On Modularity in Term Rewriting and Narrowing

Christian Prehofer*

Technische Universität München**

Abstract. We introduce a modular property of equational proofs, called modularity of normalization, for the union of term rewrite systems with shared symbols. The idea is, that every normalization with $R = R_1 + R_2$ may be obtained by first normalizing with R_1 followed by an R_2 normalization.
We develop criteria for this that cover non-convergent TRS R, where, as the main restriction, R_1 is required to be left-linear and convergent. As interesting applications we consider solving equations modulo a theory given by a TRS. Here we present a modular narrowing strategy that can be combined with nearly all common narrowing strategies. Furthermore, we also prove some modularity results for decidability of unification and matching (via termination of narrowing).

1 Introduction

We study a modular property of equational normalization proofs, called modularity of normalization, for term rewrite systems with shared symbols. The idea is, that every normalization with $R = R_1 + R_2$ may be obtained by first normalizing with R_1 followed by an R_2 normalization. We examine this idea in the context of rewrite systems, with possible extension to more complex and possibly hybrid systems.

There has been considerable work to show modularity for properties of term rewrite systems (TRS) such as confluence and termination, e.g. [16, 24, 10]. Most of the results on modularity concern the union of term rewrite systems with disjoint signatures, only a few also cover shared symbols [17, 20]. For the union of TRS with shared symbols, many approaches are based on commutation criteria, see Figure 1 for the common definitions, where $R = R_1 + R_2$ is assumed. For an overview see [16].

For convergent R, some criteria for modularity of normalization follow from known results. Our main contribution is to extend these to the case where R_2 is not convergent. This extension allows for much wider applications, e.g. to functional programming or functional-logic programming languages based on narrowing [11], where ground-convergence and non-termination are prevalent.

* Research supported by the DFG under grant Br 887/4-2, *Deduktive Programmentwicklung* and by ESPRIT WG 6028, *CCL*.

** Full Address: Fakultät für Informatik, 80290 München, Germany. Tel: +49 89 2105 2693, Fax: +49 89 2105 8183 E-mail: prehofer@informatik.tu-muenchen.de

Modularity of normalization has some interesting applications for equational reasoning: assume $R\downarrow \subseteq R_1\downarrow R_2\downarrow$ holds, then

- R has unique normal forms (wrt. reduction) iff R_1 and R_2 have. With this argument we present results that overlap with an open problem in Middeldorp [16] (also in [15]).
- We can combine arbitrary complete narrowing strategies for R_1 and R_2 to yield a complete R-narrowing strategy, which only considers solutions of the above form (see Section 5.2) and thus reduces the search space. For instance, an optimized strategy for R_1 can be used, which may not be applicable to R.
- We can combine unification procedures of R_1 and R_2 to obtain a unification procedure for R (see Section 5.1). This may yield new results about decidability of matching and unification.
- We have a notion of incremental or partial evaluation, e.g. if R_2 is yet unknown. Furthermore, if R_1 is convergent but R_2 is not, we have a way to prefer deterministic operations of R_1. This can be applied to incremental constraint solving based on rewriting.
- If R_1 and R_2 terminate, we have an effective way to compute all R-normal forms of a term without divergence, even if R does not terminate.

Before we examine these applications in Section 5, we study in Section 4 criteria for modularity of normalization. This property is easy to show for convergent R if R_1 and R_2 share no symbols: only R_1 must be left-linear and R_2 may not have collapsing rules. Otherwise, if they share symbols, critical pairs have to be considered as well. As we will see, the property that comes closest to modularity of normalization is commutation-over. Commutation-over usually requires left-linearity of R_1 and — in almost any reasonable setting — also right-linearity of R_2. This last requirement is prohibitive for many applications.

For modularity of normalization, convergence and left-linearity of R_1 are required in most criteria, but right-linearity of R_2 is not required.

In addition we observe that the termination of narrowing with a TRS R is a stronger property than decidability of unification modulo R. That is, there exist convergent TRSs, such that unification modulo R is decidable, but any complete narrowing strategy cannot terminate.

2 Preliminaries

An **abstract reduction system** (ARS) R is a binary relation on some set A. For $(a, b) \in R$ we write $a\ R\ b$. An element a is in R**-normal form** if no b with $a\ R\ b$ exists. For some reduction R, we denote the transitive closure by R^+, the reflexive transitive closure by R^*, and its reverse by R^{-1}. The union of two abstract reduction systems is written as $R_1 + R_2$. The **normalizations** of an ARS R are defined as

