Theoretical Computer Science 67 (1989) 283-301
North-Holland

TERM-REWRITING SYSTEMS WITH RULE PRIORITIES *

J.C.M. BAETEN

Programming Research Group, University of Amsterdam, 1018 WV Amsterdam, The Netherlands

J.A. BERGSTRA

Programming Research Group, University of Amsterdam, 1018 WV Amsterdam and Department of Philosophy, State University of Utrecht, 3508 TB Utrecht, The Netherlands

J.W. KLOP

Centre for Mathematics and Computer Science, 1009 AB Amsterdam and Department of Computer Science, Free University, 1007 MC Amsterdam, The Netherlands

W.P. WEIJLAND

Centre for Mathematics and Computer Science, 1009 AB Amsterdam, The Netherlands

Abstract. In this paper we discuss term-rewriting systems with *rule priorities*, which simply is a partial ordering on the rules. The procedural meaning of such an ordering then is, that the application of a rule of lower priority is allowed only if no rule of higher priority is applicable. The semantics of such a system is discussed. It turns out that the class of all *bounded* systems indeed has such a semantics.

1. Introduction

Term-rewriting systems are an important tool to analyze the consistency of algebraic specifications, and are also becoming increasingly important for implementation. Some general references for algebraic specifications are [9, 11, 12, 15, 18]. Some general references for term-rewriting systems are [13, 19, 20, 16].

For implementation purposes it is sometimes convenient to write down term-rewriting systems (TRS's) where some ambiguities between the rules are present, while adopting some restrictions on the use of these rewrite rules to the effect that the ambiguities are not actually "used". The mechanism that we discuss in this paper consists of giving priority to some rules over others in cases of "conflict". Such a priority ordering on the rules has been used in a rather extended way, as is for instance the case in programming languages such as HOPE, ML or MIRANDA and in syntax editors like those used in MENTOR or TYPOL, where the pretty printer is directed by pattern matching rules with priorities, or in specification languages such as OBJ [10] where reductions of terms can be forbidden depending

* Partial support received from the European Communities under ESPRIT Contract No. 432, Meteor (an integrated formal approach to industrial software development).

0304-3975/89/$3.50 © 1989, Elsevier Science Publishers B.V. (North-Holland)

on their sorts. In fact, our interest in this subject began when we tried to give a formal semantics to Backus' system FP (Functional Programming) (see [1, 2]). This frequent use is due to the strong (although natural) expressive power of such a system and its intuitive appeal. Another extension of the purely equational formalism, which retains the initial algebra semantics and also increases expressive power, is the introduction of conditional equations, see [21, 14, 5].

Here we consider a TRS with rule priorities, called a *priority rewrite system* (PRS). We study the effect of such a priority assignment to rules, without imposing further restrictions such as choosing a certain reduction strategy in combination with rule priorities. That is, we wish to consider the priority mechanism on itself. As to the *executability* of the specification given by a PRS this is a drawback: in general a PRS without more will not be an executable specification. In fact, it turns out that it is rather problematic whether a "pure PRS" has a well-defined semantics at all. It may even be the case that a pure PRS does not possess a well-defined semantics (i.e. does not determine an actual rewrite relation). Apart from the fact that PRS's have some interesting mathematical properties, we find that it is worth-while to establish some facts about them in order to get a better understanding of both their expressive power and their complications. Moreover, a decent subclass of PRS's can be determined which does possess a well-defined semantics and we will also establish a general theorem ensuring confluence for several of such PRS's. A typical example we will consider is the class of all TRS's with a so-called *specificity ordering*.

The theory of PRS's is also useful in connection with *modularity*: we can break up a specification in a number of (parametrized) smaller specifications in ways that are not expressible by means of equational specifications.

This article is a major revision of [3], which itself is a revision of [2].

2. Priority rewrite systems

In this section we will present the basic definitions of term-rewriting systems with rule priorities (often called a *priority rewrite system* or PRS, for short) and define what it means for such a PRS to be *well-defined.* We start out with some examples, to give the reader an intuitive idea of a PRS.

Example 2.1. Consider the signature for the natural numbers with predecessor, successor, sum and zero, and the rewrite rules in Table 1. Without the arrow this

Table 1.

	$r1$:	$P(0) \to 0$
	$r2$:	$P(S(x)) \to x$
$\downarrow$	$r3$:	$x + 0 \to x$
	$r4$:	$x + y \to S(x + P(y))$

set of rewrite rules is ambiguous (i.e. more than one rule can be applied to a certain redex), and does not implement our intention (to specify predecessor and sum on the natural numbers). The arrow now means that the third rule ($r3$) has priority over the fourth ($r4$). However, there is a caveat: the term $x+P(S(0))$ does not match the left-hand side of $r3$; but this does not mean that $r3$ may be "by-passed" in favour of applying $r4$ on this term. We may only by-pass $r3$ if, in no subsequent reduction of $y=P(S(0))$, we will get a match with the left-hand side of $r3$. So, in this case, we are not allowed to by-pass $r3$ and the correct reduction is

$$x+P(S(0)) \overset{r2}{\longrightarrow} x+0 \overset{r3}{\longrightarrow} x.$$

Example 2.2. Finite sets of natural numbers with insertion and deletion. The signature consists of

sorts	NAT, SET
functions	S : NAT → NAT
	ins : NAT × SET → SET
	del : NAT × SET → SET
constants	$0 \in$ NAT
	$\emptyset \in$ SET
variables	$x, y, \ldots \in$ NAT
	$X, Y, \ldots \in$ SET.

The rewrite rules for insertion and deletion are shown in Table 2. Again, $r3$ has priority over $r4$. That $r4$ is "correct" is because if one is allowed to use it, then $\mathrm{del}(x, X)$ does not match the left-hand side of $r3$, so X is not of the form $\mathrm{ins}(x, Y)$; in other words, "$x \notin X$", hence $X-\{x\}=X$.

Table 2.

	$r1$:	$\mathrm{ins}(x, \mathrm{ins}(x, X)) \to \mathrm{ins}(x, X)$
	$r2$:	$\mathrm{ins}(x, \mathrm{ins}(y, X)) \to \mathrm{ins}(y, \mathrm{ins}(x, X))$
│	$r3$:	$\mathrm{del}(x, \mathrm{ins}(x, X)) \to \mathrm{del}(x, X)$
↓	$r4$:	$\mathrm{del}(x, X) \to X$

Example 2.3. The factorial function. Add rules for multiplication to the rules of Table 1. Then factorial can be specified as in Table 3.

Table 3.

│	$\mathrm{Fac}(0) \to S(0)$
↓	$\mathrm{Fac}(x) \to x \cdot \mathrm{Fac}(P(x))$

Example 2.4. In a signature containing booleans, one may encounter rules for equality as in Table 4. Thus, for any specification, containing booleans, adding these

Table 4.

$\downarrow$	$\mathrm{eq}(x, x) \to \mathrm{T}$
	$\mathrm{eq}(x, y) \to \mathrm{F}$

equations describes the equality function on a certain sort. We claim that, without using rewrite rules with priority, such a parametrized specification *cannot be found*! Even when using auxiliary sorts and functions, or even conditional equations, such a specification cannot be found. One can see this from the fact that otherwise each initial algebra would be decidable, the proof of which requires a very systematic analysis of initial algebra semantics in the light of computability theory. In essence, this work has been carried out in [6, 7], see also [8].

Our conclusion is, that equational specifications do not support proper modularization (in unexpected cases). We claim that priority rewrite systems support modularity much better.

Let us now turn to the formal definition of rule priorities together with its mechanism of blocking rule applications.

Definition 2.5. A *priority rewrite system*, or PRS for short, is a pair $(\mathbb{R}, <)$, where $\mathbb{R}$ is a term-rewriting system and $<$ is a partial order on the set of rules of $\mathbb{R}$.

