Confluence of Term Rewriting: Theory and Automation

dissertation

by

Bertram Felgenhauer

submitted to the Faculty of Mathematics, Computer Science and Physics of the University of Innsbruck

in partial fulfillment of the requirements for the degree of Doctor of Philosophy

advisor: Univ.-Prof. Dr. Aart Middeldorp

Innsbruck, 3 February 2015

dissertation

Confluence of Term Rewriting: Theory and Automation

Bertram Felgenhauer (1016683)
bertram.felgenhauer@uibk.ac.at

3 February 2015

advisor: Univ.-Prof. Dr. Aart Middeldorp

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt durch meine eigenhändige Unterschrift, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Stellen, die wörtlich oder inhaltlich den angegebenen Quellen entnommen wurden, sind als solche kenntlich gemacht.
Die vorliegende Arbeit wurde bisher in gleicher oder ähnlicher Form noch nicht als Magister-/Master-/Diplomarbeit/Dissertation eingereicht.

Datum	Unterschrift

Abstract

The topic of this thesis is confluence of first-order term rewriting systems, both its theory and its automation. Term rewrite systems are a model of computation in which terms are successively modified by replacing instances of left-hand sides of equations by the corresponding instance of the right-hand side. Confluence is an important property of rewrite systems which is intimately connected to uniqueness of normal forms, and therefore well-definedness of functions. In the absence of termination, confluence expresses a kind of deterministic behavior: for any two computations starting at the same initial state, it is always possible to continue both of them until they reach a common state again. Like most interesting properties in computer science, confluence is an undecidable property of term rewrite systems. Therefore, determining whether a system is confluent is often challenging. Nevertheless, there are automated confluence provers that attempt to solve this task automatically. The goal of this thesis is to advance the state of the art of the field of automated confluence proving. To this end, we extend the theory of confluence in several areas to obtain confluence criteria that are both powerful and implementable.

Acknowledgments

I owe thanks to many people who made writing this thesis possible. First and foremost, thanks to my advisor Aart Middeldorp, whose logical rigor and encyclopedic knowledge has been an inspiration, and whose patience has been almost inexhaustible. I'm also indebted to the whole Computational Logic group at Innsbruck whose hospitality and friendly atmosphere made it a pleasure to work there.

The research described in this thesis was supported by FWF project P22467.

Contents

Chapter 1

Introduction

This thesis is concerned with the study and automatic checking of confluence of first-order term rewrite systems.

First-order term rewriting is a simple but powerful computational model which underlies much of declarative programming and automated theorem proving. The objects of study are term rewrite systems, which consist of rules like

$$\mathsf{f}(0, y) \to y \qquad\qquad \mathsf{f}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{f}(x, y))$$

with terms on both the left-hand and right-hand sides of each rule. Such rules can be use to modify terms. To do so, one instantiates a rule by replacing the variables x, y, ... by terms (with equal variables representing equal terms). Then, if the resulting left-hand side of a rule equals a subterm of the term, we may replace it by the corresponding right-hand side. Often there are several possible replacements, in which case we may choose any of them. For example, underlining the replaced term in each step, $\mathsf{f}(\mathsf{f}(\mathsf{s}(0), 0), 0)$ rewrites to $\mathsf{s}(0)$ in several ways:

$$\mathsf{f}(\underline{\mathsf{f}(\mathsf{s}(0), 0)}, 0) \to \mathsf{f}(\mathsf{s}(\underline{\mathsf{f}(0, 0)}), 0) \to \underline{\mathsf{f}(\mathsf{s}(0), 0)} \to \mathsf{s}(\underline{\mathsf{f}(0, 0)}) \to \mathsf{s}(0)$$
$$\mathsf{f}(\underline{\mathsf{f}(\mathsf{s}(0), 0)}, 0) \to \underline{\mathsf{f}(\mathsf{s}(\mathsf{f}(0, 0)), 0)} \to \mathsf{s}(\underline{\mathsf{f}(\mathsf{f}(0, 0), 0)}) \to \mathsf{s}(\underline{\mathsf{f}(0, 0)}) \to \mathsf{s}(0)$$
$$\mathsf{f}(\underline{\mathsf{f}(\mathsf{s}(0), 0)}, 0) \to \underline{\mathsf{f}(\mathsf{s}(\mathsf{f}(0, 0)), 0)} \to \mathsf{s}(\mathsf{f}(\underline{\mathsf{f}(0, 0)}, 0)) \to \mathsf{s}(\underline{\mathsf{f}(0, 0)}) \to \mathsf{s}(0)$$

In general, such term rewrite systems can encode arbitrary programs. Given such a program, several questions present themselves:

1. Does the program terminate?
2. Are the program results reproducible, or can different runs with the same input end in irreconcilable states?

The corresponding properties are *termination* and *confluence*. The latter property ensures, in particular, that if a program terminates in a final state, then that final state is unique. Even in the simple term rewriting model of computation, all of these properties are undecidable. Nevertheless, much progress has been made towards establishing termination of term rewrite systems automatically, and many termination criteria that are useful for automation are known. Since 2006 termination proofs generated by automatic tools can be certified with the help of theorem provers.

When the research described in this thesis started, confluence of term rewriting systems had received far less attention than termination, even though the

property is of similar practical importance: where termination allows us to conclude that a given (possibly non-deterministic) algorithm will always terminate, confluence implies that the algorithm has consistent behavior; in particular, if the algorithm terminates on some given input, it will always yield the same result. At the time there was one automated confluence prover (ACP [6]) and no effort towards certification. Now, four years later, there are four tools in development for first-order term rewrite systems, namely ACP, CoLL,[1] Saigawa,[2] and our own tool CSI [78]. A number of important techniques for proving confluence and nonconfluence can be certified using the certifier CeTA [72].

Despite the automation background, the main contributions of this thesis are of theoretical nature. After some preliminaries in Chapter 2, we will explore the decreasing diagrams technique in Chapter 3. This technique can establish confluence of abstract reduction systems, which are rewrite systems whose objects have no structure (unlike terms, which may be regarded as trees.) We present a novel proof of this known result, and show that the new proof method allows the technique to be generalized, for example to Church-Rosser modulo. After this purely theoretical result, Chapter 4 is devoted to making the decreasing diagrams technique applicable to term rewrite systems, and furthermore, how the resulting criteria can be automatically applied. In Chapter 5 we revisit the famous result that confluence is a modular property for term rewrite systems, and related results. We develop a framework, called *layer systems*, to capture common proof ideas. For automation, the most useful result will be that introducing types (with subtypes) into a TRS preserves confluence in many cases. As a result, many TRSs can be split into several smaller TRSs whose confluence can be analyzed separately. In Chapter 6 we present a cubic time algorithm for deciding whether a term rewrite system without variables (a ground term rewrite system) is confluent. This is of purely practical interest; it was already known that confluence for ground term rewrite systems is decidable in polynomial time. Chapter 7 is concerned with certification of non-confluence using tree automata techniques. Finally, Chapter 8 presents conclusions and future work.

[1] `http://www.jaist.ac.jp/~s1310032/coll/`
[2] `http://www.jaist.ac.jp/project/saigawa/`

Chapter 2

Preliminaries

In this chapter we introduce basic notions of abstract rewriting, term rewriting, and tree automata, which are used throughout this thesis. Readers interested in studying these subjects are referred to [8, 70] for term rewriting and [13] for tree automata.

2.1 Abstract Reduction Systems

We consider abstract reduction systems, which are directed graphs with objects as nodes and transitions as edges. Their semantics is that from each object, it is possible to perform reductions to successor objects. In this way, one can model arbitrary computations in a very abstract way. Typically the objects are endowed with further structure; we will consider terms as objects in Section 2.2.

Definition 2.1. An *abstract reduction system* (ARS) or *rewrite relation* is a pair $\langle \mathcal{A}, \to \rangle$, where $\mathcal{A}$ is a set of objects, and $\to \subseteq \mathcal{A} \times \mathcal{A}$ is a binary relation on objects. We write $\leftarrow$, $\leftrightarrow$, $\to^n$, $\to^=$, $\to^+$, $\to^*$ for the inverse of $\to$, the symmetric closure of $\to$, the n-th power of $\to$, the reflexive closure of $\to$, the transitive closure of $\to$ and the reflexive transitive closure of $\to$, respectively.

Note that $\to^0$ is the identity relation. We often omit the set of objects and talk of a relation $\to$ as a rewrite relation. In addition to $\to$, we use other arrows like $\Rightarrow$, $\rightharpoonup$, etc. to denote abstract reduction systems, with their mirror image ($\Leftarrow$, $\leftharpoonup$, etc.) denoting their inverse. Symmetric symbols like $\mapsto$ may denote symmetric rewrite relations. Abstract reduction systems induce rewrite sequences.

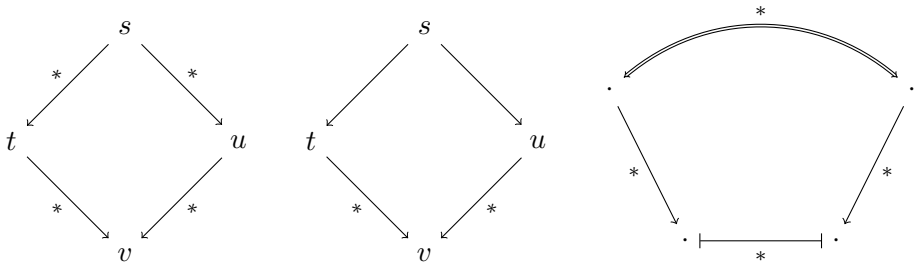
Definition 2.2. Let $\langle \mathcal{A}, \to \rangle$ be an ARS. Then $s \to t$ is called a *rewrite step*, with *source* s and *target* t. An $\mathcal{A}$*-rewrite sequence* is a sequence of objects $s_0, \ldots, s_n \in \mathcal{A}$ such that

$$s_0 \to s_1 \to \cdots \to s_n$$

We abbreviate this situation as $s_0 \to^* s_n$. The object s_n is called a *descendant* (or *successor*) of s_0; s_0 is a *predecessor* of s_n. In the case of $n = 1$ we talk about *immediate* successors and predecessors. We also say that s_0 *reaches* s_n, and that s_n is *reachable* from s_0. If s has no immediate successors then s is a *normal form*. Rewrite sequences may also be infinite, in which case we have $s_0, s_1, \cdots \in \mathcal{A}$ and $s_i \to s_{i+1}$ for all $i \in \mathbb{N}$:

$$s_0 \to s_1 \to \cdots \to s_i \to s_{i+1} \to \cdots$$

Figure 2.1: An abstract reduction system.