$$R\downarrow = \{(r, s) | (r\ R^*\ s) \text{ and s is in R-normal form}\}\ .$$

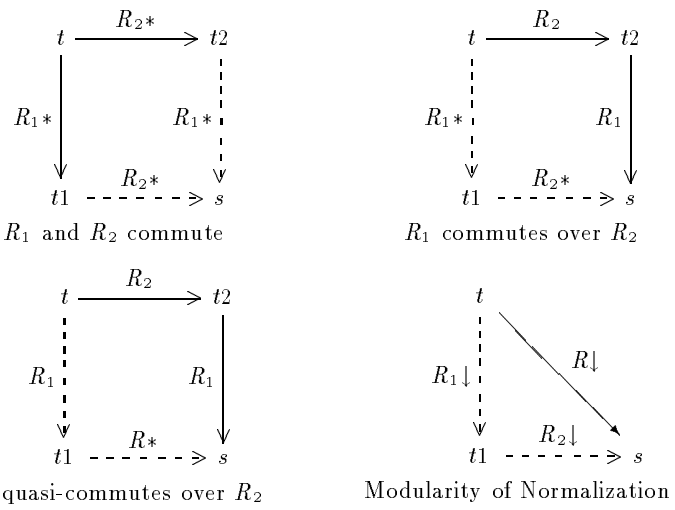


Fig. 1. Definitions of Commutation Properties

For an ARS $R = R_1 + R_2$ we call $R\downarrow \subseteq R_1\downarrow R_2\downarrow$ **modularity of normalization** (see also Figure 1).

We assume the standard notation of term rewriting, see e.g. [7]. **Positions in terms** are described by sequences over natural numbers. The subterm of s at position p is written as $s|_p$. A term t with the subterm at position p replaced by s is written as $t[s]_p$. **Substitutions** are finite mappings from variables to terms.

Define $Var(s)$ as the set of all variables occuring in a term s. A **rewrite rule** is a pair of terms written as $l \to r$. As in [9], we do not generally assume that l is a not a variable and that $Var(r) \subseteq Var(l)$. A **rewrite step** from a term s to t is defined as

$$s \xrightarrow[p,\theta]{l \to r} t$$

where $s|_p = \theta l$ and $t = s[\theta r]_p$ for some substitution θ. For such a rewrite step we often leave some of the parameters implicit. We write $t \xrightarrow{R} s$ or $t\ R\ s$ to denote a rewrite step with some rule $r \in R$.

A rewrite rule $l \to r$ is **variable-preserving** if $Var(l) = Var(r)$, and **collapsing** if $r \in Var(l)$. We write $t \leadsto^{l \to r}_{p,\sigma} s$ for a **narrowing** step from t to s if σ is a most general unifier of $t|_p$ and l and $s = \sigma t[r]_p$. We also write $t \leadsto^R s$ to describe a narrowing step with some rule from R.

A term is **linear** if each variable occurs at most once in it. A rewrite rule $l \longrightarrow r$ is **left-linear** if l is linear and **right-linear** if r is linear. Two positions in a term are **independent** if none is below the other. Let **parallel reduction** from s at independent positions with rules from R be written as $s \twoheadrightarrow^R t$ or as $sR^{\|}t$.

A TRS R is **confluent**, if for all reductions $s \xrightarrow{*}{}^{R} s_1$ and $s \xrightarrow{*}{}^{R} s_2$ the two terms are joinable, i.e. $s_1 \xrightarrow{*}{}^{R} s'$ and $s_2 \xrightarrow{*}{}^{R} s'$. A TRS is **convergent** if it is terminating and confluent.

Notice that in our setting, an (R_1, R_2) **critical pair** (s, t) is defined such that $\exists u.s \xleftarrow{R_1} u \xrightarrow{R_2} t$ where $u = \theta l$ for some rule $l \to r$ and one of the two reductions is at the root with $l \to r$ and the other is with a rule $l' \to r'$ at position p such that θ is a most general unifier of $l|_p$ and l'.

3 Commutation-over vs. Modular Normalization

In this section, we review known criteria for commutation-over. These will prepare the results on modularity of normalization. In the following, we assume a modular term rewriting system $R = R_1 + R_2$, where R_1 and R_2 may share function symbols.

In general, modularity of normalization and commutation are orthogonal (and similarly for commutation-over, if R_1 terminates): commutation only shows that an equivalent $R_1^* R_2^*$ derivation exists for any R reduction, but not how to find it. Conversely, $R\!\downarrow \subseteq R_1\!\downarrow R_2\!\downarrow$ implies commutation only for normalizing reductions. For convergent ARS, commutation-over is strictly stronger.

The following criteria for quasi-commutation has been shown in [21].

Theorem 1 (Raoult and Vuillemin). *Assume R_1 is a left-linear and R_2 is a right-linear rewrite system. If R_2^{-1} has no overlaps with R_1 then R_1 quasi-commutes over R_2.*

The original work in [21] shows this result for parallel reductions (i.e. with $R_i^{\|}$ for each R_i-reduction), an extension that will be used later. Note that the overlapping restriction implies that R_2 is non-collapsing. The above result has been extended by Geser [9] to allow for overlaps:

Theorem 2. *Assume R_1 is a left-linear and R_2 is a right-linear rewrite system. If all (R_2^{-1}, R_1)-critical pairs are in $R_1 R^*$, then R_1 quasi-commutes over R_2.*

It is easy to see that this theorem applies to disjoint TRSs modulo the linearity and collapse restrictions. The following result was shown independently in [18] and [2].