As a notation in a listing of rewrite rules we write $\downarrow^{r1}_{r2}$ when $r1 > r2$.

Definition 2.6. Let r be a rewrite rule of the PRS $\mathbb{R}$.

(i) An instantiation (possibly containing variables) of the left-hand side of r is called an *r-redex*. Note that this is regardless of whether the r-redex, in view of the priority restrictions, is actually "enabled", i.e. is allowed to be rewritten according to rule r.

(ii) A closed instantiation (closed instance) $t \to s$ of the rewrite rule r is called a *rewrite*. We will write $t \to^r s$ or r: $t \to s$.

(iii) The closure of the relation $\to$ under contexts is *one-step reduction*, and denoted by $\to$.

(iv) The transitive and reflexive closure of the relation $\to$ is (*more-step*) *reduction*, denoted $\twoheadrightarrow$.

Definition 2.7. Let $F(t_1, \ldots, t_n)$ be some term in a TRS. A reduction of $F(t_1, \ldots, t_n)$ is called *internal* if it proceeds entirely in the arguments $t_1, \ldots, t_n$ (so the head-symbol F is "unaffected").

Now we can formulate in a first approximation what reduction relation a PRS is meant to describe: Let r be a rule of the PRS $\mathbb{R}$ and let t be an r-redex. Then t may be rewritten according to r if *for no rule $r' > r$ it is possible to rewrite t, by means of*

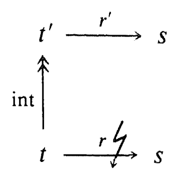

Fig. 1.

an internal reduction, to an r'-redex t' (see Fig. 1). To see why the reduction to a "higher" redex scheme, blocking the "lower" reduction of t, must be *internal*, one should consider that only internal reductions preserve the "identity" of the term-to-be-reduced, in casu t. The following example may clarify this: Consider the PRS in Table 2, and consider the $r4$-rewrite

$$\mathrm{del}(0, \mathbf{del(0, ins(0, \emptyset))}) \xrightarrow{r4} \mathbf{del(0, ins(0, \emptyset))}.$$

Intuitively, this application of $r4$ is correct since the bold part in the left-hand side denotes a set not containing 0. But if we had stipulated above that the internal reduction could be *any* reduction, the present application of $r4$ would be illegal since the right-hand side is also a $r3$-redex and $r3 > r4$. The point is that the priority provides us with some sort of a *matching* mechanism by rewriting the arguments of the term in order to prove them "equal" to the ones in the rule with higher priority. Indeed, application of $r4$ on a term $\mathrm{del}(t, T)$ is only allowed if it is not the case that both $t \twoheadrightarrow s$ and $T \twoheadrightarrow \mathrm{ins}(s, S)$ for some s, S, that is, if there is an internal reduction of the form $\mathrm{del}(t, T) \overset{\mathrm{int}}{\twoheadrightarrow} \mathrm{del}(s, \mathrm{ins}(s, S))$. In such an internal reduction, the right-hand side "matches" with the left-hand side with respect to the equality theory induced by the reduction relation.

In the following definition we will present a formal criterion for a rewrite to be "enabled". It is important to note that in fact we make a choice here. For instance, in [2, 4] different notions were used.

Definition 2.8. Let R be a set of rewrites for the PRS $\mathbb{R}$ (i.e. closed instantiations of rules of $\mathbb{R}$). The rewrite $t \rightarrow^{r} s$ is *correct* (w.r.t. R) if there is no internal R-reduction $t \overset{R}{\twoheadrightarrow} t'$ to an r'-rewrite $t' \rightarrow^{r'} s' \in R$ with $r' > r$. So in the situation of Fig. 2, the rewrite $t \rightarrow^{r} s$ is *not* correct w.r.t. R.

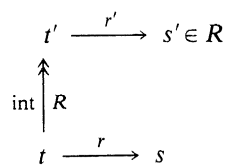

Fig. 2.

Definition 2.9. R is called *sound* if all its rewrites are correct with respect to R. R is *complete* if it contains all rewrites which are correct w.r.t. R.

In Fig. 2 ${}^{R}\twoheadrightarrow$ denotes a reduction using only rewrites from R. Note that if R is sound and $t \rightarrow^{r} s$ is correct w.r.t. R, then $R' = R \cup \{t \rightarrow^{r} s\}$ need not be sound, since $t \rightarrow^{r} s$ may be used in an internal R'-reduction making some other rewrite $t^* \rightarrow s^*$ illegal.

Finally, note that the concept of completeness of Definition 2.9 has nothing to do with the notion "complete" for TRS's, defined as meaning "confluent and terminating" (see, e.g., [17]).

Clearly, if a PRS $\mathbb{R}$ determines a reduction relation R as its semantics, we will require that R is sound (i.e. it may not contain forbidden rewrites). Now it might be thought that all we have to do is to look for a *maximal* sound rewrite set of $\mathbb{R}$. However, such a maximal sound rewrite set will not be unique in general, and therefore does not qualify as the semantics of $\mathbb{R}$; furthermore, we will require the semantics of $\mathbb{R}$ to contain all r-rewrites for rules r which have maximal priority, and a maximal sound rewrite set need not obey this requirement, as the following example shows.

Example 2.10. Let $\mathbb{R}$ be the PRS with rules and priorities in Table 5. Then $R_1 = \{0 \rightarrow 1, A(1) \rightarrow 2\} \cup \{A(t) \rightarrow 3\text{: all closed } t \text{ except } 0, 1\}$ is a maximal sound rewrite set (the intended semantics!), but also $R_2 = \{A(1) \rightarrow 2\} \cup \{A(t) \rightarrow 3\text{: all closed } t \text{ except } 1\}$ is a maximal sound rewrite set. As a candidate for the semantics of $\mathbb{R}$, R_2 is unsatisfactory as it does not contain the maximum priority rule instance $0 \rightarrow 1$. To fix this problem we require that the semantics R of a PRS $\mathbb{R}$ is also complete, since there is no reason to exclude from R a rewrite $t \rightarrow s$ which cannot be shown illegal by R. Note that the rewrite set R_2 is not complete (as $0 \rightarrow 1$ is correct w.r.t. R_2), but that R_1 is.

Table 5.

	$0 \rightarrow 1$
$\downarrow$	$A(1) \rightarrow 2$
	$A(x) \rightarrow 3$

Definition 2.11. Assume the PRS $\mathbb{R}$ has a unique sound and complete rewrite set R; then R is called the *semantics* of $\mathbb{R}$; furthermore, $\mathbb{R}$ will be called *well-defined.*

The idea behind Definition 2.11 is that a rewrite is part of the semantics of $\mathbb{R}$ if and only if there is no way to show that it is illegal using *legal* rewrites only. Obviously, such a definition has a circular nature and as a consequence there are PRS's that do not have a proper semantics, as is shown by the following example.

Table 6.

$r1$:	$1 \to A(1)$
↓ $r2$:	$A(0) \to 1$
$r3$:	$A(x) \to 0$

Example 2.12. Consider the PRS $\mathbb{R}$, with rules and priorities as in Table 6. We allow the reduction $A(1) \to 0$ if and only if *not* $1 \twoheadrightarrow 0$. However, one can easily verify that $1 \twoheadrightarrow 0$ if and only if $A(1) \twoheadrightarrow 0$, since its left-hand side (i.e. 1) only matches the first rule in $\mathbb{R}$. Therefore, $A(1) \to 0$ actually "blocks itself" and it is not quite clear whether or not this reduction should be part of the semantics of $\mathbb{R}$.

What actually *is* the problem in Example 2.12 is that every internal reduction sequence from $A(1)$ to $A(0)$ uses the rewrite $A(1) \to 0$. Thus, $A(1) \to 0$ is part of the semantics of such a PRS iff it is not. We will return to this problem later on (see Example 3.15).