(a) Confluence. (b) Local confluence. (c) CR modulo.

Figure 2.2: Confluence properties of ARSs.

Two objects s and t are *joinable*, $s \downarrow t$, if $s \to^* \cdot {}^*\!\leftarrow t$. They are *convertible* if there is a *conversion* $s \leftrightarrow^* t$.

Example 2.3. Let $\mathcal{A} = \{a, b, c, d\}$ and $\to = \{(b, a), (b, c), (c, b), (c, d)\}$. Then $\langle \mathcal{A}, \to \rangle$ is an ARS, which is depicted in Figure 2.1. In this ARS, we have $a \to^* a$, and a is reachable from c because $c \to b \to a$. The objects a and d are distinct normal forms. Therefore, a and d are not joinable. However, a and b are convertible, because $a \leftarrow b \to c \to d$.

Rather than individual objects and rewrite sequences, we are usually interested in global properties of ARSs.

Definition 2.4. Let $\langle \mathcal{A}, \to \rangle$ be an ARS. Then $\to$ is

- *terminating* if it admits no infinite rewrite sequences,
- *confluent* if whenever $t \,{}^*\!\leftarrow s \to^* u$, then there is an object v such that $t \to^* v \,{}^*\!\leftarrow u$. Using relation algebra, $\to$ is confluent if
$$\overset{*}{\leftarrow} \cdot \overset{*}{\to} \subseteq \overset{*}{\to} \cdot \overset{*}{\leftarrow}$$
- *locally confluent* if
$$\leftarrow \cdot \to \subseteq \overset{*}{\to} \cdot \overset{*}{\leftarrow}$$
- *Church-Rosser* (CR) if
$$\overset{*}{\leftrightarrow} \subseteq \overset{*}{\to} \cdot \overset{*}{\leftarrow}$$

Two ARSs $\rightharpoonup$, $\rightharpoondown$ *commute* if

$$\overset{*}{\leftharpoondown} \cdot \overset{*}{\rightharpoondown} \subseteq \overset{*}{\rightharpoondown} \cdot \overset{*}{\leftharpoondown}$$

If $\mathcal{B} \subseteq \mathcal{A}$ then $\to$ is *terminating (confluent, locally confluent, Church-Rosser) on* $\mathcal{B}$ if $\to \cap\, \mathcal{B} \times \mathcal{B}$ is terminating (confluent, locally confluent, Church-Rosser).

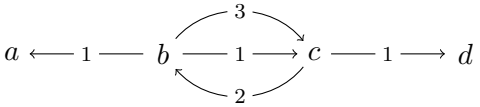


Figure 2.3: A labeled abstract reduction system.

Confluence is often depicted as a diagram as in Figure 2.2(a). In light of that picture, $t \mathrel{{}^*\!\leftarrow} s \to^* u$ is called a peak (and $t \leftarrow s \to u$ a local peak), and $t \to^* v \mathrel{{}^*\!\leftarrow} u$ a valley.

Example 2.5. Recall the ARS $\langle \mathcal{A}, \to \rangle$ from Example 2.3. It is not terminating, because there is an infinite rewrite sequence alternating between b and c:

$$b \to c \to b \to \cdots$$

It is not confluent, because a and d are not joinable, but connected by the peak $a \leftarrow b \to c \to d$. For the same reason, the ARS is not Church-Rosser. However, it is locally confluent, because both local peaks $a \leftarrow b \to c$ and $b \leftarrow c \to d$ are joinable in a valley: we have $a \leftarrow b \leftarrow c$ and $b \to c \to d$.

Lemma 2.6 (Newman's lemma [52])**.** *Let $\langle \mathcal{A}, \to \rangle$ be a terminating ARS. Then $\langle \mathcal{A}, \to \rangle$ is confluent if and only if it is locally confluent.*

Remark 2.7. The "only if" direction is not part of Newman's original lemma. But it is trivial, because confluence implies local confluence.

The main property of interest for this thesis is confluence. One of the most powerful criteria for establishing confluence of abstract rewrite systems is van Oostrom's decreasing diagrams technique [62], which will be the topic of Chapter 3. It is based on labeled ARSs.

Definition 2.8. Let L be a set of *labels*, $\mathcal{A}$ a set of objects and $\to_\alpha \subseteq \mathcal{A} \times \mathcal{A}$ a relation for each $\alpha \in L$. Then $\langle \mathcal{A}, (\to_\alpha)_{\alpha \in L} \rangle$ is a *labeled abstract reduction system*. For subsets $M \subseteq L$, we let

$$\underset{M}{\to} = \bigcup_{\alpha \in M} \underset{\alpha}{\to}$$

Finally we let $\to = \to_L$ for the unlabeled ARS underlying the labeled ARS $(\to_\alpha)_{\alpha \in L}$.

Example 2.9. We give a labeled ARS whose underlying ARS is that of Example 2.3. To this end, let $\mathcal{A} = \{a, b, c, d\}$ as before, $L = \{1, 2, 3\}$ and

$$\underset{1}{\to} = \{(b, a), (b, c), (c, d)\} \qquad \underset{2}{\to} = \{(c, b)\} \qquad \underset{3}{\to} = \{(b, c)\}$$

This labeled ARS is depicted in Figure 2.3. Note that both $b \to_1 c$ and $b \to_3 c$; the relations $\to_\alpha$ are not required to be disjoint.

In the setting of decreasing diagrams, L is equipped with a well-founded order $> \subseteq L \times L$, which is used to compute subsets $M \subseteq L$ based on labels.

Definition 2.10. Let $> \subseteq L \times L$ be a quasiorder and $\alpha, \beta \in L$. Then we define

$$\vee\alpha = \{\alpha' \in L \mid \alpha > \alpha'\} \qquad \vee\alpha\beta = \vee\alpha \cup \vee\beta$$

Definition 2.11. Let L be a set of labels equipped with a well-founded order $>$, and $(\to_\alpha)_{\alpha\in L}$ be a labeled ARS. If for every local peak $u \,{}_\alpha\!\leftarrow \cdot \to_\beta v$ can be *joined decreasingly*, i.e.,

$$u \underset{\vee\alpha}{\overset{*}{\longleftrightarrow}} \cdot \underset{\beta}{\overset{=}{\longrightarrow}} \cdot \underset{\vee\alpha\beta}{\overset{*}{\longleftrightarrow}} \cdot \underset{\alpha}{\overset{=}{\longleftarrow}} \cdot \underset{\vee\beta}{\overset{*}{\longleftrightarrow}} v$$

then $(\to_\alpha)_{\alpha\in L}$ is *locally decreasing*.

Example 2.12. We continue Example 2.9. Let L be equipped with the order $3 > 2 > 1$. Then $\vee 1 = \varnothing$, $\vee 3 = \{1, 2\}$ and $\vee 12 = \{1\}$. Consequently,

$$\underset{\vee 3}{\longrightarrow} = \{(b, a), (b, c), (c, b), (c, d)\} \qquad \underset{\vee 12}{\longrightarrow} = \{(b, a), (b, c), (c, d)\}$$

The example labeled ARS is not locally decreasing, because the local peak

$$a \underset{1}{\leftarrow} b \underset{1}{\rightarrow} c$$

cannot be joined decreasingly. Note that this local peak is distinct from

$$a \underset{1}{\leftarrow} b \underset{3}{\rightarrow} c$$

which can be joined decreasingly by $a \,{}_1\!\leftarrow b \,{}_2\!\leftarrow c$.

It is well known that locally decreasing ARSs are confluent. A proof of this fact can be found Chapter 3.

Next we present two useful facts about confluence of ARSs.

Lemma 2.13. *An ARS is confluent if and only if it is Church-Rosser.*

As a final concept for abstract rewriting we consider rewriting modulo, which is concerned with pairs of relations $\to$ and $\mapsto$, where $\mapsto$ is symmetric. A systematic discussion of rewriting modulo can be found in [56].

Definition 2.14. Let $\to$ and $\mapsto$ be rewrite relations on the same set of objects. Let $\Leftrightarrow = \leftrightarrow \cup \mapsto$. We say that $\to$ is Church-Rosser modulo $\mapsto$ if

$$\overset{*}{\Leftrightarrow} \subseteq \overset{*}{\to} \cdot \overset{*}{\mapsto} \cdot \overset{*}{\leftarrow}$$

For rewriting modulo, we distinguish between local peaks $\leftarrow \cdot \to$ and local cliffs $\leftarrow \cdot \mapsto$ or $\mapsto \cdot \to$, with the visual idea that $\mapsto$ steps are horizontal like in Figure 2.2(c).

2.2 Term Rewriting

Term rewrite systems are particular abstract reduction systems where the objects are terms and the rewrite steps are induced by certain rewrite rules. First we introduce terms.

2.2.1 Terms and Contexts

Definition 2.15. A *signature* $(\mathcal{F}, \mathcal{V})$ consists of a set $\mathcal{F}$ of function symbols and a countably infinite set of variables $\mathcal{V}$ disjoint from $\mathcal{F}$, where each function symbol $f \in \mathcal{F}$ has an associated *arity* $\mathrm{arity}(f) \in \mathbb{N}$. Functions with arity 0 are called *constants*. The set of *terms* over $(\mathcal{F}, \mathcal{V})$ is $\mathcal{T}(\mathcal{F}, \mathcal{V})$, which is defined inductively by

- $v \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ if $v \in \mathcal{V}$, and
- $f(t_1, \ldots, t_n) \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ if $f \in \mathcal{F}$, $n = \mathrm{arity}(f)$ and $t_i \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ for $1 \leqslant i \leqslant n$.

