rules are

$$\frac{l_2 \rightarrow_S r_2 \qquad l_1[x]_p \rightarrow_R r_1}{l_1\sigma[X(l_2)]_p < r_1\sigma}, \qquad \frac{l_2 \rightarrow_S r_2[x]_p \qquad l_1 \rightarrow_R r_1}{l_2\sigma < r_2\sigma[X(r_1)]_p},$$

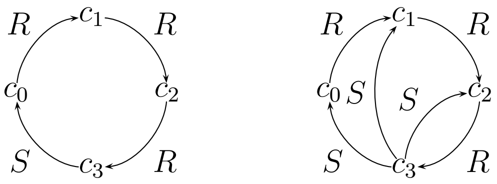
where x is a non-linear variable in l_1 and r_2, respectively and σ maps x to $X(r_2)$

or $X(l_1)$, respectively.. Thus terms with context variables are introduced into the process and also the Deduce rules for skeleton critical pairs depend on context unification. This is problematic in particular for using reduction orderings. For usual first-order non-ground terms one defines $s \prec t$ iff $s\sigma \prec t\sigma$ for all ground instances. One could use a similar definition for context terms. It is however questionable whether this criterion is effective in practice. It is also possible to code context terms as second-order terms, but not as higher-order patterns. Then it is questionable, whether reduction orderings still exist for this coding. A partial solution the problems associated with non-linear variables is to detect regularities in the overlap structure, that is to guess contexts manually. A simple example—also using Refine rules—in terms of semilattices is discussed in [16]. An entirely satisfactory solution to non-ground non-symmetric completion is currently not available.

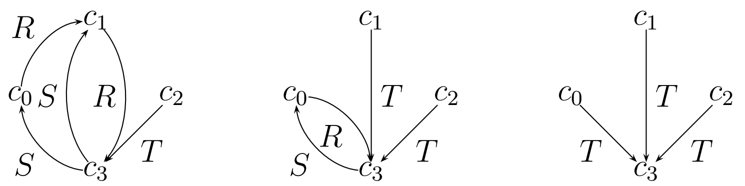
8 A Simple Application

We now use non-symmetric completion to develop a rule-based declarative algorithm for cycle elimination in digraphs or for constructing a partial ordering from a given quasiordering. The only difference is the treatment of the reflexive part. We disregard it for the sake of simplicity. This is interesting for the reachability analysis of state transition systems, since factoring out cycles compresses the state space. To collapse a cycle into a single node of a digraph, we must be able to find unique representants of the (partial) equivalence classes. We therefore extend the states of our completion procedures to include a set E of unordered equalities and a set T of equational rewrite rules. Here we only exemplify this algorithm. A separate paper will be devoted to the discussion of Knuth-Bendix completion with equalities and inequalities, including further applications.

Example 8.1 Given the precedence $c_i \succ c_j$ iff $i < j$ we consider the simple cycle at the left-hand side of the following diagram.



Note that in every presentation of a cycle, there must be at least one R- and one S-step (if the cycle can be ordered). The critical pairs $c_3 < c_1$ (from $c_3 \rightarrow_S c_0$ and $c_0 \rightarrow_R c_1$) and $c_3 < c_2$ (from $c_3 \rightarrow_S c_1$ and $c_1 \rightarrow_R c_2$) that arise from the left-hand diagram are oriented and depicted in the right-hand one. We now detect $c_2 \rightarrow_R c_3 \rightarrow_S c_2$, thus $c_2 \sim c_3$, which yields the equational rule $c_2 \rightarrow_T c_3$. This rule can now be used for simplification. Every occurrence of c_2 in I, R and S can be replaced by c_3. This situation is shown at the left-hand side of the following diagram.



We now detect $c_1 \rightarrow_R c_3 \rightarrow_S c_1$. Thus the equational rule $c_1 \rightarrow_T c_3$ can be used for simplification. This situation is shown at the center of the above diagram. We now detect $c_0 \rightarrow_R c_3 \rightarrow_S c_0$. Thus the equational rule $c_0 \rightarrow_T c_3$ can be used for simplification. This final situation is shown at the right-hand side of the above diagram. Obviously, the T-rules determine canonical representants of equivalence classes.

In more general cases, also nested loops must be treated and simplifications between T-rules must be computed. But this is beyond the scope of this paper.