In the following we will use some extra notations.

Definition 2.13. Let $\mathbb{R}$ be a PRS, then the set of all rewrites for $\mathbb{R}$ is denoted by $R_{\max}$. Next assume $R \subseteq R_{\max}$ is a set of rewrites for $\mathbb{R}$; then the *closure* $c(R)$, often denoted by R^c, of R is the set of all rewrites which are correct with respect to R.

Lemma 2.14. *Let R, S be sets of rewrites for the PRS* $\mathbb{R}$.

(i) *R is sound* $\Leftrightarrow$ $R \subseteq R^c$,

(ii) *R is complete* $\Leftrightarrow$ $R \supseteq R^c$.

(iii) *R is sound and complete* $\Leftrightarrow$ $R = R^c$.

(iv) $R \subseteq S \Rightarrow R^c \supseteq S^c$.

(v) $R \supseteq S$, *S is sound and complete* $\Rightarrow$ *R is complete.*

(vi) $R \subseteq S$, *S is sound and complete* $\Rightarrow$ *R is sound.*

Lemma 2.14 follows directly from Definitions 2.9 and 2.13. From (iii) it follows that any rewrite set is sound and complete for $\mathbb{R}$ if and only if it is a *fixed point* of the closure map c. Furthermore, from (iv) we find that c is an *antimonotonic* mapping on the powerset of $R_{\max}$.

Proposition 2.15. *The direct sum of two well-defined PRS's need not be well-defined.*

The proof of Proposition 2.15 is given by the following example.

Example 2.16 (G.J. Akkerman). Consider the following PRS's $\mathbb{P}$ and $\mathbb{R}$ in Tables 7 and 8 respectively. Considering $\mathbb{P}$ we note that all reducts of $D(x)$ are either of the

Table 7.

↓ $F(B(0,1)) \to 2$
$F(D(x)) \to B(x,x)$
$D(x) \to F(D(x))$

Table 8.

$\mathrm{or}(x,y) \to x$
$\mathrm{or}(x,y) \to y$

form $F^k(D(x))$, or of the form $F^k(B(x, x))$, so $D(x)$ cannot be reduced to $B(0, 1)$. Therefore, $\mathbb{P}$ is a well-defined PRS (in some sense its rules are nonoverlapping). Clearly, $\mathbb{R}$ is well-defined since it is a TRS, thus having $R_{\max}$ as its semantics. However, the direct sum $\mathbb{P}\oplus\mathbb{R}$ of $\mathbb{P}$ and $\mathbb{R}$ is not well-defined, for consider the following rewrite x: $F(D(\text{or}(0, 1))) \to B(\text{or}(0, 1), \text{or}(0, 1))$. Assume $\mathbb{P}\oplus\mathbb{R}$ has a sound and complete rewrite set R such that $x \in R$; then we have the following internal reduction in R:

$$\begin{aligned} F(D(\text{or}(0, 1))) &\to F(F(D(\text{or}(0, 1)))) \\ &\to F(B(\text{or}(0, 1), \text{or}(0, 1))) \to \cdots \to F(B(0, 1)) \end{aligned}$$

contradicting the soundness of R. On the other hand, if $x \notin R$ then x is incorrect with respect to R (since R is complete) and so there exists a reduction sequence $D(\text{or}(0, 1)) \overset{R,\text{int}}{\twoheadrightarrow} B(0, 1)$ in R. Investigating all such possible reductions one easily verifies that they all contain the rewrite x again therefore x has to be an element in R. This is a contradiction. Thus $\mathbb{P}\oplus\mathbb{R}$ is not a well-defined PRS.

Open question. Clearly, the PRS's introduced in this section are (in general) not executable since it is not decidable whether or not there exists an internal reduction from a "lower" LHS to a "higher" one. Until now, it is still an open question what classes of PRS's *are* executable, however. It would be very interesting to establish a result of this kind in order to be able to turn the priority mechanism into a executable programming language.

3. Fixed points

In this section we will present some more theory on sound and complete rewrite sets. In particular we will investigate the structure of the complete lattice $(R_{\max}, \subseteq)$ together with the closure map c. From now on we write $x, y, z, \ldots$ for rewrites from $R_{\max}$ and $r, r', \ldots$ will denote rules from the PRS $\mathbb{R}$. Furthermore, LHS(x) and RHS(x) will denote the left-hand and right-hand sides of the rewrite x, i.e. $x \equiv \text{LHS}(x) \to \text{RHS}(x)$.

Definition 3.1. Let O be a rewrite set. We write $x \lhd O$ (*O obstructs x*), if there is an internal reduction of LHS(x) (say this is an r-redex) to a "higher" redex (i.e. an r'-redex with $r' > r$), such that the internal reduction uses precisely all rewrites in O. Furthermore, we write $x \lhd\!\lhd y$ if there exists an obstruction $x \lhd O$ such that $y \in O$.

In Fig. 3 we have $x \lhd \{x_1, \ldots, x_n\}$ and $x \lhd\!\lhd x_k$ for all $1 \leqslant k \leqslant n$. An element (x, O) of $\lhd$ will be called an *obstruction* and O will be called an *obstruction of* x. We may have that an obstruction is empty, i.e. $x \lhd \emptyset$. For instance, in Example 2.12 we find that the rewrite x: $A(0) \to 0$ has an empty obstruction since its left-hand side is identical with the left-hand side of $r2$ which has higher priority.

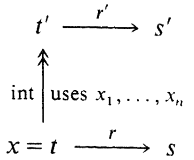

Fig. 3.

From the antimonotonic mapping c we easily construct a monotonic mapping, called $T_{\mathbb{R}}$.

Definition 3.2. Suppose R is a rewrite set for the PRS $\mathbb{R}$; then define $T_{\mathbb{R}}(R) = (R^c)^c$.

Since c is antimonotonic, it follows directly that $T_{\mathbb{R}}$ is monotonic. Note that if R is a fixed point of c then it is a fixed point of $T_{\mathbb{R}}$. In order to be able to find fixed points of $T_{\mathbb{R}}$, let us consider the following construction.

Definition 3.3. Let $\mathbb{R}$ be a PRS. Then for all ordinals α we define

$T_{\mathbb{R}}\uparrow 0 = \emptyset$,	$T_{\mathbb{R}}\downarrow 0 = \emptyset^c$,
$T_{\mathbb{R}}\uparrow \alpha + 1 = T_{\mathbb{R}}(T_{\mathbb{R}}\uparrow \alpha)$,	$T_{\mathbb{R}}\downarrow \alpha + 1 = T_{\mathbb{R}}(T_{\mathbb{R}}\downarrow \alpha)$,
$T_{\mathbb{R}}\uparrow \alpha = \bigcup_{\beta<\alpha} (T_{\mathbb{R}}\uparrow \beta)$, if α is a limit ordinal;	$T_{\mathbb{R}}\downarrow \alpha = \bigcap_{\beta<\alpha} T_{\mathbb{R}}\downarrow \beta$, if α is a limit ordinal.

Clearly, $T_{\mathbb{R}}\uparrow \alpha$ is the *α-repetition of* $T_{\mathbb{R}}$ starting from $\emptyset$, and so is $T_{\mathbb{R}}\downarrow \alpha$ but then starting from $\emptyset^c$. Recall, that $\emptyset^c$ does not need to be equal to $R_{\max}$. It is a well-known fact that any monotonic mapping such as $T_{\mathbb{R}}$ on a complete lattice $(\emptyset^c, \subseteq)$ has a *least fixed point* (lfp) and a *greatest fixed point* (gfp). Furthermore, there exists an ordinal α such that $\text{lfp}(T_{\mathbb{R}}) = T_{\mathbb{R}}\uparrow \alpha$ and $\text{gfp}(T_{\mathbb{R}}) = T_{\mathbb{R}}\downarrow \alpha$ which is a consequence of a well-known theorem from Knaster and Tarski [22]. The smallest ordinal α such that $T_{\mathbb{R}}\uparrow \alpha$ is a fixed point is called the *closure ordinal* for $T_{\mathbb{R}}$.