Example 2.16. Let the signature $(\mathcal{F}, \mathcal{V})$ be defined by $\mathcal{F} = \{\mathsf{c}, \mathsf{f}, \mathsf{g}\}$ and $\mathcal{V} = \{x, y, z, \ldots\}$, with arities 0 for c, 1 for f and 2 for g. Some example terms from $\mathcal{T}(\mathcal{F}, \mathcal{V})$ are

$$t_1 = x \qquad t_2 = \mathsf{c} \qquad t_3 = \mathsf{g}(\mathsf{f}(\mathsf{f}(x)), \mathsf{g}(x, y)) \qquad t_4 = \mathsf{g}(\mathsf{c}, \mathsf{f}(\mathsf{c})) \qquad t_5 = \mathsf{g}(x, y)$$

Definition 2.17. A term is *ground* if it does not contain variables. The set of ground terms over $\mathcal{F}$ is denoted by $\mathcal{T}(\mathcal{F})$. A term is *linear* if every variable occurs at most once in the term. It is *flat* if it is a variable, or a function symbol applied to constants or variables. Let $\mathrm{root}(t)$ be the *root symbol* of t, i.e., $\mathrm{root}(t) = v$ if $t = v \in \mathcal{V}$ and $\mathrm{root}(t) = f$ if $t = f(t_1, \ldots, t_n)$ for $f \in \mathcal{F}$. Given a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ define $\mathcal{V}ar(t)$ and $\mathcal{F}un(t)$ by

- $\mathcal{V}ar(t) = \{v\}$, $\mathcal{F}un(t) = \varnothing$ if $t = v \in \mathcal{V}$, and
- $\mathcal{V}ar(t) = \bigcup_{1 \leqslant i \leqslant n} \mathcal{V}ar(t_i)$ and $\mathcal{F}un(t) = \{f\} \cup \bigcup_{1 \leqslant i \leqslant n} \mathcal{F}un(t_i)$ if $t = f(t_1, \ldots, t_n)$ for $f \in \mathcal{F}$.

Example 2.18. We continue Example 2.16. The terms t_2 and t_4 are ground terms, and all listed terms except t_3 are linear. The flat terms are t_1, t_2 and t_5. We have $\mathrm{root}(t_3) = \mathsf{g}$, $\mathcal{V}ar(t_3) = \{x, y\}$ and $\mathcal{F}un(t_3) = \{\mathsf{f}, \mathsf{g}\}$.

Definition 2.19. *Positions* are strings of natural numbers. The empty string is denoted by ϵ. If $p = p'q$ then p' is a *prefix* of p, written $p' \leqslant p$, and we let $p \backslash p' = q$. The position p' is a *strict prefix* of p if $p' \leqslant p$ and $p' \neq p$. If neither $p' \leqslant p$ nor $p \leqslant p'$, then the positions p' and p' are *parallel*, $p \parallel p'$. A set of position P is *pairwise parallel* if for $p, q \in P$, $p \parallel q$ or $p = q$. Given a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ its set of positions $\mathcal{P}os(t)$ and the subterm $t|_p$ at position $p \in \mathcal{P}os(t)$ are defined by

- $\mathcal{P}os(t) = \{\epsilon\}$ if $t = v$ for $v \in \mathcal{V}$,
- $\mathcal{P}os(t) = \{\epsilon\} \cup \{ip \mid 1 \leqslant i \leqslant n, p \in \mathcal{P}os(t_i)\}$ if $t = f(t_1, \ldots, t_n)$ for $f \in \mathcal{F}$,
- $t|_\epsilon = t$, and $t|_{ip} = t_i|_p$ if $t = f(t_1, \ldots, t_n)$ for $f \in \mathcal{F}$.

If $P \subseteq \mathcal{P}os(t)$ then we let $t|_P = \{t|_p \mid p \in P\}$. For $X \subseteq \mathcal{F} \cup \mathcal{V}$ we let $\mathcal{P}os_X(t) = \{p \in \mathcal{P}os(t) \mid \mathrm{root}(t|_p) \in X$, and $|t|_X = |\mathcal{P}os_X(t)|$. For $x \in \mathcal{F} \cup \mathcal{V}$ we let $\mathcal{P}os_x(t) = \mathcal{P}os_{\{x\}}(t)$ and $|t|_x = |t|_{\{x\}}$. Let $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ be terms and $p \in \mathcal{P}os(s)$. We denote by $s[t]_p$ the result of replacing the subterm of s at position p by the term t. Formally,

- $s[t]_\epsilon = t$, and
- $s[t]_{ip} = f(t_1, \ldots, t_i[t]_p, \ldots, t_n)$ if $s = f(t_1, \ldots, t_n)$ and $f \in \mathcal{F}$.

This operation is extended to sets of pairwise parallel positions P as follows: If $P = \{p_1, \ldots, p_n\}$ we let $s[t_p]_{p \in P} = s[t_{p_1}]_{p_1} \ldots [t_{p_n}]_{p_n}$.

Example 2.20. We continue Example 2.18. The positions of t_3 are $\mathcal{P}os(t_3) = \{\epsilon, 1, 11, 111, 2, 21, 22\}$. The set $\mathcal{P}os_x(t_3) = \{111, 21\}$ indicates the occurrences of x in t_3. We have $1 < 11$, $21 \parallel 22$, and $21 \setminus 2 = 1$. The set $\{1, 21, 22\}$ consists of pairwise parallel positions. Furthermore, $t_3|_{11} = \mathsf{f}(x)$, and $t_3[t_2]_1 = \mathsf{g}(c, \mathsf{g}(x, y))$, and for $P = \{1, 2\}$, $t_3[t_p]_{p \in P} = \mathsf{g}(x, \mathsf{c})$.

Definition 2.21. A *substitution* is a function $\sigma : \mathcal{V} \to \mathcal{T}(\mathcal{F}, \mathcal{V})$ mapping variables to terms, such that its *domain* $\mathrm{dom}(\sigma) = \{v \in \mathcal{V} \mid \sigma(v) \neq v\}$ is finite. For a term t and substitution σ, $t\sigma$ denotes the term obtained by replacing all variables v in t by $\sigma(v)$:

- $t\sigma = \sigma(v)$ if $t = v \in \mathcal{V}$, and
- $t\sigma = f(t_1\sigma, \ldots, t_n\sigma)$ if $t = f(t_1, \ldots, t_n)$ for $f \in \mathcal{F}$.

A substitution σ is a *variable substitution* if $\sigma(v) \in \mathcal{V}$ for all $v \in \mathcal{V}$; it is a *renaming* if σ is a bijective variable substitution. If σ and τ are substitutions, then their *composition* $\sigma\tau$ is defined by $(\sigma\tau)(v) = v\sigma\tau$.

Example 2.22. We continue Example 2.20. Let a substitution σ be given by $\sigma(x) = t_2$, $\sigma(y) = x$, and $\sigma(v) = v$ for $v \in \mathcal{V} \setminus \{x, y\}$. Then $\mathrm{dom}(\sigma) = \{x, y\}$ and $t_3\sigma = \mathsf{g}(\mathsf{f}(\mathsf{f}(\mathsf{c})), \mathsf{g}(\mathsf{c}, x))$.

Definition 2.23. A term s *matches* a term t, $s \leqslant t$, if there exists a substitution σ such that $s\sigma = t$. A term s is a *variant* of a term t if there is a renaming σ such that $s\sigma = t$.

Example 2.24. We continue Example 2.22. The term x matches any term. We have $\mathsf{g}(x, y) \leqslant \mathsf{g}(x, x)$, but not $\mathsf{g}(x, x) \leqslant \mathsf{g}(x, y)$, because no substitution can map x to both x and y. The terms $\mathsf{g}(x, \mathsf{g}(\mathsf{c}, y))$ and $\mathsf{g}(y, \mathsf{g}(\mathsf{c}, z))$ are variants.

Lemma 2.25. *Let s, t be terms. Then s is a variant of t if and only if $s \leqslant t$ and $t \leqslant s$.*

Definition 2.26. Let $(\mathcal{F}, \mathcal{V})$ be a signature and $\Box \notin \mathcal{F} \cup \mathcal{V}$ be a fresh constant (the *hole*). We write $\mathcal{V}_\Box$ for the set of symbols $\mathcal{V} \cup \{\Box\}$. *Contexts* are terms from $\mathcal{T}(\mathcal{F} \cup \{\Box\}, \mathcal{V})$ containing an arbitrary number of holes. We let $\mathcal{C}(\mathcal{F}, \mathcal{V}) = \mathcal{T}(\mathcal{F} \cup \{\Box\}, \mathcal{V})$. Contexts are partially ordered by $\sqsubseteq$, defined inductively by

- $v \sqsubseteq v$ for $v \in \mathcal{V}$,
- $\Box \sqsubseteq C$ for $C \in \mathcal{C}(\mathcal{F}, \mathcal{V})$, and
- $f(C_1, \ldots, C_n) \sqsubseteq f(D_1, \ldots, D_n)$ if $f \in \mathcal{F}$ is a function symbol of arity n and $C_i \sqsubseteq D_i$ for $1 \leqslant i \leqslant n$.

There is a corresponding partial supremum operation, $\sqcup$, which merges contexts. The strict order $\sqsubset$ is defined by $C \sqsubset D$ if $C \sqsubseteq D$ and $C \neq D$. The minimum context with respect to $\sqsubseteq$ is the empty context $\Box$. By $C[t_1, \ldots, t_n]$ we denote the result of replacing holes in C by the terms $t_1 \ldots, t_n$ from left to right.

Example 2.27. We continue Example 2.22. Some contexts from $\mathcal{C}(\mathcal{F}, \mathcal{V})$ are

$$C_1 = \Box \qquad C_2 = \mathsf{f}(\mathsf{g}(x, \mathsf{c})) \qquad C_3 = \mathsf{f}(\mathsf{g}(\Box, \mathsf{c})) \qquad C_4 = \mathsf{g}(\mathsf{g}(\Box, \Box), \Box)$$

Note that C_2 is both a context without holes and a term. We have $C_1[t] = t$ for all terms t, $C_3[x] = C_2$, and $C_4[\mathsf{f}(x), y, z] = \mathsf{g}(\mathsf{g}(\mathsf{f}(x), y), z)$. In the $\sqsubseteq$ order, we have $C_1 \sqsubset C_3 \sqsubset C_4$, while C_4 is incomparable to C_2 and C_3.

2.2.2 Term Rewrite Systems

With the necessary operations on terms in place, we can now introduce term rewrite systems.

Definition 2.28. A *rewrite rule* over a signature $(\mathcal{F}, \mathcal{V})$ is a pair (ℓ, r) written $\ell \to r$ (and occasionally $\ell \approx r$) such that $\ell \notin \mathcal{V}$ and $\mathcal{V}ar(\ell) \supseteq \mathcal{V}ar(r)$. A rule $\ell \to r$ is a *variant* of a rule $\ell' \to r'$ if there is a renaming σ such that $\ell\sigma = \ell'$ and $r\sigma = r'$. A *term rewrite system* (TRS) $\mathcal{R}$ over a signature $(\mathcal{F}, \mathcal{V})$ is a set of rewrite rules over $(\mathcal{F}, \mathcal{V})$. A TRS $\mathcal{R}$ defines a rewrite relation $\to_{\mathcal{R}}$ on terms as follows. If $C \in \mathcal{C}(\mathcal{F}, \mathcal{V})$ is a context with a single hole, $\ell \to r \in \mathcal{R}$, and σ is a substitution, then

$$C[\ell\sigma] \to_{\mathcal{R}} C[r\sigma]$$

We also write $C[\ell\sigma] \to_{p,\ell\to r} C[r\sigma]$ in order to indicate the position p of the hole of C and the rule being used in the rewrite step.