Proposition 3. *If R_1 terminates and R_1 quasi-commutes over R_2, then R_1 commutes over R_2.*

Proposition 4. *If R_1 quasi-commutes over R_2, then termination of $R = R_1 + R_2$ is a modular property.*

Let us see by examples why the linearity restrictions are required for Theorem 1:

- Right-linearity of R_2. Assume
  $R_1 : a \to b$
  $R_2 : 2x \to x + x$

Then

$$2a \xrightarrow{R_2} a + a \xrightarrow{R_1} a + b$$

but commuting the two reductions gives:

$$2a \xrightarrow{R_1} 2b \xrightarrow{R_2} b + b$$

Observe that this counter-example is strong; in most practical TRS, right-linearity is necessary.[3]

- Left-linearity of R_1. Assume
  $R_1 : x + x \to 2x$
  $R_2 : a \to b$
  Then

$$a + b \xrightarrow{R_2} b + b \xrightarrow{R_1} 2b$$

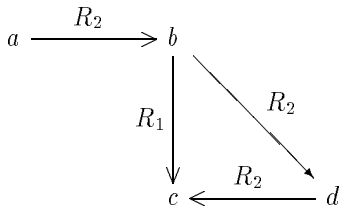
  but $a + b$ is not R_1-reducible.

In many applications, e.g. functional programming, left-linearity is a common restriction, in contrast to right-linearity. We will see in the next section that modularity of normalization does not require right-linearity.

To see that commutation-over is a very strong criterion, observe that it implies the preservation of normal forms:

Theorem 5 (Stroetmann [23]). *Assume R_1 and R_2 are abstract reduction systems. If R_1 quasi-commutes over R_2, then R_2 preserves R_1 normal forms.*

Also observe that neither preservation of normal forms nor modularity of normalization are necessary for commutation-over, even for convergent TRS. In the following example, R_1 commutes over R_2 and also $R\downarrow \subseteq R_1\downarrow R_2\downarrow$, but R_2 does not preserve R_1 normal forms.



Furthermore, it is easy to show that termination of R_1 is not necessary for commutation-over.

[3] First notice that the R_1-reduction $a \longrightarrow b$ is just an example for an arbitrary R_1-reduction. The counter-example works for any such R_1-reduction $a \longrightarrow b$, as long as R does not contain a reduction $b + b \xrightarrow{*} a + b$.

4 Criteria for Modularity of Normalization

We first review criteria for modular normalization which are based on a similar proof method as Theorem 4. Recall that commutation-over implies modularity of normalization for convergent theories:

Theorem 6. *Assume $R = R_1 + R_2$ is a confluent TRS and R_1 terminates. If R_1 commutes over R_2, then $R\downarrow \subseteq R_1\downarrow R_2\downarrow$.*

Proof. Assume $s \xrightarrow{R*} s''$ is not in $R_1\downarrow R_2\downarrow$. Then $s \xrightarrow{R_1*} s' \xrightarrow{R_2*} s'' \in R\downarrow$ follows from commutation-over. Now $s \notin R_1\downarrow$ must hold, i.e. $s' \xrightarrow{R_1} s_1$. From confluence and since $s'' \in R\downarrow$, we get $s_1 \xrightarrow{R} s''$. Apply commutation and the same argument again to this reduction; repeating this gives an infinite R_1-reduction, which is a contradiction. □

For convergent TRS, the simplest approach to obtain $R\downarrow \subseteq R_1\downarrow R_2\downarrow$ is to show that R_2 preserves R_1 normal forms. A necessary criterion has been presented for this case by Stroetmann [23]:

Theorem 7 (Stroetmann). *Assume R_1 and R_2 are TRSs such that R_1 is left-linear. Then R_1 normal forms are preserved by R_2 iff for all (R_2^{-1}, R_1)-critical pairs[4] (s_1, t_2), i.e. $s_1 \xrightarrow{R_2} t_1 \xrightarrow{R_1} t_2$, the term s_1 is R_1-reducible.*

Stroetmann uses the following easy result in [23] without explicitly stating it:

Proposition 8. *Assume $R = R_1 + R_2$ is a convergent TRS. Assume further that R_1 normal forms are preserved by R_2. Then $R\downarrow \subseteq R_1\downarrow R_2\downarrow$*

Using the above theorems, we easily get:

Corollary 9. *Assume $R = R_1 + R_2$ is a confluent TRS and R_1 is terminating. If R_1 is left-linear and for all (R_2^{-1}, R_1)-critical pairs (s, t) the term s is R_1-reducible, then $R\downarrow \subseteq R_1\downarrow R_2\downarrow$.*

4.1 Modular Normalization for Non-Convergent TRS

The above criteria for modularity of normalization have two main limitations. First, they assume a confluent TRS R. In practice, TRSs are often only ground confluent or divergent. In this section, we show that in many cases normalization with a convergent part of the TRS can be safely performed first, followed by rewriting with the non-convergent part.