Lemma 3.4. *For all ordinals* α *we have* $(\bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow \beta)^c = \bigcap_{\beta<\alpha} (T_{\mathbb{R}}\uparrow \beta)^c$.

Proof. ($\subseteq$): Since $(\bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow \beta) \supseteq T_{\mathbb{R}}\uparrow \gamma$ for all ordinals $\gamma < \alpha$ we have $(\bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow \beta)^c \subseteq (T_{\mathbb{R}}\uparrow \gamma)^c$ (Lemma 2.14(iv)) for all $\gamma < \alpha$, and therefore $(\bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow \beta)^c \subseteq \bigcap_{\beta<\alpha} (T_{\mathbb{R}}\uparrow \beta)^c$.

($\supseteq$): If $x \notin (\bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow \beta)^c$ and $x \lhd \{y_1, \ldots, y_k\}$ for some obstruction $\{y_1, \ldots, y_k\} \subseteq \bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow \beta$. Since $\{y_1, \ldots, y_k\}$ is finite and $(T_{\mathbb{R}}\uparrow \beta)_{\beta<\alpha}$ is nondecreasing, there exists some $\gamma < \alpha$ such that $\{y_1, \ldots, y_k\} \subseteq T_{\mathbb{R}}\uparrow \gamma$. Then x has an obstruction in $T_{\mathbb{R}}\uparrow \gamma$, i.e. $x \notin (T_{\mathbb{R}}\uparrow \gamma)^c$, so $x \notin \bigcap_{\beta<\alpha} (T_{\mathbb{R}}\uparrow \beta)^c$. Hence $(\bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow \beta)^c \supseteq \bigcap_{\beta<\alpha} (T_{\mathbb{R}}\uparrow \beta)^c$. $\square$

Theorem 3.5. *For all ordinals α we have*
(i) $(T_{\mathbb{R}}\uparrow\alpha)^c = T_{\mathbb{R}}\downarrow\alpha$,
(ii) $(T_{\mathbb{R}}\downarrow\alpha)^c = T_{\mathbb{R}}\uparrow\alpha+1$.

Proof. By transfinite induction on α.
($\boldsymbol{\alpha=0}$) (i): $(T_{\mathbb{R}}\uparrow 0)^c = \emptyset^c = T_{\mathbb{R}}\downarrow 0$ and (ii): $(T_{\mathbb{R}}\downarrow 0)^c = (\emptyset^c)^c = T_{\mathbb{R}}(\emptyset) = T_{\mathbb{R}}\uparrow 1$.
($\boldsymbol{\alpha+1}$) (i):

$$(T_{\mathbb{R}}\uparrow\alpha+1)^c = ((T_{\mathbb{R}}\downarrow\alpha)^c)^c \qquad \text{(induction)}$$

$$= T_{\mathbb{R}}(T_{\mathbb{R}}\downarrow\alpha) = (T_{\mathbb{R}}\downarrow\alpha+1);$$

(ii):
$$(T_R\downarrow\alpha+1)^c = (((T_{\mathbb{R}}\downarrow\alpha)^c)^c)^c = T_{\mathbb{R}}((T_{\mathbb{R}}\downarrow\alpha)^c) = T_{\mathbb{R}}(T_{\mathbb{R}}\uparrow\alpha+1) \qquad \text{(induction)}$$

$$= T_{\mathbb{R}}\uparrow\alpha+2.$$

limit ordinals α (i):

$$(T_{\mathbb{R}}\uparrow\alpha)^c = \left(\bigcup_{\beta<\alpha} T_{\mathbb{R}}\uparrow\beta\right)^c \qquad \text{(Definition 3.3)}$$

$$= \bigcap_{\beta<\alpha} (T_{\mathbb{R}}\uparrow\beta)^c \qquad \text{(Lemma 3.4)}$$

$$= \bigcap_{\beta<\alpha} (T_{\mathbb{R}}\downarrow\beta) \qquad \text{(induction)}$$

$$= T_{\mathbb{R}}\downarrow\alpha;$$

(ii):
$$(T_{\mathbb{R}}\downarrow\alpha)^c = \left(\bigcap_{\beta<\alpha} T_{\mathbb{R}}\downarrow\beta\right)^c = \left(\bigcap_{\beta<\alpha} (T_{\mathbb{R}}\uparrow\beta)^c\right)^c \qquad \text{(induction)}$$

$$= \left(\bigcup_{\beta<\alpha} (T_{\mathbb{R}}\uparrow\beta)^c\right)^c \qquad \text{(Lemma 3.4)}$$

$$= T_{\mathbb{R}}(T_{\mathbb{R}}\uparrow\alpha) = (T_{\mathbb{R}}\uparrow\alpha+1). \qquad \square$$

Corollary 3.6. *For all ordinals we have* $(\bigcap_{\beta<\alpha} T_{\mathbb{R}}\downarrow\beta)^c = \bigcup_{\beta<\alpha} (T_{\mathbb{R}}\uparrow\beta)^c$.

The proof of Corollary 3.6 follows immediately from Theorem 3.5. Apparently, we needed an inductive argument for this result, whereas Lemma 3.4 can be proved directly from the definitions. Note that for the closure ordinal α of $T_{\mathbb{R}}$ we also have $\mathrm{gfp}(T_{\mathbb{R}}) = T_{\mathbb{R}}\downarrow\alpha$.

We have Theorem 3.5 only because earlier we have set $T_{\mathbb{R}}\downarrow 0 = \emptyset^c$. A more natural choice would probably be to set $T_{\mathbb{R}}\downarrow 0 = R_{\max}$, in which case we work within $(R_{\max}, \subseteq)$ instead of the smaller lattice $(\emptyset^c, \subseteq)$. Fortunately, however, it turns out that the fixed points of $T_{\mathbb{R}}$ in both lattices coincide and thus both lattices give rise to different iterations approaching the same fixed points.

Proposition 3.7. *The greatest fixed points of $T_{\mathbb{R}}$ in both lattices $(R_{\max}, \subseteq)$ and $(\emptyset^c, \subseteq)$ are equal.*

Proof. Let gfp($T_{\mathbb{R}}$) denote the greatest fixed point with respect to $(R_{\max}, \subseteq)$. Since $\emptyset \subseteq \text{gfp}(T_{\mathbb{R}})^c$ we find by Lemma 2.14(iv) that $\emptyset^c \supseteq \text{gfp}(T_{\mathbb{R}})^{cc} = \text{gfp}(T_{\mathbb{R}})$. Thus gfp($T_{\mathbb{R}}$) is a subset of $\emptyset^c$, and since $\emptyset^c$ is a subset of $R_{\max}$ we conclude that gfp($T_{\mathbb{R}}$) is the greatest fixed point in $(\emptyset^c, \subseteq)$. □

Proposition 3.8. *For all ordinals α we have*
- (i) *$T_{\mathbb{R}}\uparrow\alpha$ is sound,*
- (ii) *$T_{\mathbb{R}}\downarrow\alpha$ is complete.*

Proof. Since $(T_{\mathbb{R}}\uparrow\alpha)^c = T_{\mathbb{R}}\downarrow\alpha$ (Theorem 3.5), and $T_{\mathbb{R}}\downarrow\alpha \supseteq \text{gfp}(T_{\mathbb{R}}) \supseteq \text{lfp}(T_{\mathbb{R}}) \supseteq T_{\mathbb{R}}\uparrow\alpha$, it follows that $(T_{\mathbb{R}}\uparrow\alpha)^c \supseteq T_{\mathbb{R}}\uparrow\alpha$. Hence by Lemma 2.14(i) we find that $T_{\mathbb{R}}\uparrow\alpha$ is sound. Similarly, it follows that $T_{\mathbb{R}}\downarrow\alpha$ is complete. □

Corollary 3.9. *If for some ordinal α $T_{\mathbb{R}}\uparrow\alpha = T_{\mathbb{R}}\downarrow\alpha$, then $\mathbb{R}$ is well-defined.*

In [2, 3] a similar result—in a less general form—is presented as the *stabilization lemma.* We will return to this subject later, and present an example of an explicit use of this theorem (see Example 3.17).