Example 2.29. Let $\mathcal{F} = \{\mathsf{f}, \mathsf{g}, \mathsf{c}\}$ with f of arity 2, g of arity 1 and c a constant. Then $\mathsf{f}(x, y) \to \mathsf{g}(z)$ and $x \to \mathsf{g}(x)$ are not rewrite rules because they violate $\mathcal{V}ar(\ell) \supseteq \mathcal{V}ar(r)$ and $\ell \notin \mathcal{V}$, respectively. The set $\mathcal{R}$ consisting of the rewrite rules

$$\mathsf{f}(x, x) \to \mathsf{g}(x) \qquad\qquad \mathsf{g}(x) \to \mathsf{f}(\mathsf{c}, \mathsf{g}(\mathsf{c}))$$

is a TRS. The term $t = \mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{g}(\mathsf{c}))$ can be rewritten at the root position (corresponding to $C = \Box$), at position 1 (corresponding to $C = \mathsf{f}(\Box, \mathsf{g}(\mathsf{c}))$) or at position 2 ($C = \mathsf{f}(\mathsf{g}(\mathsf{c}), \Box)$) using the first, second and second rule of $\mathcal{R}$, respectively. We have

$$t \to_{\mathcal{R}} \mathsf{g}(\mathsf{g}(\mathsf{c})) \qquad t \to_{\mathcal{R}} \mathsf{f}(\mathsf{c}, \mathsf{g}(\mathsf{c})) \qquad t \to_{\mathcal{R}} \mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c})$$

Note that $\mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c})$ cannot be rewritten at the root position by the first rule because $\mathsf{f}(x, x)$ does not match $\mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c})$.

Given a TRS $\mathcal{R}$ over $(\mathcal{F}, \mathcal{V})$, the pair $\langle \mathcal{T}(\mathcal{F}, \mathcal{V}), \to_{\mathcal{R}} \rangle$ defines an abstract rewriting system. We say that $\mathcal{R}$ is terminating, confluent, etc. if $\to_{\mathcal{R}}$ is.

Definition 2.30. A rewrite rule $\ell \to r$ is *left-linear* if ℓ is linear, *right-linear* if r is linear, *linear* if both ℓ and r are linear, *duplicating* if $|\ell|_v < |r|_v$ for some $v \in \mathcal{V}$, *collapsing* if $r \in \mathcal{V}$. We lift properties of terms to rules as well, in the same way as linearity. For example, $\ell \to r$ is ground if both ℓ and r are ground; it is left-flat if ℓ is flat. A TRS is left-linear, right-linear, ground, or non-duplicating if all its rules are. Given a TRS $\mathcal{R}$ we denote by $\mathcal{R}_{\mathsf{d}}$ the subset of its duplicating rules and by $\mathcal{R}_{\mathsf{nd}}$ the subset of its non-duplicating rules.

Example 2.31. Continuing example 2.29, the rule $\mathsf{f}(x, x) \to \mathsf{g}(x)$ is right-linear and non-duplicating but not left-linear; $\mathsf{g}(x) \to \mathsf{f}(\mathsf{c}, \mathsf{g}(\mathsf{c}))$ is linear and also non-duplicating. Therefore, $\mathcal{R}$ is non-duplicating, right-linear but not left-linear. The rule $\mathsf{g}(x) \to x$ would be a collapsing rule. On the other hand, $\mathsf{g}(\mathsf{c}) \to \mathsf{c}$ is not a collapsing rule. It is a ground rule, and also linear and non-duplicating.

The same rule can act at the same position on different terms, for example we have

$$\mathsf{f}(\mathsf{c}, \mathsf{g}(x)) \to_{2,\mathsf{g}(x)\to x} \mathsf{f}(\mathsf{c}, x)$$

and also

$$\mathsf{f}(z, \mathsf{g}(z)) \to_{2,\mathsf{g}(x)\to x} \mathsf{f}(z, z)$$

To capture this situation, we introduce the concept of *mirroring*.

Definition 2.32. Two rewrite steps $s \to_{\mathcal{R}} t$, $s' \to_{\mathcal{R}} t'$ *mirror* each other if both steps use the same rule at the same position. This notion is extended to rewrite sequences in the obvious way: empty rewrite sequences mirror each other, and $s_0 \to_{\mathcal{R}} s_1 \to^*_{\mathcal{R}} s_n$ mirrors $t_0 \to_{\mathcal{R}} t_1 \to^*_{\mathcal{R}} t_n$ if $s_0 \to_{\mathcal{R}} s_1$ mirrors $t_0 \to_{\mathcal{R}} t_1$ and $s_1 \to^*_{\mathcal{R}} s_n$ mirrors $t_1 \to^*_{\mathcal{R}} t_n$.

Besides $\to_{\mathcal{R}}$, which applies a single rule at a single position in each step, it is also possible to do rewrite steps in parallel.

Definition 2.33. Let $\mathcal{R}$ be a TRS over the signature $(\mathcal{F}, \mathcal{V})$. *Parallel reduction* $\mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}}$ is defined inductively by

- $v \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} v$ for $v \in \mathcal{V}$,
- $\ell\sigma \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} r\sigma$ if $\ell \to r \in \mathcal{R}$ and σ is a substitution, and
- $f(s_1, \ldots, s_n) \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} f(t_1, \ldots, t_n)$ if $f \in \mathcal{F}$ is a function symbol of arity n and $s_i \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} t_i$ for $1 \leqslant i \leqslant n$.

Example 2.34. Let $\mathcal{R}$ be as in Example 2.29 and $t = \mathsf{f}(\mathsf{g}(x), \mathsf{g}(x))$. Then

$$t \underset{\mathcal{R}}{\mathrel{+\!\!\!\twoheadrightarrow}} t \qquad t \underset{\mathcal{R}}{\mathrel{+\!\!\!\twoheadrightarrow}} \mathsf{f}(\mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c}), \mathsf{g}(x)) \qquad t \underset{\mathcal{R}}{\mathrel{+\!\!\!\twoheadrightarrow}} \mathsf{f}(\mathsf{g}(x), \mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c}))$$
$$t \underset{\mathcal{R}}{\mathrel{+\!\!\!\twoheadrightarrow}} \mathsf{f}(\mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c}), \mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c})) \qquad t \underset{\mathcal{R}}{\mathrel{+\!\!\!\twoheadrightarrow}} \mathsf{g}(x)$$

where we rewrite nowhere, at position 1, at position 2, at the parallel positions 1 and 2, and at the root position, respectively.

2.2.3 Termination

Given a TRS $\mathcal{R}$ we can ask the question whether $\to_{\mathcal{R}}$ is terminating. A common approach is to prove termination by exhibiting a reduction order $>$ on terms such that $\ell > r$ for all $\ell \to r \in \mathcal{R}$.

Definition 2.35. Let $\to \subseteq \mathcal{T}(\mathcal{F}, \mathcal{V})^2$ be a relation. Then $\to$ is *monotone* if $s \to t$ implies $C[s] \to C[t]$ for all single hole contexts $s \to t$. The relation is *stable* if $s \to t$ implies $s\sigma \to t\sigma$ for all substitutions σ. A well-founded order that is monotone and stable is a *reduction order*.

Lemma 2.36. *Let $\mathcal{R}$ be a TRS. Then $\to_{\mathcal{R}}$ is a monotone and stable.*

Lemma 2.37. *A TRS $\mathcal{R}$ is terminating if and only if there is a reduction order $>$ such that $\mathcal{R} \subseteq >$.*

Numerous reduction orders are known in the literature, for example the Knuth–Bendix order [44], the lexicographic and recursive path orders [15], various approaches based on well-founded monotone algebras, e.g. [17, 48], and many more.

Example 2.38. Let $\mathcal{R}$ over $\mathcal{F} = \{\mathsf{add}, \mathsf{s}, 0\}$ consist of the rules

$$\mathsf{add}(0, y) \to y \qquad \mathsf{add}(\mathsf{s}(x), y) \to \mathsf{s}(\mathsf{add}(x, y))$$

Then the monotone $\mathcal{F}$-algebra over the natural numbers $\mathbb{N}$ given by

$$0^{\mathbb{N}} = 0 \qquad \mathsf{s}^{\mathbb{N}}(x) = x + 1 \qquad \mathsf{add}^{\mathbb{N}}(x, y) = 2x + y + 1$$

gives rise to a reduction order via the evaluation map

$$[x]^{\mathbb{N}} = x \quad [0]^{\mathbb{N}} = 0^{\mathbb{N}} \quad [\mathsf{s}(t)]^{\mathbb{N}} = \mathsf{s}^{\mathbb{N}}([t]^{\mathbb{N}}) \quad [\mathsf{add}(t_1, t_2)]^{\mathbb{N}} = \mathsf{add}^{\mathbb{N}}([t_1]^{\mathbb{N}}, [t_2]^{\mathbb{N}})$$

by letting $s > t$ if $[s]^{\mathbb{N}} > [t]^{\mathbb{N}}$ for all assignments of natural numbers to the variables. Now because

$$\begin{aligned} y + 1 = [\mathsf{add}(0, y)]^{\mathbb{N}} &> [y]^{\mathbb{N}} = y \\ 2x + y + 3 = [\mathsf{add}(\mathsf{s}(x), y)]^{\mathbb{N}} &> [\mathsf{s}(\mathsf{add}(x, y))]^{\mathbb{N}} = 2x + y + 2 \end{aligned}$$

we have $\mathcal{R} \subseteq >$, and therefore termination of $\mathcal{R}$ follows.

Another technique, which is especially important for automation, is to prove termination of TRSs incrementally by considering relative rewriting.

Definition 2.39. Let $\mathcal{R}$ and $\mathcal{S}$ be TRSs over a common signature. Then $\mathcal{R}/\mathcal{S}$ denotes a *relative TRS*, whose rewrite relation is defined as

$$\xrightarrow[\mathcal{R}/\mathcal{S}]{} = \xrightarrow[\mathcal{S}]{*} \cdot \xrightarrow[\mathcal{R}]{} \cdot \xrightarrow[\mathcal{S}]{*}$$

Note that $\mathcal{R}$ and $\mathcal{R}/\varnothing$ define the same rewrite relation. Therefore, we may speak of $\mathcal{R}$ as a relative TRS.