Secondly, in the last section R_2 had to preserve R_1 normal forms. This is clearly not a necessary criterion for modularity of normalization, although it is difficult to find generalized proofs for this case.

[4] Called "non-standard reductions" in [23].

The idea of the following criterion is to transform an R-normalization $s \xrightarrow{R\downarrow} t$ into an $\xrightarrow{R_1\downarrow}\xrightarrow{R_2\downarrow}$-reduction. For this, we first seek an $s \xrightarrow{R_1*} u \xrightarrow{R_2*} t$-reduction, and then have to show that u is in R_1 normal form.

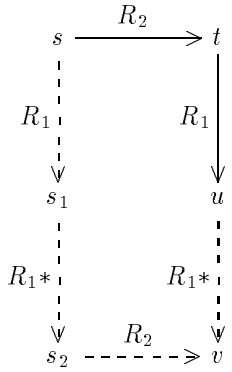
The following criterion is necessary for modularity of normalization:

Definition 10. Modularity of R_2-normalizations holds if for all s and t with t in R-normal form

$$s \xrightarrow{R_2\downarrow} t \quad \Longrightarrow \quad s \xrightarrow{R_1\downarrow}\xrightarrow{R_2\downarrow} t,$$

We will show that modularity of R_2-normalizations can be obtained by critical pair criteria. This is prepared by the following lemmata.

Lemma 11. *Assume $R = R_1 + R_2$ and R_1 is left-linear. If all (R_2^{-1}, R_1)-critical pairs are in $R_1^+ R_2^{\|}$, then for a reduction $s \xrightarrow{R_2} t \xrightarrow{R_1} u$, either $s \xrightarrow{R_1+} t \mathrel{+\!\!\!\rightarrow}^{R_2} u$ or the following diagram holds:*



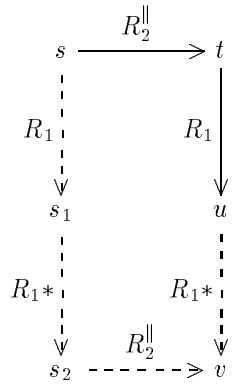
Proof. If the R_2-rule used in $s \longrightarrow t$ is right-linear, the result follows from an analysis similar to Theorem 2. The critical case is

$$s \underset{p_2}{\xrightarrow{R_2}} t \underset{p_1}{\xrightarrow{R_1}} u,$$

where position p_1 is below p_2, all other cases are as in Theorem 2. If there is a proper overlap, then the critical pair assumption immediately yields the result. Otherwise, p_1 is in a subterm of p_2 that is copied by the first R_2-step, and, since the reductions are independent, $s_1 \xrightarrow{R_1+} s_2 \xrightarrow{R_2} v$ and $u \xrightarrow{R_1*} v$. □

Next we extend this lemma to parallel reduction:

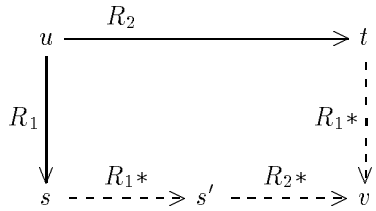
Lemma 12. *Assume $R = R_1 + R_2$ and R_1 is left-linear. If all (R_2^{-1}, R_1)-critical pairs are in $R_1 R_2^{\|}$, then for a reduction $s \xrightarrow{R_2\|} t \xrightarrow{R_1} u$, either $s \xrightarrow{R_1+} t \mathrel{+\!\!\!\rightarrow}^{R_2} u$ or the following diagram holds:*



Proof. Recall that the original work in [21] already considers parallel reduction. Hence if the R_1-reduction is inside of some reduction of $s \twoheadrightarrow^{R_2} t$ (or parallel to all of these), we can proceed as in Lemma 11, as the other R_2-reductions are independent.

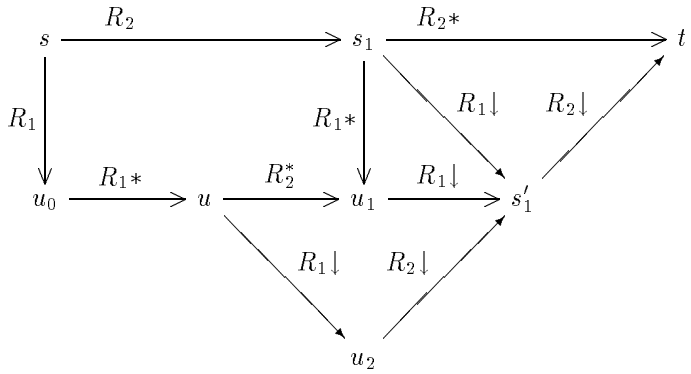
Otherwise, some R_2-reductions are below the R_1-reduction. In that case, we can apply Lemma 11 for each of these R_2-reductions. That is, we have $s \twoheadrightarrow^{R_2} s' \xrightarrow{R_2} t \xrightarrow{R_1} u$. As the R_2-reductions are below the stronger property $s' \xrightarrow{R_1} \twoheadrightarrow^{R_2} u$ can be shown similar to Lemma 11. Repeating this yields $s \xrightarrow{R_1} \twoheadrightarrow^{R_2} u$. □