The proof of Corollary 3.9 does not use other results than the fact that $T_{\mathbb{R}}$ is monotonic and that the least and greatest fixed points of $T_{\mathbb{R}}$ can be found from the closure ordinal of $T_{\mathbb{R}}$. In [4] a different priority mechanism is used, in order to model the depth-first search strategy in PROLOG, which starts from a different notion of a *correct* rewrite. Since, however, the closure map c is still antimonotonic—making $T_{\mathbb{R}}$ monotonic—we still have Corollary 3.9 for this particular case.

Example 3.10. Consider the PRS $\mathbb{R}_n$ in Table 9 which has n unary function symbols $A_1, \ldots, A_n$ in its signature, together with two constants 0 and 1. Consider the rewrites, denoted by x_k: $A_kA_{k-1}\ldots A_1(0) \to 0$ $(1 \leq k \leq n)$, then we can make the following observation.

Observation. *Let S be a rewrite set such that $A_kA_{k-1}\ldots A_1(0) \overset{S}{\twoheadrightarrow} 0$; then this reduction sequence consists of a one-step reduction via x_k.*

Table 9.

↓	$r1$:	$A_1(0) \to 1$
	$r2$:	$A_1(x) \to 0$
↓	$r3$:	$A_2(0) \to 1$
	$r4$:	$A_2A_1(x) \to 0$
		⋮
↓	$r2n-1$:	$A_n(0) \to 1$
	$r2n$:	$A_nA_{n-1}\ldots A_1(x) \to 0$

Proof. First note that the application of a rule of $\mathbb{R}_n$ will eliminate at least one symbol A_i and does not introduce new function symbols. Furthermore, note that the head symbol A_k in the left-hand side of the reduction can only be eliminated via the rules $r2k-1$ or $r2k$. Now, application of $r2k-1$ will yield the normal form 1, which cannot be reduced to 0. Therefore, A_k is eliminated by application of rule $r2k$. Since, however, every reduction of a subterm in $A_kA_{k-1}\ldots A_1(0)$ will eliminate at least one function symbol, $r2k$ has to be applied immediately (otherwise none of its reducts can develop to an *r2k-redex*). Hence the reduction is a one-step reduction via x_k (and thus $x_k \in S$). ☐

Corollary. *For all rewrite sets S and all* $1 \leqslant k < n$, $x_{k+1} \in S^c$ *iff* $x_k \notin S$.

Its proof follows from the observation above and the fact that $\{x_k\}$ is an obstruction for x_{k+1}. Next, consider the rewrite set $c^n(\emptyset) = c(c(\ldots c(\emptyset)\ldots))$, where c is the closure map from Definition 2.13. Set $c^0(\emptyset) = \emptyset$.

Proposition. *For all* $n \geqslant 0$ *we have*

(i) $c^{2n}(\emptyset) = \{x_2, x_4, \ldots, x_{2n}\}$,

(ii) $c^{2n+1}(\emptyset) = c^{2n}(\emptyset) \cup \{x_{2n+2}, x_{2n+3}, \ldots\}$.

The proposition follows easily by induction from the corollary above. From the proposition, we can see that $T_{\mathbb{R}_{2n}}$ and $T_{\mathbb{R}_{2n+1}}$ both have closure ordinal n. One can prove that, $\mathbb{R}_{2n}$ has $T_{\mathbb{R}_{2n}}\downarrow n$ as its semantics, which in the case of $\mathbb{R}_{2n+1}$ is $T_{\mathbb{R}_{2n+1}}\uparrow n$.

Example 3.10 provides us with an example of a class of PRS's with unbound closure ordinals and thus with a nontrivial example of the "stabilization lemma", i.e. Corollary 3.9. Note that the length of the rules, the number of the rules and the number of arrows of $\mathbb{R}_n$ all increase with n.

We are now already in the position to find sufficient conditions for a PRS to be well-defined. It turns out to be a sufficient condition that the relation $\lhd\!\lhd$ (see Definition 3.1) is *well-founded* with respect to $R_{\max}$, i.e. there exists no infinite sequence $(x_i)_{i\in\omega}$ of rewrites such that for all i we have $x_i \lhd\!\lhd x_{i+1}$. From the theory developed so far, this can be proved directly as is done in the following theorem.

Theorem 3.11. *If* $\mathbb{R}$ *is a PRS such that* $\lhd\!\lhd$ *is well-founded, then it has a unique sound and complete rewrite set.*

Proof. Suppose that $\mathrm{lfp}(T_{\mathbb{R}}) \neq \mathrm{gfp}(T_{\mathbb{R}})$, then there exists some $x_1 \in \mathrm{gfp}(T_{\mathbb{R}}) - \mathrm{lfp}(T_{\mathbb{R}})$ which has an obstruction O in $\mathrm{gfp}(T_{\mathbb{R}})$. Since $(\mathrm{gfp}(T_{\mathbb{R}}))^c = \mathrm{lfp}(T_{\mathbb{R}})$ we find that this obstruction cannot be entirely in $\mathrm{lfp}(T_{\mathbb{R}})$ and therefore there exists some $x_2 \in \mathrm{gfp}(T_{\mathbb{R}}) - \mathrm{lfp}(T_{\mathbb{R}})$ such that $x_1 \lhd\!\lhd x_2$. Note that it makes no difference whether or not x_1 and x_2 are equal. Since we can repeat this procedure arbitrarily many times, $\lhd\!\lhd$ is not well-founded. Hence $\mathrm{lfp}(T_{\mathbb{R}}) = \mathrm{gfp}(T_{\mathbb{R}})$ and thus, by Corollary 3.9, taking the closure ordinal of $T_{\mathbb{R}}$ for α, $\mathbb{R}$ has a unique sound and complete rewrite set. ☐

A sufficient condition for $\lhd\!\!\lhd$ to be well-founded is that the underlying TRS of $\mathbb{R}$ is *bounded.* Consider the following definition.

Definition 3.12. (i) Let $\mathbb{R}$ be a TRS, and $R = t_0 \to t_1 \to \ldots$ a possibly infinite reduction sequence in $\mathbb{R}$. Then the reduction R is *bounded* if

$$\exists n\ \forall t_i \in R\ \ |t_i| \leqslant n \quad (|t_i| \text{ is the length in symbols of } t_i).$$

(ii) Let $\mathbb{R}$ be a TRS. Then $\mathbb{R}$ is bounded if all its reduction sequences are.
(iii) Let $\mathbb{R}$ be a PRS. Then $\mathbb{R}$ is bounded if its underlying TRS is.