Definition 2.40. Let $\geqslant$ and $>$ be relations. Then $>$ and $\geqslant$ are *compatible* if $\geqslant \cdot > \cdot \geqslant \subseteq >$. If furthermore, $>$ is a well-founded order, $\geqslant$ is a preorder and both $>$ and $\geqslant$ are rewrite relations, then $(\geqslant, >)$ is a *monotone reduction pair*.

Relative termination is a powerful tool for termination analysis, because it allows for incremental termination proofs by the following theorem.

Theorem 2.41 (Geser [26])**.** *A relative TRS $\mathcal{R}/\mathcal{S}$ is terminating if $\mathcal{R} = \varnothing$ or there exists a monotone reduction pair $(\geqslant, >)$ such that $\mathcal{R} \cup \mathcal{S} \subseteq \geqslant$ and $(\mathcal{R} \setminus >)/(\mathcal{S} \setminus >)$ is terminating.* □

Example 2.42. Consider the TRS $\mathcal{R}$ over $\mathcal{F} = \{\mathsf{ack}, \mathsf{s}, 0\}$ computing the Ackermann function,

$$\begin{aligned} &(1)\ \mathsf{ack}(0, x) \to \mathsf{s}(x) && (2)\ \ \mathsf{ack}(\mathsf{s}(x), 0) \to \mathsf{ack}(x, \mathsf{s}(0)) \\ & && (3)\ \mathsf{ack}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{ack}(x, \mathsf{ack}(\mathsf{s}(x), y)) \end{aligned}$$

and the following monotone $\mathcal{F}$-algebra over the natural numbers.

$$0^{\mathbb{N}} = 2 \qquad \mathsf{s}^{\mathbb{N}}(x) = x + 1 \qquad \mathsf{ack}^{\mathbb{N}}(x, y) = x + y$$

This algebra gives rise to a reduction pair $(\geqslant, >)$ given by $s \geqslant t$ ($s > t$) if $[s]^{\mathbb{N}} \geqslant [t]^{\mathbb{N}}$ ($[s]^{\mathbb{N}} > [t]^{\mathbb{N}}$) for all possible assignments of natural numbers to the variables. Then we have

$$\begin{aligned} x + 2 = [\mathsf{ack}(0, x)]^{\mathbb{N}} &> [\mathsf{s}(x)]^{\mathbb{N}} = x + 1 \\ x + 3 = [\mathsf{ack}(\mathsf{s}(x), 0)]^{\mathbb{N}} &\geqslant [\mathsf{ack}(x, \mathsf{s}(0))]^{\mathbb{N}} = x + 3 \\ x + y + 2 = [\mathsf{ack}(\mathsf{s}(x), \mathsf{s}(y))]^{\mathbb{N}} &> [\mathsf{ack}(x, \mathsf{ack}(\mathsf{s}(x), y))]^{\mathbb{N}} = x + y + 1 \end{aligned}$$

which establishes termination of $\{(1), (3)\}/\mathcal{R}$. By Theorem 2.41, the same reduction pair reduces termination of $\mathcal{R}$ to that of the single rule (2), which is easy to establish.

2.2.4 Confluence

We now turn to some basic definitions and results for analyzing confluence of TRSs. A key notion is that of critical pairs, which are essential for determining whether a TRS is locally confluent or not.

Definition 2.43. Let s and t be terms. A substitution σ is a *unifier* of s and t if $s\sigma = t\sigma$. In this case, s and t are *unifiable*. The substitution σ is a *most general unifier* (mgu) if for any substitution τ with $s\tau = t\tau$, there is a substitution ρ such that $\sigma\rho = \tau$.

Lemma 2.44. *Let s and t be unifiable terms. Then s and t have a most general unifier. Furthermore, the mgu is unique up to renaming of variables.*

Example 2.45. Let $t_1 = \mathsf{g}(x)$, $t_2 = \mathsf{f}(x, x)$, $t_3 = \mathsf{f}(\mathsf{f}(\mathsf{a}, y), \mathsf{f}(z, \mathsf{g}(z)))$ and $t_4 = \mathsf{f}(y, \mathsf{g}(y))$. Then t_1 is not unifiable with t_2, t_3 or t_4 because they have a different root symbol. The term t_2 is unifiable with t_3; an mgu is

$$\sigma = \{x \mapsto \mathsf{f}(\mathsf{a}, \mathsf{g}(\mathsf{a})), y \mapsto \mathsf{g}(\mathsf{a}), z \mapsto \mathsf{a}\}$$

The terms t_2 and t_4 are not unifiable, because a unifier τ would have to satisfy

$$y\tau = x\tau = \mathsf{g}(y)\tau = \mathsf{g}(y\tau)$$

and $t = \mathsf{g}(t)$ has no finite solution.

Definition 2.46. Let $\mathcal{R}$ be a TRS over $(\mathcal{F}, \mathcal{V})$, and let $\ell_1 \to r_1$ and $\ell_2 \to r_2$ be variants of rules from $\mathcal{R}$ with no common variables. Let $p \in \mathcal{P}os_{\mathcal{F}}(\ell_2)$, such that ℓ_1 and $\ell_2|_p$ are unifiable with mgu σ. If $\ell_1 \to r_1$ is a variant of $\ell_2 \to r_2$, we further assume that $p \neq \epsilon$. Then $(l_1 \to r_1, p, l_2 \to r_2)_\sigma$ is a *critical overlap* which induces the *critical peak*

$$\ell_2[r_1]_p\sigma \underset{\mathcal{R}}{\longleftarrow} \ell_2\sigma \underset{\mathcal{R}}{\longrightarrow} r_2\sigma$$

The corresponding *critical pair* is $\langle \ell[r']_p\sigma, r\sigma \rangle$.

Lemma 2.47 (Critical pair lemma [34])**.** *Let $\mathcal{R}$ be a TRS. Then $\mathcal{R}$ is locally confluent if all its critical pairs are joinable by $\to_{\mathcal{R}}$.*

Combining Lemma 2.47 with Newman's lemma, we obtain the Knuth-Bendix criterion for confluence of terminating TRSs.

Lemma 2.48 (Knuth-Bendix [44])**.** *A terminating TRS is confluent if and only if all its critical pairs are joinable.*

Example 2.49. The terminating TRS made of the four rules

$$\mathsf{sub}(x,x) \to 0 \quad \mathsf{sub}(\mathsf{s}(x),\mathsf{s}(y)) \to \mathsf{sub}(x,y) \quad \mathsf{sub}(x,0) \to x \quad \mathsf{sub}(0,y) \to 0$$

has two critical pairs up to symmetry:

$$0 \underset{\mathcal{R}}{\longleftarrow} \mathsf{sub}(\mathsf{s}(x),\mathsf{s}(x)) \underset{\mathcal{R}}{\longrightarrow} \mathsf{sub}(x,x) \qquad 0 \underset{\mathcal{R}}{\longleftarrow} \mathsf{sub}(0,0) \underset{\mathcal{R}}{\longrightarrow} 0$$

Note that the second critical pair arises in three ways, from overlapping any two of the first, third or last rules. All these critical pairs are joinable (in particular, $\mathsf{sub}(x,x) \to 0$), and therefore the TRS is locally confluent. By Theorem 2.48 we conclude that the TRS is confluent.

Definition 2.50. A TRS is *orthogonal* if it is left-linear and has no critical pairs.

Lemma 2.51. *Orthogonal TRSs are confluent.*

Example 2.52. Both the TRSs for addition (Example 2.38) and for the Ackermann function (Example 2.42) are orthogonal, and therefore confluent. It is important to note that absence of critical pairs alone does not ensure confluence. This is demonstrated by the following TRS from [34]:

$$\mathsf{f}(x,x) \to \mathsf{A} \qquad \mathsf{f}(x,\mathsf{g}(x)) \to \mathsf{B} \qquad \mathsf{c} \to \mathsf{g}(\mathsf{c})$$

which has no critical pairs, but nevertheless has a non-joinable peak:

$$\mathsf{A} \underset{\mathcal{R}}{\longleftarrow} \mathsf{f}(\mathsf{c},\mathsf{c}) \underset{\mathcal{R}}{\longrightarrow} \mathsf{f}(\mathsf{c},\mathsf{g}(\mathsf{c})) \underset{\mathcal{R}}{\longrightarrow} \mathsf{B}$$

2.2.5 Redex Patterns

Definition 2.53. Let $\mathcal{R}$ be a TRS. A *redex pattern* is a pair $\pi = \langle p_\pi, \ell_\pi \to r_\pi \rangle$ consisting of a position p_π and a rewrite rule $\ell_\pi \to r_\pi \in \mathcal{R}$. A redex pattern π *matches* a term t if $t|_{p_\pi}$ is an instance of ℓ_π. If π matches t, then π and t uniquely determine a term t^π such that $t \to_{p_\pi, \ell_\pi \to r_\pi} t^\pi$. We denote this rewrite step by $t \to^\pi t^\pi$.

Note that two rewrite steps mirror each other if and only if their redex patterns are the same.

Remark 2.54. It may seem odd that the rewrite pattern ends up as the superscript of a rewrite step. The main reason for this switch of positioning is convenience: we will use redex patterns in connection with labels in Chapter 4, and those labels will be written as subscripts.

Definition 2.55. Let π_1 and π_2 be redex patterns that match a common term. They are called parallel ($\pi_1 \parallel \pi_2$) if $p_{\pi_1} \parallel p_{\pi_2}$. If $p_{\pi_2} \leqslant p_{\pi_1}$ and $p_{\pi_1} \backslash p_{\pi_2} \in \mathcal{P}os_{\mathcal{F}}(l_{\pi_2})$ then π_1 and π_2 overlap critically; otherwise they are called orthogonal ($\pi_1 \perp \pi_2$). Note that $\pi_1 \parallel \pi_2$ implies $\pi_1 \perp \pi_2$. We write $P \perp Q$ if $\pi \perp \pi'$ for all $\pi \in P$ and $\pi' \in Q$ and similarly $P \parallel Q$ if $\pi \parallel \pi'$ for all $\pi \in P$ and $\pi' \in Q$.

We will mainly use redex patterns to describe parallel rewrite steps.