Lemma 13. *Assume two TRS R_1 and R_2 such that R_1 is left-linear. Then there is a critical pair proof of $s = t$ or the following diagram holds:*



Lemma 14. *Assume $R = R_1 + R_2$ is a TRS such that R_1 is left-linear and convergent. If all (R_1, R_2)-critical pairs (s, t) are in $R_1^* R_2^* R_1^{-1*}$ and all (R_2^{-1}, R_1)-critical pairs are in $R_1 R_2^{\parallel}$, then modularity of R_2-normalizations holds.*

Proof. Consider a reduction $s \xrightarrow{R_2\downarrow} t$. If s is in R_1-normal form, the case is trivial. Otherwise, we give a proof by induction on the length of the R_2-reduction. The base case, $s \longrightarrow s$, is trivial, since s is in R-normal form. For the induction step, consider the following diagram:



In the above picture, we get $s_1 \xrightarrow{R_1{\downarrow}} \xrightarrow{R_2{\downarrow}} t$ from the induction hypothesis, then u_0 and s_1 can be joined to u_1 by Lemma 13 or by assumption on critical pairs. Then $u_1 \xrightarrow{R_1} s_1'$ is obtained from the confluence of R_1, and finally $u \xrightarrow{R_1{\downarrow}} u_2 \xrightarrow{R_2{\downarrow}} s_1'$ is obtained as in case 3 of the following theorem (this case does not assume this lemma). □

Finally, we are ready for the counterpart of Theorem 9 wrt. a non-convergent R_2.

Theorem 15. *Let $R = R_1 + R_2$ be a TRS such that R_1 is left-linear and convergent. Assume all (R_2^{-1}, R_1)-critical pairs are in $R_1 R_2^{\|}$. If modularity of R_2-normalizations holds, then $R{\downarrow} \subseteq R_1{\downarrow}R_2{\downarrow}$.*

Proof. Assume $s \xrightarrow{R{\downarrow}} t$. To show $s \xrightarrow{R_1{\downarrow}} \xrightarrow{R_2{\downarrow}} t$, we embed this into a more general case

$$s \xrightarrow{R*} t \xrightarrow{R_2{\downarrow}} t',$$

where t is in R_1-normal form. We show $s \xrightarrow{R_1{\downarrow}} \xrightarrow{R_2{\downarrow}} t'$ by induction on the number of R_2 steps in $s \xrightarrow{R*} t$. The base case follows trivially from modularity of R_2-normalizations.

In the remaining case, we reduce this problem to a reduction with fewer R_2-reductions. For this, we construct in the following a reduction $s \xrightarrow{R*} s' \xrightarrow{R_2*} t$. Then by modularity of R_2-normalizations we know that there exists s'' with $s' \xrightarrow{R_1{\downarrow}} s'' \xrightarrow{R_2{\downarrow}} t'$ and can reduce the case to $s \xrightarrow{R*} s'' \xrightarrow{R_2{\downarrow}} t$. There are three cases:

1.
$$s \xrightarrow{R*} s' \xrightarrow{R_2} t$$
Reduce to the case $s \longrightarrow s'$.

2.

$$s \xrightarrow{R*} s_1 \xrightarrow{R_2} s_2 \xrightarrow{R_1} t$$

This case is a special case of the following. If the reduction from s_1 to t is covered by a critical pair, see case 3, otherwise, from Lemma 11 we obtain $s_1 \xrightarrow{R_1*} s' \xrightarrow{R_2*} t$, since t is in R_1-normal form.

3.

$$s \xrightarrow{R*} s_1 \xrightarrow{R_2} s_2 \xrightarrow{R_1} s_3 \xrightarrow{R_1*} t$$

Here Lemma 12 applies and we obtain:

$$\begin{array}{ccccc} s & \xrightarrow{R*} & s_1 & \xrightarrow{R_2} & s_2 \xrightarrow{R_1*} t \\ & & \downarrow{\scriptstyle R_1+} & & \downarrow{\scriptstyle R_1*} \\ & & s_1' & \xrightarrow{R_2\,\|} & s_2' \end{array}$$

Since R_1 is confluent, $s_2' \xrightarrow{R_1*} t$ follows. Now repeat the same argument to the reduction $s_1' \xrightarrow{R_2\|} s_2' \xrightarrow{R_1*} t$, each time lifting an R_1-reduction leftwise over an $R_2^{\|}$-reduction. This procedure increases the length of the R_1-reduction starting from s_1. Thus it must terminate, otherwise we would get an infinite R-reduction starting from s_1. Hence we get $s_1 \xrightarrow{R_1*} s_2' \xrightarrow{R_2*} t$ in this case as well.

□

We conclude this section with some examples:

Example 1. The following TRS is a standard example formalizing natural numbers.