Proposition 3.13. (i) *If the underlying TRS of a PRS is strongly normalizing, then it is bounded.*
(ii) *Equivalence of terms in a bounded and confluent TRS is a decidable property* (*two terms are* equivalent *if they are related by the symmetric closure of* $\twoheadrightarrow$).
(iii) *The direct sum of bounded TRS's need not be bounded.*

Proof (i): Since every term t has only finitely many reducts, the maximum length of all reducts of t is an upper bound. For (ii) and (iii), see [17] ((iii) uses a counterexample, similar to one given in [23]). □

Proposition 3.14. *If* $\mathbb{R}$ *is bounded then* $\lhd\!\!\lhd$ *is well-founded.*

Proof. Let $(x_i)_{i\in\omega}$ be an infinite sequence such that $x_i \lhd\!\!\lhd x_{i+1}$ for all i; then for every i there is an *internal* reduction from LHS(x_i) using x_{i+1}. Therefore, for some sequence of nonempty contexts $C_i[\ ]$ we have that LHS$(x_i) \twoheadrightarrow C_i[\text{LHS}(x_{i+1})]$. But then the reduction of LHS(x_1) is not bounded, since for every n it is reducible to $C_1[C_2[C_3[\ldots C_n[\text{LHS}(x_{n+1})]\ldots]]]$, which is a term with length $>n$. □

Note, that if $\mathbb{R}$ is a TRS, then $\lhd\!\!\lhd$ is well-founded since there are no obstructions. Let us consider some examples of PRS's that are not bounded.

Example 3.15. Consider the PRS $\mathbb{R}$, with rules and priorities as in Table 10. Note that

$$\text{lfp}(T_{\mathbb{R}}) = \{1 \to A(1), A(0) \to 1\}, \qquad \text{gfp}(T_{\mathbb{R}}) = R_{\max} - \{A(0) \to 0\}.$$

$\mathbb{R}$ does not have any sound and complete rewrite set since it has no other fixed points and the least and the greatest fixed point do not coincide. To see this, we show that

$$\text{gfp}(T_{\mathbb{R}}) - \text{lfp}(T_{\mathbb{R}}) = \{A^{n+2}(0) \to 0, A^n(1) \to 0\colon n > 0\}.$$

Table 10.

	$r1$:	$1 \to A(1)$
↓	$r2$:	$A(0) \to 1$
	$r3$:	$A(x) \to 0$

(i) The rewrite $x: A^n(1) \to 0$, being an instance of rule $r3$, is incorrect with respect to $\mathrm{lfp}(T_{\mathbb{R}}) \cup \{x\}$, since it allows the internal reduction: $A^n(1) \xrightarrow{\text{int},r1} A^{n+1}(1) \xrightarrow{\text{int},x} A(0)$ and $A(0)$ is a redex of $r2$ which has higher priority. Note that x is a "selfobstructor", i.e. all obstructions of x contain the rewrite x itself, and therefore x is correct with respect to $\mathrm{lfp}(T_{\mathbb{R}})$.

(ii) Since $A^{n+1}(0) \xrightarrow{\text{int},r2} A^n(1)$, the rewrite $A^n(0) \to 0$ has an obstruction via $A^n(1) \to 0$ and thus is not an element of $(\mathrm{gfp}(T_{\mathbb{R}}))^c = \mathrm{lfp}(T_{\mathbb{R}})$.

Note that $\mathrm{lfp}(T_{\mathbb{R}}) = T_{\mathbb{R}}\uparrow 1$ and $\mathrm{gfp}(T_{\mathbb{R}}) = T_{\mathbb{R}}\downarrow 1$.

Example 3.16. Consider the PRS $\mathbb{R}$, with rules and priorities as in Table 11. Note that

$$\mathrm{lfp}(T_{\mathbb{R}}) = \{1 \to A(2), 2 \to A(1), A(0) \to 1\},$$

$$\mathrm{gfp}(T_{\mathbb{R}}) = R_{\max} - \{A(0) \to 0\}$$

which can be seen as follows. We have to prove that

$$\mathrm{gfp}(T_{\mathbb{R}}) - \mathrm{lfp}(T_{\mathbb{R}}) = \{A^{n+1}(0) \to 0, A^n(1) \to 0, A^n(2) \to 0: n > 0\}.$$

(i) Note that all rewrites $A^{n+1}(0) \to 0$, $A^n(1) \to 0$, $A^n(2) \to 0$ are correct with respect to $\mathrm{lfp}(T_{\mathbb{R}})$ and thus are in $(\mathrm{lfp}(T_{\mathbb{R}}))^c = \mathrm{gfp}(T_{\mathbb{R}})$.

Table 11.

	$r1$:	$1 \to A(2)$
	$r2$:	$2 \to A(1)$
↓	$r3$:	$A(0) \to 1$
↓	$r4$:	$A(x) \to 0$

(ii) The rewrites x: $A(1) \to 0$ and y: $A(2) \to 0$ in $\mathrm{gfp}(T_{\mathbb{R}})$ are "mutual obstructors", in the sense that they both are part of an obstruction for the other: $A(1) \to 0$ is incorrect with respect to $\mathrm{gfp}(T_{\mathbb{R}})$ since $A(1) \xrightarrow{\text{int},r1} A(A(2)) \xrightarrow{\text{int},y} A(0)$, and similarly, $A(2) \to 0$ is incorrect since $A(2) \xrightarrow{\text{int},r2} A(A(1)) \xrightarrow{\text{int},x} A(0)$. Since both x and y are correct with respect to $\mathrm{lfp}(T_{\mathbb{R}})$ they are in $(\mathrm{lfp}(T_{\mathbb{R}}))^c = \mathrm{gfp}(T_{\mathbb{R}})$.

(iii) Finally, observe that $A^{n+1}(0) \xrightarrow{\text{int},r3} A^n(1)$ and $A^n(1) \xrightarrow{\text{int},r1} A^{n+1}(2)$, thus in the presence of both $A(1) \to 0$ and $A(2) \to 0$, the rewrites $A^{n+1}(0) \to 0$, $A^n(1) \to 0$, $A^n(2) \to 0$ are incorrect.

Again we have $\mathrm{lfp}(T_{\mathbb{R}}) = T_{\mathbb{R}}\uparrow 1$ and $\mathrm{gfp}(T_{\mathbb{R}}) = T_{\mathbb{R}}\downarrow 1$. Note that both S_1: $\mathrm{lfp}(T_{\mathbb{R}}) \cup \{A(1) \to 0\}$ and S_2: $\mathrm{lfp}(T_{\mathbb{R}}) \cup \{A(2) \to 0\}$ are sound. One can easily check that

$$\mathrm{lfp}(T_{\mathbb{R}}) \cup \{A^{2n+2}(0) \to 0, A^{2n+1}(1) \to 0, A^{2n+2}(2) \to 0: n \geq 0\},$$

$$\mathrm{lfp}(T_{\mathbb{R}}) \cup \{A^{2n+2}(0) \to 0, A^{2n+2}(1) \to 0, A^{2n+1}(2) \to 0: n \geq 0\}$$

are both sound and complete. They are obtained from S_1 and S_2 by repeatedly applying $T_{\mathbb{R}}$. Thus $\mathbb{R}$ has (at least) two sound and complete rewrite sets.

Example 3.17. The PRS in Table 1 (Example 2.1) is not bounded. Therefore, in order to prove that it is well-defined we cannot use Proposition 3.14. We will prove that it *is* well-defined by finding the closure ordinal α of $T_{\mathbb{R}}$ and using Corollary 3.9. Define the interpretation $[\cdot]$ from closed terms to natural numbers by

$$[0]=0, \qquad [S(t)]=\text{succ}([t]),$$
$$[P(t)]=\text{pred}([t]), \qquad [t+s]=[t]+[s],$$

where t, s are closed and pred, succ, 0 and + are the usual functions on the set of natural numbers. Then, define

$$R=\{P(0)\to 0,\ P(S(t))\to t,\ t+0\to t\colon t \text{ closed}\}$$
$$\cup\{t+s\to S(t+P(s))\colon t, s \text{ closed}, [s]\neq 0\}.$$

Claim 1. *If* $s \ {}^{R}\!\twoheadrightarrow 0$, *then* $[s]=0$.