Definition 2.56. We assign to each parallel rewrite step $s \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} t$ a set of redex patterns $P(s \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} t)$ as follows:

- $P(v \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} v) = \varnothing$,
- $P(\ell\sigma \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} r\sigma) = \{\langle \epsilon, \ell \to r \rangle\}$, and
- $P(f(s_1, \ldots, s_n) \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} f(t_1, \ldots, t_n))$, if $s_i \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} t_i$ for $1 \leqslant i \leqslant n$, is given by $\{\langle ip, \ell \to r \rangle \mid 1 \leqslant i \leqslant n,\ \langle p, \ell \to r \rangle \in P(s_i \mathrel{+\!\!\!\twoheadrightarrow_{\mathcal{R}}} t_i)\}$

We write $s \mathrel{+\!\!\!\twoheadrightarrow^P_{\mathcal{R}}} t$ if $P = P(s \mathrel{+\!\!\!\twoheadrightarrow^P_{\mathcal{R}}} t)$. Note that s and P determine t uniquely; we therefore denote t by s^P.

Remark 2.57. Equivalently, we could define parallel steps by starting with a set of pairwise parallel redex patterns P. Then if all redex patterns in $P = \{\pi_1, \ldots, \pi_n\}$ match a term s, they induce a parallel rewrite step $s \mathrel{+\!\!\!\twoheadrightarrow^P_{\mathcal{R}}} s^P$, where s^P denotes $s^{\pi_1 \cdots \pi_n}$. Note that because $\pi_1, \ldots, \pi_n$ are pairwise parallel, s^P is independent of their order.

Example 2.58. We revisit Example 2.31. As before, let $t = \mathsf{f}(\mathsf{g}(x), \mathsf{g}(x))$, and $\mathcal{R}$ consist of the rules

$$(1)\ \mathsf{f}(x, x) \to \mathsf{g}(x) \qquad\qquad (2)\ \mathsf{g}(x) \to \mathsf{f}(\mathsf{c}, \mathsf{g}(\mathsf{c}))$$

Then

$$t \xrightarrow[\mathcal{R}]{\varnothing} t \qquad t \xrightarrow[\mathcal{R}]{\{\langle 1,(2)\rangle\}} \mathsf{f}(\mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c}), \mathsf{g}(x))$$

$$t \xrightarrow[\mathcal{R}]{\{\langle 2,(2)\rangle\}} \mathsf{f}(\mathsf{g}(x), \mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c})) \qquad t \xrightarrow[\mathcal{R}]{\{\langle 1,(2)\rangle, \langle 2,(2)\rangle\}} \mathsf{f}(\mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c}), \mathsf{f}(\mathsf{g}(\mathsf{c}), \mathsf{c}))$$

$$t \xrightarrow[\mathcal{R}]{\{\langle \epsilon,(1)\rangle\}} \mathsf{g}(x)$$

2.3 Tree Automata

Definition 2.59. A *(bottom-up) tree automaton* $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ over a signature $\mathcal{F}$ consists of a set of states Q disjoint from $\mathcal{F}$, a set of final states $Q_f \subseteq Q$, and a set of transitions Δ of shape $f(q_1, \ldots, q_n) \to q$ where the root $f \in \mathcal{F}$ has arity n and $q, q_1, \ldots, q_n \in Q$.

For the sake of simplicity, we only consider automata without ϵ-transitions. We regard Δ as a TRS over the signature $\mathcal{F} \cup Q$, with the states as constants.

Example 2.60. Let $\mathcal{F} = \{\mathsf{F}, \mathsf{T}, \neg, \vee, \wedge\}$, $Q = \{0, 1, 2, 3\}$, $Q_f = \{1\}$ and Δ consist of the transitions

$$\begin{array}{rrrr} \mathsf{F} \to 0 & \mathsf{T} \to 1 & \neg(0) \to 1 & \neg(1) \to 0 \\ \vee(0,0) \to 0 & \vee(0,1) \to 1 & \vee(1,0) \to 1 & \vee(1,1) \to 1 \\ \wedge(0,0) \to 0 & \wedge(0,1) \to 0 & \wedge(1,0) \to 0 & \wedge(1,1) \to 1 \\ \mathsf{F} \to 2 & \neg(3) \to 3 & & \end{array}$$

Then $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ is a bottom-up tree automaton.

Definition 2.61. Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a tree automaton. A substitution σ over $\mathcal{F} \cup Q$ is a *state substitution* if $\sigma(v) \in Q$ for all $v \in \mathcal{V}$. A term t is *accepted in state* q if $t \to_\Delta^* q$; t is *accepted by* $\mathcal{A}$ if it is accepted in a final state. The *language accepted by* $\mathcal{A}$ is $\mathcal{L}(\mathcal{A}) = \{t \mid t \to_\Delta^* q \text{ for some } q \in Q_f\}$. We call $\mathcal{A}$ *deterministic* if no two rules in Δ have the same left-hand side. For convenience, we often write $\to_\mathcal{A}$ for $\to_\Delta$.

Example 2.62. Consider the automaton $\mathcal{A}$ from Example 2.60. Then the term $t = \wedge(\neg(\mathsf{F}), \mathsf{F})$ is accepted in state 0:

$$\wedge(\neg(\mathsf{F}), \mathsf{F}) \to_\Delta \wedge(\neg(0), \mathsf{F}) \to_\Delta \wedge(\neg(0), 0) \to_\Delta \wedge(1, 0) \to_\Delta 0$$

Note that t is not accepted by $\mathcal{A}$ because 0 is not a final state and $t \not\to_\Delta^* 1$. The language accepted by $\mathcal{A}$ are those terms over $\mathcal{F}$ that are true when F, T, $\neg$, $\vee$ and $\wedge$ are interpreted as false, true, negation, disjunction and conjunction, respectively. The automaton $\mathcal{A}$ is not deterministic, because there are two transitions with left-hand side F.

We recall that every tree automaton can be reduced to an equivalent automaton where all states are useful.

Definition 2.63. Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a tree automaton. We say that a state $q \in Q$ is *reachable* if $t \to_\mathcal{A}^* q$ for some term $t \in \mathcal{T}(\mathcal{F})$; $q \in Q$ is *productive* if $C[q] \to_\mathcal{A}^* q'$ for some context C and state $q' \in Q_f$. Finally, an automaton $\mathcal{A}$ is *trim* if all its states are both reachable and productive.

Proposition 2.64. *For any tree automaton $\mathcal{A}$ there is an equivalent tree automaton $\mathcal{A}'$ that is trim. If $\mathcal{A}$ is deterministic, then $\mathcal{A}'$ is also deterministic.*

Example 2.65. Consider once more the automaton $\mathcal{A}$ from Example 2.60. The states 0 and 1 are both reachable ($\mathsf{F} \to_\Delta 0$, $\mathsf{T} \to_\Delta 1$) and productive (1 is a final state, and $\neg(0) \to_\Delta 1$). The state 2 is reachable ($\mathsf{F} \to_\Delta 2$), but not productive, while 3 is neither productive nor reachable. The corresponding trim automaton is obtained by restricting the states to $\{0, 1\}$, and dropping all transitions that involve other states. The resulting automaton with 2 states and 12 transitions is actually deterministic, because the transition $\mathsf{F} \to 2$ was removed, making the left-hand sides unique.

Definition 2.66. Given a set of terms L, and a TRS $\mathcal{R}$, $\mathcal{R}(L)$ ($\mathcal{R}^*(L)$) is the set of one-step (many-step) descendants of L:

$$\mathcal{R}(L) = \{t' \mid t \in L,\, t' \in \mathcal{T}(\mathcal{F}, \mathcal{V}),\, t \to_\mathcal{R} t'\}$$
$$\mathcal{R}^*(L) = \{t' \mid t \in L,\, t' \in \mathcal{T}(\mathcal{F}, \mathcal{V}),\, t \to_\mathcal{R}^* t'\}$$

A language L is closed under rewriting by $\mathcal{R}$ if $\mathcal{R}(L) \subseteq L$.

Example 2.67. Let $\mathcal{A}$ be the automaton from Example 2.60. Then $\mathcal{L}(\mathcal{A})$ is closed under rewriting by the rule $\wedge(x, x) \to x$ (because replacing $\wedge(t, t)$ by t does not change the truth value of a propositional formula, but not closed under rewriting by the rule $\mathsf{F} \to \mathsf{T}$ (because $\neg(\mathsf{F}) \in \mathcal{L}(\mathcal{A})$ but $\neg(\mathsf{T}) \notin \mathcal{L}(\mathcal{A})$).

We will see in Chapter 7 how to decide whether $\mathcal{L}(\mathcal{A})$ is closed under rewriting by a TRS $\mathcal{R}$.

Chapter 3

Abstract Decreasing Diagrams

In this chapter we revisit the decreasing diagrams technique [62]. We present a well-founded proof order for establishing the correctness of the decreasing diagrams technique and use the same construction to obtain generalizations of decreasing diagrams for Church-Rosser modulo. Furthermore, we will investigate variations of the technique that allow labeling objects as well as steps in ARSs, whereas the standard decreasing diagrams technique only allows labeling rewrite steps.

This chapter is an extended version [21], except that the presentation of a second proof order, $\succ_{ilpo}$, which is based on a lexicographic order, has been omitted.

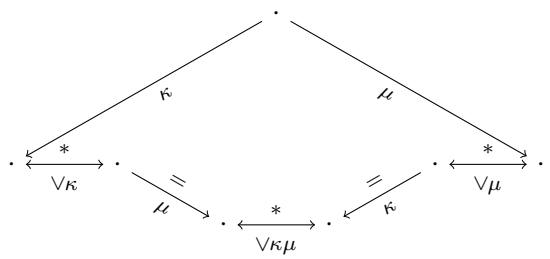
3.1 Introduction

In this chapter we revisit the decreasing diagrams technique [62] for proving confluence. This technique proves confluence of a rewrite relation by decomposing it into a labeled ARS $(\to_\kappa)_{\kappa\in L}$. Then, if every local peak $u \,{}_\kappa\!\leftarrow \cdot \to_\mu v$ can be joined decreasingly, that is, there is a joining conversion

$$u \underset{\vee\kappa}{\overset{*}{\longleftrightarrow}} \cdot \underset{\mu}{\overset{=}{\longrightarrow}} \cdot \underset{\vee\kappa\mu}{\overset{*}{\longleftrightarrow}} \cdot \underset{\kappa}{\overset{=}{\longleftarrow}} \cdot \underset{\vee\mu}{\overset{*}{\longleftrightarrow}} v$$

(see Figure 3.1(a)), the relation $\to_L$ is confluent.

We exhibit several well-founded orders on proofs (i.e., conversions) that allow us to prove termination of the proof transformation system defined by the locally decreasing diagrams. A similar approach is used in the correctness proof for completion by Bachmair and Dershowitz [9]. Rather than working on proofs



(a) Locally decreasing diagram.

$\acute{\kappa}\grave{\mu}$

$\{\kappa>\}[\grave{\mu}]\{\kappa\mu>\}[\acute{\kappa}]\{\mu>\}$

(b) Interpretations.