9 Conclusion

We have seen how rewriting and completion can be extended beyond equations to non-symmetric transitive relations and quasiorderings. A satisfactory solution to the general non-ground case is still an open question. It depends on appropriate ordering constraints and the problem of context unification. Other interesting questions are the following. How to combine non-symmetric and equational completion for reasoning about graphs modulo an equivalence (like in section 8)? Are there mechanisms to make ground non-symmetric completion terminating, like its equational counterpart? Are normal systems unique, like equational canonical term rewrite systems? The last question is related to the uniqueness of clause sets that are saturated up to redundancy in ordered resolution.

Besides ordered resolution and ordered chaining calculi, a main application of non-symmetric rewriting is the reachability analysis in state transition systems, whose states are modeled by first-order terms. Also the analysis of execution sequences in rewriting logic [12] seems interesting. We have seen a simple example in section 8. Unlike the usual search techniques, part of the state graph can be precompiled. This could be beneficial in particular, when variables are involved and the presentation of the system is reused for different queries.

At a more general level, the results in this paper shed some light on the similarities and differences between formal reasoning with equalities and inequalities. Rewriting with inequalities seems to be more complex than equational rewriting. But, since inequational specifications are ubiquituous in mathematics and computer science, it also seems to be not less interesting for applications.

References

[1] F. Baader and T. Nipkow. *Term rewriting and all that.* Cambridge University Press, 1998.

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

[3] L. Bachmair and H. Ganzinger. Rewrite techniques for transitive relations. In *Ninth Annual IEEE Symposium on Logic in Computer Science*, pages 384–393. IEEE Computer Society Press, 1994.

[4] L. Bachmair and D. Plaisted. Termination orderings for associative-commutative rewriting systems. *J. Symbolic Computation*, 1(4):329–349, 1985.

[5] L. Bachmair, C.R. Ramakrishnan, I.V. Ramakrishnan, and A. Tiwari. Normalization via rewrite closures. In P. Narendran and M. Rusinowitch, editors, *Rewriting Techniques and Applications, 10th International Conference, RTA-99*, volume 1631 of *LNCS*, pages 190–204. Springer-Verlag, 1999.

[6] R. Bündgen. Buchberger's algorithm: The term rewriter's point of view. *Theoretical Computer Science*, 159(2):143–190, 1996.

[7] H. Comon. Completion of rewrite systems with membership constraints. In *Int. Coll. on Automata, Languages and Programming, ICALP'92*, volume 623 of *LNCS*. Springer-Verlag, 1992.

[8] C. Delor and L. Puel. Extension of the associative path ordering to a chain of associative-commutative symbols. In C. Kirchner, editor, *Rewriting Techniques and Applications*, volume 690 of *LNCS*, pages 389–404. Springer-Verlag, 1993.

[9] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 6, pages 244–320. Elsevier, 1990.

[10] D. Kapur. Shostak's congrucence closure as completion. In H. Comon, editor, *Rewrite Techniques and Applications, 8th International Conference, RTA-97*, volume 1232 of *LNCS*, pages 23–37. Springer-Verlag, 1997.

[11] J. Levy and J. Agustí. Bi-rewrite systems. *J. Symbolic Computation*, 22:279–314, 1996.

[12] J. Meseguer. Rewriting as a unified model of concurrency. Technical Report SRI-CSL-90-02R, SRI International, 1990.

[13] M. Schmidt-Schauss. A decision algorithm for stratified context unification. Technical Report Frank-report-12, FB Informatik, J.W. Goethe-Universität Frankfurt am Main, 1999.

[14] K. Stokkermans. *A Categorial Framework and Calculus for Critical-Pair Completion.* PhD thesis, Research Institute for Symbolic Computation, Johannes Kepler University, Linz, 1995.

[15] G. Struth. Non-symmetric rewriting. Technical Report MPI-I-96-2-004, Max-Planck-Institut für Informatik, Saarbrücken, 1996.

[16] G. Struth. *Canonical Transformations in Algebra, Universal Algebra and Logic.* PhD thesis, Institut für Informatik, Universität des Saarlandes, 1998.

[17] G. Struth. An algebra of resolution. In L. Bachmair, editor, *Rewriting Techniques and Applications, 11th International Conference*, volume 1833 of *LNCS*, pages 214–228. Springer-Verlag, 2000.

[18] G. Struth. Deriving focused calculi for transitive relations. In A. Middeldorp, editor, *Rewriting Techniques and Applications, 12th International Conference*, volume 2051 of *LNCS*, pages 291–305. Springer-Verlag, 2001.