Proof. Use induction on the formation of s. □

Claim 2. *If* $[s]=0$, *then* $s \ {}^{R}\!\twoheadrightarrow 0$.

Proof. First prove with induction on n that $\forall m, n\ S^{m}(0)+S^{n}(0) \ {}^{R}\!\twoheadrightarrow S^{m+n}(0)$. Then use this fact to show with induction on t that $\forall$closed $t\ \exists n\ t \ {}^{R}\!\twoheadrightarrow S^{n}(0)$. The claim follows from this observation and Claim 1. □

Claim 3. $T_{\mathbb{R}}\uparrow 1 = T_{\mathbb{R}}\downarrow 1$.

Proof. From Claims 1 and 2. □

By Corollary 3.9 it follows that $\mathbb{R}$ is well-defined.

The fixed-point theory presented in this section seems to provide us with some elegant tools to find a semantics (if it exists) for a PRS. There are still a few open questions that are worth presenting at the end.

Open questions. (1) Is the mapping $T_{\mathbb{R}}$ (Definition 3.2) continuous, instead of only monotonic? In other words, do we have that for all collections $(X_i)_{i\in\omega}$ of subsets of $R_{\max}$: $T_{\mathbb{R}}(\bigcup_{i\in\omega} X_i)=\bigcup_{i\in\omega} T_{\mathbb{R}}(X_i)$?

(2) Is the closure ordinal of $T_{\mathbb{R}}$ always finite? In Example 3.10 we presented an infinite sequence $(\mathbb{R}_n)_{n\in\omega}$ of PRS's with increasing closure ordinals. It is not clear whether or not there exist finite PRS's with closure ordinal ω or even larger. If this is not the case, all transfinite induction arguments can be eliminated from the proofs in this section.

(3) The stabilization lemma (Corollary 3.9) provides us with a sufficient condition for a PRS to be well-defined. Is this condition also necessary? That is, can we find a PRS, with closure ordinal α which is well-defined and such that $T_{\mathbb{R}}\uparrow\alpha\neq T_{\mathbb{R}}\downarrow\alpha$?

4. Left-linear priority rewrite systems

Up to this point, no requirement was made as to the left-linearity of the rules in a PRS. In this section, we will restrict our attention to PRS's which have left-linear rules (i.e. no left-hand side has a multiple occurrence of the same variable), in order to prove (under certain circumstances) a confluence result for them.

We expect that some confluence results can also be obtained for suitably restricted PRS's with non-left-linear rules, as in Examples 2.2 and 2.4, but we will not attempt to do so here. First we will prove a "general" theorem, namely confluence for essentially regular TRS's. Ambiguities in the rewrite rules of a TRS may be an obstacle to confluence (see, e.g., [16, 17]). Yet, we may allow the presence of ambiguities if there is some additional mechanism (such as rule priorities) which prevents the ambiguities to be actually "used". We will conceive such a "desambiguating" mechanism as a restriction of the sets R_i of rewrites r_i: $t_{i,k} \rightarrow s_{i,k}$.

In the following we write $t(x_1, \ldots, x_n)$ for an open term containing variables only from $x_1, \ldots, x_n$, but not necessarily containing all of them.

Definition 4.1. Let r: $t(x_1, \ldots, x_n) \rightarrow s(x_1, \ldots, x_n)$ be a rewrite rule, and let $t(\rho(x_1), \ldots, \rho(x_n))$ be an r-redex for some substitution ρ. Let t' be another redex occurring in some $\rho(x_i)$. Then this redex occurrence is called a *small* redex occurrence of t' in t.

Definition 4.2. Let $\mathbb{R}$ be a left-linear TRS (possibly ambiguous). Suppose that $R_{\max}$ is partitioned into "enabled" rewrites (E) and "disabled" rewrites (D): $R_{\max} = D \cup E$. Then $(\mathbb{R}, E)$ is called a *restricted* TRS.

The idea behind Definition 4.2 is that we are able to block the use of the rewrites from D, in order to avoid ambiguities. Although, formally, D is denoted as a set, in any practical implementation one may think of a rule or some other mechanism. The reduction relation defined by a restricted TRS is precisely the reduction relation induced by the set of enabled rewrites E.

In the sequel we will consider a well-defined PRS as such a restricted TRS, in the sense that its semantics is precisely its set of enabled rewrites. Note that a PRS without a semantics has no such set.

In the following definition we recall the concept of *a critical pair of terms* (see [13]), well-known in the area of Knuth-Bendix completions. Our definition will be self-contained though.

Definition 4.3. Let r: $t \rightarrow s$, r': $t' \rightarrow s'$ be two different rewrites in E (i.e. the triples (r, t, s) and (r', t', s') are different; thus we may have that, e.g., $r = r'$ and $t = t'$ or, e.g., that $r = r'$ and $s = s'$). Let r be of the form $g \rightarrow d$ (so t is an instantiation of g). Now, r: $t \rightarrow s$ and r': $t' \rightarrow s'$ together form a *critical pair of rewrites* if t' is a subterm of t (possibly equal to t) and t' is an instantiation of a nonvariable subterm of g.

Definition 4.4. E is called *closed under small redex contractions* if the following holds: Let r be a rule of the form $g(x_1, \ldots, x_n) \rightarrow d(x_1, \ldots, x_n)$ and $g(t_1, \ldots, t_n) \rightarrow$

$d(t_1, \ldots, t_n) \in E$, where all $t_1, \ldots, t_n$ are closed terms, and assume there exist (zero or more-step) reductions $t_i \overset{E}{\twoheadrightarrow} s_i$ using rewrites from E, then $g(s_1, \ldots, s_n) \rightarrow d(s_1, \ldots, s_n) \in E$.

Using the two definitions above we are now in the position to present the definition of an important property of restricted TRS's.

Definition 4.5. The restricted TRS $(\mathbb{R}, E)$ *is essentially nonambiguous* if

(1) E contains no critical pair of rewrites,

(2) E is closed under small redex contractions.

$(\mathbb{R}, E)$ is *essentially regular* if it is essentially nonambiguous and the rules are left-linear.

We now have immediately the Church–Rosser Theorem for essentially regular restricted TRS's.

Theorem 4.6. *If the restricted TRS* $(\mathbb{R}, E)$ *is essentially regular, then it is ground confluent.*

Proof. It is entirely similar to the unrestricted regular case (see, e.g., [17]). It proceeds as follows, assuming $\rightarrow$ and $\twoheadrightarrow$ to denote reductions in E (cf. Fig. 4).

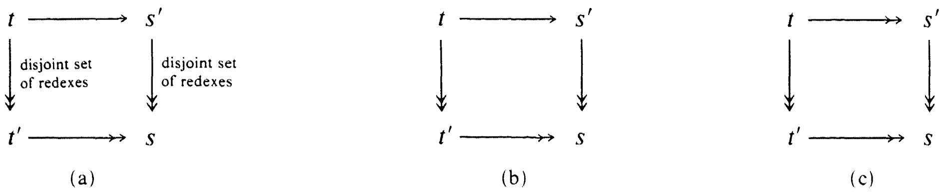

Fig. 4.