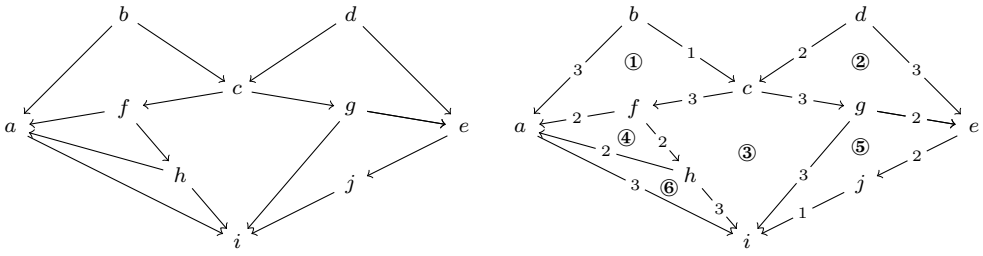
Figure 3.1: Interpreting peak and joining sequence of decreasing diagrams.

directly, we develop the orders in the setting of involutive monoids, which capture the essential structure of proofs—proofs may be concatenated and reversed.

This work is partly inspired by [38], where Jouannaud and van Oostrom define a well-founded order on proofs in order to establish that local decreasingness implies confluence. In [36], Jouannaud and Liu propose a simplified version of this proof order. The orders presented here are much simpler.

The remainder of this chapter is structured as follows: Section 3.2 presents involutive monoids. In Section 3.3 we develop orders on so-called French strings that entail the decreasing diagrams technique. Then, in Section 3.4, we extend our approach to the Church-Rosser modulo property, using an extension of French strings that we call Greek strings, leading to a generalization of a results by Ohlebusch [56] and Jouannaud and Liu [36]. In Section 3.5, we consider Bognar's point-decreasing diagrams [10] (which labels objects of ARSs rather than steps), and show that they can be obtained as a special case of standard decreasing diagrams. This leads to Section 3.6, where we explore the possibility of labeling both objects and steps of a decreasing diagram, obtaining point-step decreasing diagrams that generalize both decreasing diagrams and point-decreasing diagrams. In Section 3.7 we present two common extensions of decreasing diagrams for covering commutation and generalizing the well-founded order on labels to reduction pairs (*extended decreasingness*). Finally, we conclude in Section 3.8.

Throughout we illustrate our constructions by means of the following running example.



(a) Rewrite relation. (b) Labeled ARS.

Figure 3.2: Decomposing a rewrite relation.

Example 3.1. The rewrite relation $\to$ on objects $\{a, \ldots, j\}$ as presented in Figure 3.2(a) is the union of the family of rewrite relations $(\to_\kappa)_{\kappa \in L}$ in Figure 3.2(b), indexed by concrete labels $L = \{1, 2, 3\}$ and having individual rewrite relations:

$$\underset{1}{\to} = \{(b, c), (j, i)\}$$
$$\underset{2}{\to} = \{(d, c), (f, a), (f, h), (g, e), (h, a), (e, j)\}$$
$$\underset{3}{\to} = \{(b, a), (d, e), (c, f), (c, g), (g, i), (a, i), (h, i)\}$$

We will show how each of the transformation steps, indicated by the numbers, leading from the initial conversion $a \ {}_3\!\leftarrow b \to_1 c \ {}_2\!\leftarrow d \to_3 e$ to the final valley

$$\frac{t \to u}{t = u}\ (\text{step}) \qquad \frac{}{t = t}\ (\text{refl}) \qquad \frac{u = t}{t = u}\ (\text{sym}) \qquad \frac{t = u \quad u = v}{t = v}\ (\text{trans})$$

Figure 3.3: Equational logic for abstract rewrite relations.

$$\dfrac{\dfrac{\vdots\ P \quad \vdots\ Q}{t = u \quad u = v} \quad \vdots\ R}{t = v \qquad v = w \over t = w} \overset{\text{assoc}}{\Longrightarrow} \dfrac{\vdots\ P \quad \dfrac{\vdots\ Q \quad \vdots\ R}{u = v \quad v = w}}{t = u \qquad u = w \over t = w}$$

$$\dfrac{\dfrac{\vdots\ P \quad \vdots\ Q}{t = u \quad u = v}}{t = v \over v = t} \overset{\text{anti}}{\Longrightarrow} \dfrac{\dfrac{\vdots\ Q}{u = v} \quad \dfrac{\vdots\ P}{t = u}}{v = u \quad u = t \over v = t}$$

$$\dfrac{\vdots\ P \quad \dfrac{}{u = u}}{t = u} \overset{\text{r.id}}{\Longrightarrow} \vdots\ P \qquad \dfrac{\dfrac{}{t = t} \quad \vdots\ P}{t = u} \overset{\text{l.id}}{\Longrightarrow} \vdots\ P$$

$$\dfrac{\dfrac{\vdots\ P}{t = u}}{u = t \over t = u} \overset{\text{inv}}{\Longrightarrow} \vdots\ P \qquad \dfrac{\dfrac{}{t = t}}{t = t} \overset{\text{i.id}}{\Longrightarrow} \dfrac{}{t = t}$$

Figure 3.4: Normalizing equational logic proofs into conversions.

$a \to_3 i\ {}_1\!\leftarrow j\ {}_2\!\leftarrow e$ entails a decrease in each of our proof orders, which are based on some well-founded order $>$ on L.

Throughout we assume $>$ is a well-founded partial order on the labels L.

3.2 Involutive Monoids

A conversion $t \leftrightarrow^* u$ for a rewrite relation $\to$ is a witness to a proof that t is equal to u in the equational logic induced by $\to$, see Figure 3.3.

Remark 3.2. Because of the absence of term structure the equational logic is particularly simple: terms t, u, v are constants and the usual substitution and congruence rules are superfluous.

However, conversions correspond only to a subset of the equational logic proofs. For example, in a conversion symmetry is never applied below transitivity. In general, conversions can be identified with equational logic proofs that are in normal form with respect to the transformations in Figure 3.4.[1] Since these

[1] Reductions can be identified with proofs of rewrite logic in normal form.

transformations are confluent and terminating, every equational logic proof can be transformed into a conversion so one may restrict attention to the latter, a result known as logicality of rewriting with respect to equational logic.

Involutive monoids, see e.g. [35], are the natural algebraic structure to interpret such equational proofs *and* their transformations. In a slogan: involutive monoids are to conversions what monoids are to reductions.[2] More precisely, involutive monoids are obtained by abstracting the equalities into primitives a, b, c,..., interpreting transitivity as composition $(\cdot)$, symmetry as inversion $({}^{-1})$, reflexivity as identity (e), and equipping these with the laws in Definition 3.3 corresponding to the transformations of Figure 3.4.

Definition 3.3. A *monoid* is a (*carrier*) set equipped with an associative binary operation $(\cdot)$ and an identity element (e). An *involutive* monoid is a monoid equipped with an anti-automorphic involution $({}^{-1})$, i.e., satisfying the following laws:[3]

$$\begin{array}{rcll} (P \cdot Q) \cdot R & = & P \cdot (Q \cdot R) & \text{(assoc)} \\ P \cdot e & = & P & \text{(r.id)} \\ e \cdot P & = & P & \text{(l.id)} \end{array} \qquad \begin{array}{rcll} (P \cdot Q)^{-1} & = & Q^{-1} \cdot P^{-1} & \text{(anti)} \\ (P^{-1})^{-1} & = & P & \text{(inv)} \\ e^{-1} & = & e & \text{(i.id)} \end{array}$$

Involutive monoids are the main algebraic structure into which conversions and transformations on them will be interpreted in this chapter. Note that concatenation of conversions is a *partial* operation. Therefore, they do not form an involutive monoid by themselves. This will be the topic of the next section. We now illustrate involutive monoids first by some (mostly well-known) examples from algebra to be used later, and next by our main example, the involutive monoid of French strings.

Example 3.4. (i) The integers with addition, zero, and negation $(\mathbb{Z}, +, 0, -)$ constitute an involutive monoid. In general, any group constitutes an involutive monoid.

(ii) The monoid of natural numbers with addition and zero $(\mathbb{N}, +, 0)$ constitute an involutive monoid when equipped with the identity map, as do the multisets over L with multiset sum and the empty multiset $([L], \uplus, [])$. Commutative monoids give rise to involutive monoids in this way.

(iii) (Ordinary) strings over an alphabet L of *labels* or *letters* κ, equipped with juxtaposition, the empty string ϵ, and string reversal constitute an involutive monoid.

(iv) Natural number pairs with pointwise addition, the pair $(0, 0)$, and swapping constitute an involutive monoid. In fact, any monoid $(A, \cdot, e)$ gives rise to an involutive monoid on $A \times A$ by equipping it with pointwise composition, the pair (e, e), and swapping.

[2] For *term* rewriting the involutive monoid is to be extended with operations corresponding to the function symbols and laws for them yielding (equational) proof term algebras.

[3] Actually $e^{-1} = e$ is derivable: $e^{-1} = e \cdot e^{-1} = (e^{-1})^{-1} \cdot e^{-1} = (e \cdot e^{-1})^{-1} = (e^{-1})^{-1} = e$.

Our interpretations of conversions with respect to a family $(\to_\kappa)_{\kappa\in L}$ of rewrite relations indexed by labels in L, will all factor through an interpretation (see Definition 3.9) that only keeps the labels, equipping them with accents according to the direction (forward or backward) of the individual steps in the conversion. We call such strings of accented labels French strings.

Definition 3.5. For a given alphabet L, let a *French* letter be an accented (grave $\grave{\kappa}$ or acute $\acute{\kappa}$) letter. We will use the circumflex as in $\hat{\kappa}$ to denote a French letter having κ as label and carrying either a grave or acute accent. The set $\widehat{L}$ of *French* strings over L, i.e., strings of French letters, equipped with juxtaposition, the empty string, and mirroring $^{-1}$ given by $\grave{\kappa}^{-1} = \acute{\kappa}$ and $\acute{\kappa}^{-1} = \grave{\kappa}$, constitute an involutive monoid. The order $>$ on labels is extended to French letters: we let $\hat{\kappa} > \hat{\mu}$ and $\kappa > \hat{\mu}$ iff $\kappa > \mu$.

For instance, mirroring the French string $\acute{3}\grave{1}\acute{2}\grave{3}$ over the alphabet of Example 3.1 yields $\acute{3}\grave{2}\acute{1}\grave{3}$. In case the alphabet is a singleton set, the French letters over the alphabet are identified with the accents, denoted by $\diagdown$ and $\diagup$. French strings are to involutive monoids what (ordinary) strings are to monoids. To make this precise, we need the standard notion of a homomorphism as a structure preserving map.