R_1:
- $s(x) * s(y) \to s(y + (x * s(y)))$
- $s(x) + y \to s(x + y)$
- $x + s(y) \to s(x + y)$

R_2:
- $0 + x \to x$
- $0 * x \to 0$
- $x * 0 \to 0$

Here $R{\downarrow} \subseteq R_1{\downarrow}R_2{\downarrow}$ holds, although R_2 contains collapsing rules.

Example 2. This example models integers with lists and assumes a many-sorted setting; extending our results to this seems straightforward.

R_1:
- $length(empty) \to 0$
- $length(cons(x, w)) \to s(length(w))$
- $first(cons(x, w)) \to x$
- $last(cons(x, w)) \to w$

R_2:
- $s(x) * s(y) \to s(y + (x * s(y)))$
- $s(x) + y \to s(x + y)$
- $x + s(y) \to s(x + y)$
- $0 + x \to x$
- $0 * x \to 0$
- $x * 0 \to 0$
- $p(s(x) \to x$
- $s(p(x)) \to x$
- $-s(x) \to p(-x)$
- $-p(x) \to s(-x)$
- $p(x) + y \to p(x + y)$
- $p(x) * y \to (x * y) + (-y)$
- $-0 \to 0$

Here $R\downarrow \subseteq R_1\downarrow R_2\downarrow$ holds, assuming that lists and integers have different sorts. Notice that R_2 is only ground confluent [19] and non right-linear.

5 Applications of Modular Normalization

We first show two simple results here; applications to equational reasoning by narrowing are elaborated in the following subsections. A simple consequence of $R\downarrow \subseteq R_1\downarrow R_2\downarrow$ is the modularity of unique normal forms:

Proposition 16. *Assume a TRS $R = R_1 + R_2$ and $R\downarrow \subseteq R_1\downarrow R_2\downarrow$. Then R has unique normal forms if R_1 and R_2 have unique normal forms.*

Proof. Assume a term t has two R-normal distinct forms. As for both normal forms $R_1\downarrow R_2\downarrow$-reductions must exist, this easily leads to a contradiction. The other direction is trivial. □

This result together with Theorem 15 does not fully answer the open question in Middeldorp [15]: modularity of unique normal forms (wrt. reduction) for left linear rules (of disjoint TRSs). Recall that for a disjoint union of two TRSs with the restriction in the last theorem, there can be no critical pairs. Then we obtain modularity of unique normal forms directly from the last theorem and Proposition 16, which is more general than the result by Middeldorp, since it does not assume disjoint signatures. That is, only one TRS must be non-collapsing and and the other convergent and left-linear. If both are collapsing, no counterexample with disjoint signatures is known. Note that it has been proved for disjoint union of left-linear and non-collapsing TRSs [15].

Modular normalization can also be used as a modular way to guarantee confluence of terminating TRSs.

Corollary 17. *Assume $R = R_1 + R_2$ terminates and $R\downarrow \subseteq R_1\downarrow R_2\downarrow$. Then R is confluent if R_1 and R_2 are.*

5.1 Modular Normalization and Decidability of Unification

The goal of this section is to show modular properties for termination of narrowing and decidability of unification. We assume here a naive notion of narrowing. That is, we start with narrowing derivations from some term and assume no further pruning of the search space.

We first observe the following important fact: termination of naive narrowing for a certain convergent TRS R not only means that matching is decidable for R. It entails the stronger property that all terms matching an instance of a certain term t can be finitely enumerated (more precisely, the equivalence classes can be described, since only maximally general terms are enumerated). Thus, termination of narrowing is a very strong property.

In particular, unification is decidable for a convergent TRS R, if narrowing terminates for R. This is easy to verify: enumerate by narrowing all possible matchers for the instances of two terms to unify and compare these pair-wise. This is in essence the same as adding a new rule $x = x \to true$ to a TRS to perform unification by narrowing.

It is well known that there exist theories with a (ground-)convergent TRS, for which matching is decidable, but unification is not. For instance, take natural numbers defined in [8]. Hence we get the following result:

Proposition18. *There exist (ground-)convergent TRS such that matching is decidable, but any complete narrowing strategy cannot terminate.*

Proof. If narrowing terminates, then unification would be decidable, as described above, which is a contradiction. □

A similar result for unification is easy to show. The problem that narrowing enumerates the full (constructor) term algebra has already been noticed by Bockmayr in [3].[5] The point here is that any complete strategy must have an infinite search space.

Proposition19. *There exist convergent TRS R such that R-unification is decidable, but any complete narrowing strategy cannot terminate.*

Proof. Consider for instance the TRS R:

$$p(s(x)) \to x, s(p(x)) \to x$$

Clearly, unification is decidable, but a term x has infinitely many instances that are not R-equivalent. Hence narrowing with the additional rule $x = x \to true$ must have an infinite search space. □

In the remainder of this section, we apply the results in previous section to decide unification and matching problems for TRSs with $R{\downarrow} \subseteq R_1{\downarrow}R_2{\downarrow}$. If a term t is to be matched with s, then it is sufficient to consider only $R_1{\downarrow}R_2{\downarrow}$-reductions. Assuming that narrowing (using any complete strategy) with R_1 terminates, we can finitely enumerate all possible matchers for a term t and apply matching (or unification) to each of these. Hence we get:

[5] Alexander Bockmayr also provided the example in the following proof.