First prove that if both $t \rightarrow s'$ and $t \twoheadrightarrow t'$ by reducing a set of pairwise disjoint redexes, then an s can be found such that $t' \twoheadrightarrow s$ and $s' \twoheadrightarrow s$, the latter again via the reduction of disjoint redexes (see Fig. 4(a)). The proof follows from a straightforward analysis of cases depending on the relative position of the redex reduced in the step $t \rightarrow s'$ with respect to the disjoint redexes that are reduced in $t \twoheadrightarrow t'$. Here we use the property "essentially regular". From this fact (Fig. 4(a)) we immediately find the so-called *parallel moves lemma* (see Fig. 4(b)), which reads: $t \twoheadrightarrow s' \;\&\; t \twoheadrightarrow t' \Rightarrow \exists s\colon s' \twoheadrightarrow s \;\&\; t' \twoheadrightarrow s$. This lemma finally yields the full confluence property (see Fig. 4(c)). □

Definition 4.7. Let $\mathbb{R}$ be a PRS. We say that the ordering $<$ of the rules in $\mathbb{R}$ is a *specificity ordering* if

(i) $r < s \Leftrightarrow$ the LHS of s is a substitution instance of the LHS of r,

(ii) no ambiguities occur between incomparable rules,

(iii) ambiguities between comparable rules consist of overlaps at the roots only.

The third condition tells us that left-hand sides of rules with lower priority do not unify with proper subterms of higher priority rules. For instance,

$$L(L(x)) \to \cdots$$
$$L(x) \to \cdots$$

is not a specificity ordering since the second LHS unifies with a proper subterm of the first. Note, that condition (i) in Definition 4.7 still holds.

Theorem 4.8. *Well-defined, left-linear PRS's with specificity ordering are ground confluent.*

Proof. If $\mathbb{R}$ is a well-defined, left-linear PRS such that its priority relation is a specificity ordering, then $\mathbb{R}$ contains no critical pairs of rewrites in its semantics. To see this, assume that x and y form a critical pair of rewrites originating from the rules r and r' respectively; then, clearly, r and r' are overlapping and thus it follows by Definition 4.7(ii) and r and r' are comparable, $r > r'$ say. Furthermore, it follows from Definition 4.7(iii) that r and r' are overlapping at the root (hence $\text{LHS}(x) = \text{LHS}(y)$) and therefore y has an empty obstruction. But then y is not correct (with respect to $\emptyset$, hence with respect to the semantics of $\mathbb{R}$) and thus not in the semantics of $\mathbb{R}$.

Furthemore, the semantics R of $\mathbb{R}$ is closed under small ("internal") redex contractions, since if it were not, then for some rewrite x in R there would exist an internal reduction of LHS(x) to a term which is the left-hand side of a rewrite y, which is an instance of the same rule and which is not in R. Thus y is incorrect with respect to R and there is an internal reduction from LHS(y) to the left-hand side of a rewrite z with higher priority. Since x and y are instances of the same rule, there exists an internal reduction from LHS(x) via LHS(y) to LHS(z) using rewrites in R, and therefore x is incorrect as well. This is a contradiction, since x was in R.

Thus $\mathbb{R}$ is essentially nonambiguous, and since it is left-linear it is essentially regular. Now apply Theorem 4.6. □

Example 4.9. Consider the PRS from Example 2.1 (Table 1). Obviously, the PRS from Table 1 is left-linear and the priority relation is a specificity ordering. In Example 3.17 we have shown that it is well-defined (despite the fact that it is not bounded), and thus it is confluent. Extending this PRS with the rules for the factorial function in Table 3 (see Example 2.3) we find that the priority relation is still a specificity ordering, and since the resulting PRS is well-defined and left-linear, it is confluent.

Open question. What kind of conditions can be found for a PRS to be terminating? Clearly, a restricted TRS can turn a nonterminating TRS into a terminating one. It would therefore be interesting to find a class of terminating PRS's with a nonterminating underlying TRS.

References

[1] J. Backus, Can programming be liberated from the Von Neumann style? A functional style and its algebra programs, *Comm. ACM* **21** (1978).

[2] J.C.M. Baeten, J.A. Bergstra and J.W. Klop, Priority rewrite systems, Report CS-R8407, Centre for Mathematics and Computer Science, Amsterdam, 1984.

[3] J.C.M. Baeten, J.A. Bergstra and J.W. Klop, Term rewriting systems with priorities, in: P. Lescanne, ed., *Proc. Rewriting Techniques and Applications*, Lecture Notes in Computer Science **256** (Springer, Berlin, 1987) 83-94.

[4] J.C.M. Baeten and W.P. Weijland, A semantics for PROLOG via term rewrite rewrite systems, in: S. Kaplan and J.P. Jouannaud, eds., *Proc. 1st Internat. Workshop on Conditional Term Rewriting*, Paris, 1987, Lecture Notes in Computer Science **308** (Springer, Berlin, 1987) 3-14.

[5] J.A. Bergstra and J.W. Klop, Conditional rewrite rules: confluence and termination, *J. Comput. System Sci.* **32** (1986) 323-362.

[6] J.A. Bergstra and J.W. Klop, Algebraic specifications for parametrized data types with minimal parameter and target algebras, in: M. Nielsen and E.M. Schmidt, eds., *Proc. ICALP 1982*, Lecture Notes in Computer Science **140** (Springer, Berlin, 1982) 23-43.

[7] J.A. Bergstra and J.W. Klop, Initial algebra specifications for parametrized data types, *Elektron. Informationsverarb. Kybernet.* **19** (1983) 17-31.

[8] J.A. Bergstra and J.V. Tucker, Algebraic specifications of computable and semicomputable data types, *Theoret. Comput. Sci.* **50** (1987) 137-181.

[9] H. Ehrig and B. Mahr, *Fundamentals of Algebraic Specification 1*, EATCS Monographs on Theoretical Computer Science (Springer, Berlin, 1985).

[10] K. Futatsugi, J.A. Goguen, J.P. Jouannaud and J. Meseguer, Principles of OBJ2, in: *Proc. 12th ACM Symp. on Principles of Programming Languages*, 1985.

[11] J.A. Goguen, J.W. Thatcher and E.G. Wagner, An initial algebra approach to the specification, correctness and implementation of abstract datatypes, in: R.T. Yeh, ed., *Current Trends in Programming Methods IV, Data Structuring* (Prentice-Hall, Englewood Cliffs, NJ, 1978).

[12] J.A. Goguen, J.W. Thatcher and J.B. Wright, Abstract datatypes as initial algebras and correctness of datatype representations, in: *Proc. ACM Conf. on Computer Graphics, Pattern Recognition and Data Structure*, New York, 1975.

[13] G.Huet and D.C. Oppen, Equations and rewrite rules, a survey, in: R. Book, ed., *Formal Languages, Perspectives and Open Problems* (Academic Press, New York, 1980) 349-405.

[14] S. Kaplan, Conditional rewrite rules, *Theoret. Comput. Sci.* **33** (1984) 175-193.

[15] H.A. Klaeren, *Algebraische Spezifikation, eine Einführung*, Springer Lehrbuchreihe Informatik (Springer, Berlin, 1983).

[16] J.W. Klop, *Combinatory Reduction Systems*, Mathematical Centre Tract **127** (Amsterdam, 1980).

[17] J.W. Klop, Term rewriting systems: a tutorial, *Bull. EATCS* **32** (1987) 143-182.

[18] B. Kutzler and F. Lichtenberger, *Bibliography on Abstract Datatypes*, Informatische Fachberichte **68** (Springer, Berlin, 1983).

[19] M.J. O'Donnell, *Computing in Systems Described by Equations*, Lecture Notes in Computer Science (Springer, Berlin, 1977).

[20] M.J. O'Donnell, *Equational Logic as a Programming Language* (MIT Press, Cambridge, MA, 1985).

[21] U. Pletat, G. Engels and H.D. Ehrich, Operational semantics of algebraic specifications with conditional equations, Forschungsbericht 118/81, Abteilung Informatik, Univ. Dortmund, 1981.

[22] A. Tarski, A lattice theoretical fixed point theorem and its applications, *Pacific J. Math.* **5** (1955) 285-309.

[23] Y. Toyama, On the Church-Rosser property of the direct sum of term rewriting systems, *J. ACM* **34** (1987) 128-143.