Proposition 20. *Assume a convergent TRS R such that $R\downarrow \subseteq R_1\downarrow R_2\downarrow$ and assume further narrowing terminates for R_1. Then matching for R is decidable if matching with R_2 is decidable.*

In fact, we can show even more with termination of narrowing:

Proposition 21. *Assume a convergent TRS R such that $R\downarrow \subseteq R_1\downarrow R_2\downarrow$ and narrowing terminates for R_1 and for R_2. Then there is a complete and terminating narrowing strategy for R.*

To show possible applications of the above results, we mention some termination criteria for narrowing. Their combination using the above results is straightforward. If all right hand sides are (either constructor terms or) ground terms, then semantic unification is decidable, i.e. basic narrowing terminates [13]. Let R be a convergent rewrite system in which every left hand side is of the form $f(t_1, \ldots, t_n)$, such that each t_i is either a variable or a ground term. Then narrowing terminates [6].

5.2 Modular Narrowing Strategies

We now apply the results of the preceding sections to optimize narrowing strategies. For optimizing narrowing, we cannot naively use similar methods as for rewriting. It is easy to see that it is rarely sufficient to apply narrowing with R_1 as long as possible and then to use R_2. A naive non-deterministic narrowing procedure looks as follows:

```
FUNCTION solve(t,s)
  IF t and s unify, THEN success
  ELSE
    select any narrowing step t ⇝^R t'
    solve(t', s)
```

Many optimizations have been developed to remove redundancies from the immense search space of this unrestricted notion of narrowing, for an overview see [11]. With modularity of normalization, we can prune narrowing derivations that do not yield an $R_1\downarrow R_2\downarrow$-reduction. Thus we gain the following optimization for arbitrary narrowing strategies:

Theorem 22. *Assume a TRS R with $R\downarrow \subseteq R_1\downarrow R_2\downarrow$. Then the following is a complete (non-deterministic) narrowing strategy:*

Input: *t and s, where s is in R-normal form.*
Output: *R-matcher of t and s based on an $R_1{\downarrow}R_2{\downarrow}$ derivation, if narrowing is successful.*

Do the following steps:

1. *Use any narrowing strategy that is complete for R_1 to compute $t{\leadsto}^{R_1}_{\theta_0} n_1$, where n_1 is in R_1-normal form.*
2. *Use any narrowing strategy that is complete for R_2 on n_1.*
   If any derivation is of the form

$$n_1{\leadsto}^{R_2}_{\theta_1}\ldots{\leadsto}^{R_2}_{\theta_k} n_k\,,$$

   where $\theta = \theta_k\ldots\theta_1$, and θn_1 is R_1-reducible THEN stop (prune derivation)

Since the used narrowing strategies are free to compute normal forms, any complete strategies can be integrated with the above optimizations. For instance, basic narrowing [13], needed narrowing [1], and LSE-narrowing [4] are possible candidates.

Example 3. Assume

$$R_1 = \{x + 0 \to x, x + s(y) \to s(x + y), x + (-y) \to -(-x + y)\}$$

$$R_2 = \{-0 \to 0, --x \to x, s(-s(x)) \to -x\},$$

Geser [9] showed that R_1 commutes over R_2. Furthermore, it can be shown that $R{\downarrow} \subseteq R_1{\downarrow}R_2{\downarrow}$. Now for instance the R_2-narrowing step with the rule $-0 \to 0$

$$-x + x{\leadsto}^{R_2}_{\{x\mapsto 0\}} 0 + 0$$

can be pruned since $\{x \mapsto 0\}(-x + x) = -0 + 0$ is R_1-reducible.

Lazy Narrowing It should be mentioned that the above techniques can be used similarly for lazy narrowing [22] or lazy unification [12], but to the latter only to some limited extent. That is, when performing decomposition with a defined symbol f, i.e. if $f(t_1,\ldots,t_n) =^? f(t'_1,\ldots,t'_n)$ is transformed to $t_1 =^? t'_1,\ldots,t_n =^? t'_n$, the full term structure is lost and reducibility at the root f cannot be discovered. If only constructors are decomposed as in lazy narrowing, nothing is lost, as reducibility at constructor positions is impossible by definition.

6 Conclusions

We have presented criteria for modular TRS that are easier to obtain than most other existing modular criteria, as we only consider normalizing reductions. Here, our main contribution was to show that neither confluence nor termination of R_2 are needed for modular normalization. This allows for applications to

functional or logic programming languages, where only ground confluence and non-termination are common.

The main applications of modular normalization are solving equations, i.e. combining and optimizing narrowing strategies. Compared to almost any other improvements of narrowing strategies, this approach considers the global aspects of narrowing. For instance, with the advancement of functional-logic languages [11], modular aspects may gain importance.

It seems interesting to extend the above results to conditional rules which may also subsume logic programming [5]. In particular for the latter it seems interesting to isolate a convergent subset of the rules and to apply these deterministic rules eagerly.

A further aspect of modularity of normalization is incremental computation or partial evaluation; that is, normalization with R_1 may be performed safely before using or before even knowing R_2.

Acknowledgements. The author wishes to thank Alexander Bockmayr for providing the example for Proposition 19, as well as Alfons Geser for his careful reading of an earlier version of this paper.

References

1. S. Antoy, R. Echahed, and M. Hanus. A needed narrowing strategy. In *Proc. 21st ACM Symposium on Principles of Programming Languages*, pages 268–279, Portland, 1994.
2. L. Bachmair and N. Dershowitz. Commutation, transformation, and termination. In Joerg H. Siekmann, editor, *Proc. 8th Int. Conf. Automated Deduction*. LNCS 607, 1986.
3. A. Bockmayr. Narrowing with inductively defined functions. Technical report, Univ. Kaiserslautern, 1986. SEKI Memo 25/86.
4. Alexander Bockmayr, Stefan Krischer, and Andreas Werner. An optimal narrowing strategy for general canonical systems. In Michaël Rusinowitch and Jean-Luc Rémy, editors, *Conditional Term Rewriting Systems, Third International Workshop*, LNCS 656, pages 483–497, Pont-à-Mousson, France, July 8–10, 1992. Springer-Verlag.
5. P. G. Bosco, E. Giovanetti, and C. Moiso. Narrowing vs. SLD-resolution. *Theoretical Computer Science*, 59:3–23, 1988.
6. Jim Christian. Some termination criteria for narrowing and E-narrowing. In Kapur [14], pages 582–588.
7. N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In Jan Van Leeuwen, editor, *Handbook of Theoretical Computer Science Volume B: Formal Models and Semantics*, pages 243–320. Elsevier, 1990.
8. Nachum Dershowitz, Subrata Mitra, and G. Sivakumar. Decidable matching for convergent systems (preliminary version). In Kapur [14], pages 589–602.
9. A. Geser. *Relative Termination*. PhD thesis, Univ. Passau, 1990.
10. B. Gramlich. Generalized sufficient conditions for modular termination of rewriting. In H. Kirchner and G. Levi, editors, *Algebraic and Logic Programming: Proc. of the Third International Conference*, pages 53–68. Springer, Berlin, Heidelberg, 1992.

11. M. Hanus. The integration of functions into logic programming: A survey. 1994. To appear in Journal of Logic Programming.
12. M. Hanus. Lazy unification with simplification. In *Proc. 5th European Symposium on Programming*, pages 272–286. Springer LNCS 788, 1994.
13. Jean-Marie Hullot. Canonical forms and unification. In W. Bibel and R. Kowalski, editors, *Proceedings of 5th Conference on Automated Deduction*, pages 318–334. Springer Verlag, LNCS, 1980.
14. Deepak Kapur, editor. *11th International Conference on Automated Deduction*, LNAI 607, Saratoga Springs, New York, USA, June 15–18, 1992. Springer-Verlag.
15. Aart Middeldorp. Modular aspects of properties of term rewriting systems related to normal forms. In *Proc. 3rd Int. Conf. Rewriting Techniques and Applications*, pages 263–277. LNCS 355, 1989.
16. Aart Middeldorp. *Modular Properties of Term Rewriting Systems.* PhD thesis, Free University Amsterdam, 1990.
17. Aart Middeldorp and Y. Toyama. Completeness of combinations of constructor systems. In *Proc. 4th Int. Conf. Rewriting Techniques and Applications.* LNCS 488, 1991.
18. T. Nipkow and G. Weikum. Operationelle Semantik axiomatisch spezifizierter Abstrakter Datentypen. Master's thesis, TH Darmstadt, 1982. In German.
19. W. Nutt and P. Réty and. Basic narrowing revisited. In C. Kirchner, editor, *Unification.* Academic Press, 1990.
20. M. R. K. Krisna Rao. Completeness of hierarchical combinations of term rewriting system. In R.K. Shyamasundar, editor, *Foundations of Software Technology and Theoretical Computer Science*, pages 125–138. LNCS 761, 1993.
21. J. Raoult and J. Vuillemin. Operational and semantic equivalences between recursive programms. *J. of th ACM*, 27:772–796, 1980.
22. U. S. Reddy. Narrowing as the operational semantics of functional languages. In *Symposium on Logic Programming*, pages 138–151. IEEE Computer Society, Technical Committee on Computer Languages, The Computer Society Press, July 1985.
23. K. Stroetmann. The union of rewrite systems. unpublished, 1992.
24. Y. Toyama. On the Church-Rosser property for the direct sum of term rewriting systems. *Journal of the ACM*, 34(1):128–143, 1987.