Definition 3.6. A *homomorphism* from the involutive monoid $(A, \cdot, e, ^{-1})$ to the involutive monoid $(B, \cdot', e', ^{-1'})$ is a map h from A to B such that for all a, b, c in A, $h(a \cdot b) = h(a) \cdot' h(b)$, $h(e) = e'$, and $h(a^{-1}) = h(a)^{-1'}$. The homomorphism is an *isomorphism* if there exists a homomorphism that is inverse to it.

Proposition 3.7. *The involutive monoid on French strings $\widehat{L}$ is the free involutive monoid over L. That is, any map from L into the carrier of some involutive monoid, extends, via the map $\kappa \mapsto \grave{\kappa}$, uniquely to an involutive monoid homomorphism on $\widehat{L}$.*

This is a well-known fact and moreover easy to show. It is implicit in the old proofs of logicality of rewriting for the special case of (abstract) rewrite relations as noted above and explicitly proven in e.g. [35, Proposition 2].

Proof. Consider the term rewrite system obtained by orienting the laws of Definition 3.3 from left to right into term rewrite rules:

$$\begin{array}{rcl} c(c(x,y),z) & \to & c(x,c(y,z)) \\ c(x,e) & \to & x \\ c(e,x) & \to & x \end{array} \qquad \begin{array}{rcl} i(i(x)) & \to & x \\ i(c(x,y)) & \to & c(i(y),i(x)) \\ i(e) & \to & e \end{array}$$

This term rewriting system is confluent and terminating, as tools nowadays can show automatically, and has as closed normal forms[4] e and the elements of the set N given by:

$$N ::= \kappa \mid i(\kappa) \mid c(\kappa, N) \mid c(i(\kappa), N)$$

[4]Think of these closed normal forms as (empty) conversions.

Therefore, equipping $\{e\} \cup N$ with operations c, e, and i, in each case followed by taking normal forms, constitutes a free involutive monoid. This monoid is easily seen to be isomorphic to the one on French strings via the bijection between N and $\widehat{L}$ induced by $\kappa \mapsto \grave{\kappa}$. □

We conclude this section by giving some examples of homomorphisms linking up the various involutive monoids presented above.

Example 3.8. (i) Mapping a French string over L to the natural number pair of grave, acute accents in it, is a homomorphism. In turn, mapping a natural number pair to its sum is also a homomorphism. Their composition maps a French string to its *length*, e.g. $\acute{1}\grave{1}\grave{2}\acute{2}\grave{1} \mapsto (3, 2) \mapsto 5$.

(ii) Mapping a French string over L to an ordinary string over L by forgetting accents, is a homomorphism. In turn, mapping a string over L to the multiset of letters in it is also a homomorphism. Their composition maps a French string to its *multiset*, e.g. $\acute{1}\grave{1}\grave{2}\acute{2}\grave{1} \mapsto 11221 \mapsto [1, 1, 1, 2, 2]$.

(iii) Mapping a French string over L to the French string of its accents by forgetting the letters is a homomorphism.

3.3 Proof Orders and Confluence

In this section we present two novel proof orders, i.e., well-founded orders on proofs in equational logic, factoring these through their interpretation into the French string of their (accented) labels. They are shown both to be proof orders for decreasing diagrams, yielding alternative proofs showing that a locally decreasing rewrite relation is confluent. Both proof orders are flexible, in a sense to be explained in the next section.

3.3.1 Proof Orders via French Strings

A proof order is a well-founded order on conversions, i.e., on proofs in equational logic. Proof orders can be generated by proof rewrite systems as introduced in the context of completion by Bachmair and Dershowitz [9]. The objects of a proof rewrite system are conversions and its rewrite steps allow one to replace a subproof, i.e., a conversion between two terms, occurring in it by another such conversion between the same two terms.[5] The idea is to stepwise transform proofs into simpler ones, the usual goal being to obtain a valley proof (sometimes called a rewrite proof), i.e., a pair of reductions from the source and target of the original conversion, to a common reduct. Here, we adapt these ideas by factoring through an interpretation into the involutive monoid of French strings, the advantage being that they can easily be dealt with algebraically.

Definition 3.9. The *interpretation* of a conversion for an L-indexed $(\to_\kappa)_{\kappa \in L}$ family of rewrite relations, is the French string over L that is the stepwise juxtaposition of the labels in the conversion, where a label carries a grave (acute)

[5] As before, we deal here only with the special case where the terms are constants.

accent in case the corresponding step in the conversion is a forward (backward) step.

Example 3.10. The successive conversions of Example 3.1 are interpreted as the successive French strings in the following transformation, where we have underlined in each step the substring being replaced:

$$\underline{\acute{3}\grave{1}}\acute{2}3 \Rightarrow_{①} 2\acute{3}\underline{\acute{2}\grave{3}} \Rightarrow_{②} 2\underline{\acute{3}\grave{3}}\grave{2} \Rightarrow_{③} \underline{2\grave{2}}\grave{3}\acute{3}\grave{2} \Rightarrow_{④} 2\grave{3}\underline{\acute{3}\grave{2}} \Rightarrow_{⑤} \underline{2\grave{3}}\acute{1}\acute{2} \Rightarrow_{⑥} \grave{3}\acute{1}\acute{2}$$

Equipping French strings with a well-founded order or with a terminating (French string) rewrite system, gives rise to a proof order, via this interpretation. Among the well-founded orders on French strings, the monotone ones are of special interest.

Definition 3.11. A *well-founded* involutive monoid is an involutive monoid equipped with a well-founded order $\gg$ on its carrier. It is *monotone* if the algebraic operations are so with respect to the order, that is, all French string s, t, p satisfy:

1. if $s \gg t$ then $ps \gg pt$ and $sp \gg tp$.
2. if $s \gg t$ then $s^{-1} \gg t^{-1}$.

Theorem 3.12. *Let the French strings equipped with $\gg$ be a well-founded involutive monoid. Then if for all labels $\kappa, \mu \in L$ and French strings s, r over L (only over $\emptyset$ if $\gg$ is monotone, i.e., then $s = r = \epsilon$):*

$$s\acute{\kappa}\grave{\mu}r \gg s\{\kappa{>}\}[\grave{\mu}]\{\kappa\mu{>}\}[\acute{\kappa}]\{\mu{>}\}r$$

and $(\to_\kappa)_{\kappa\in L}$ is locally decreasing, then $\to_L$ has the Church-Rosser property.

In the statement of the theorem we have employed the EBNF notations [ ] and { } to express option and arbitrary repetition respectively, and used $\vec{\kappa}>$ to denote a French letter to which (at least) one letter in the vector $\vec{\kappa}$ $>$-relates. For instance, $[\grave{\mu}]$ denotes either ϵ or $\grave{\mu}$, and $\{\kappa{>}\}$ denotes an arbitrary French string of letters to which κ $>$-relates.

Proof. It suffices to show that any conversion between two objects a and b that is not yet a valley, can be transformed into another conversion between a and b that is more like a valley w.r.t. some well-founded order. If a conversion is not yet a valley, then it contains some local peak, say with interpretation $\acute{\kappa}\grave{\mu}$. By the assumption that the rewrite relation is locally decreasing, the local peak can be transformed into a conversion having interpretation of shape $\{\kappa{>}\}[\grave{\mu}]\{\kappa\mu{>}\}[\acute{\kappa}]\{\mu{>}\}$, see Figure 3.1(b). Using the assumption that $s\acute{\kappa}\grave{\mu}r \gg s\{\kappa{>}\}[\grave{\mu}]\{\kappa\mu{>}\}[\acute{\kappa}]\{\mu{>}\}r$ and well-foundedness of $\gg$, eventually a conversion without local peaks, i.e., a valley proof, is obtained.

If $\gg$ is monotone, then the comparison for $s = r = \epsilon$ extends to arbitrary strings s, r immediately. □

A well-founded order satisfying the (displayed) condition of the theorem is called a well-founded involutive monoid *for decreasing diagrams*. The two well-founded involutive monoids for decreasing diagrams to be presented below are obtained via further homomorphisms of the French strings into well-founded involutive monoids. The first one is not monotone but has 'small' images, whereas the second one is monotone but has 'large' images.

3.3.2 A Monotone Order

Definition 3.13. Let L be an alphabet with precedence $>$. We denote by $\gg_{mul}$ and $(\gg_1, \gg_2)_{lex}$ the multiset extension of $\gg$ and the lexicographic product of $\gg_1$ and $\gg_2$, respectively. The order $\gg_\bullet$ on French strings is defined recursively as follows: $s \gg_\bullet t$ iff

$$\langle s \rangle^f \; ((>, \gg_\bullet)_{lex})_{mul} \; \langle t \rangle^f$$

where $\langle s \rangle^f = [(\acute{\kappa}, q) \mid s = p\acute{\kappa}q] \cup [(\grave{\kappa}, p) \mid s = p\grave{\kappa}q]$ collects acute letters together with their suffix in s and grave letters together with their prefix in s into a multiset, and $>$ on French letters just compares their labels. For the following discussion, we define $\gg_\bullet^\Lambda = ((>, \gg_\bullet)_{lex})_{mul}$.

Note that Definition 3.13 is a proper recursive definition: The multiset extension of the lexicographic product of two orders can be computed by comparing only elements present in the compared multisets, and all French strings occurring in $\langle s \rangle^f$ are proper substrings of s.

Example 3.14. Recall the interpretations from Example 3.10. We order the set L of labels by the well-founded order $3 > 1, 2$, and consider how to compare the first interpretation to the last one. Because $\langle \epsilon \rangle^f = \varnothing$, while $\langle \grave{1}\acute{2}\grave{3} \rangle^f$ is a non-empty multiset, we have $\acute{3}\grave{1}\acute{2} \gg_\bullet \epsilon$. Therefore,

$$\langle \acute{3}\grave{1}\acute{2}\grave{3} \rangle^f = [(\acute{3}, \grave{1}\acute{2}\grave{3}), (\acute{2}, \grave{3}), (\grave{1}, \acute{3}), (\grave{3}, \acute{3}\grave{1}\acute{2})] \gg_\bullet^\Lambda [(\acute{1}, \acute{2}), (\acute{2}, \epsilon), (\grave{3}, \epsilon)] = \langle \grave{3}\acute{1}\acute{2} \rangle^f$$
$$\acute{3}\grave{1}\acute{2}\grave{3} \gg_\bullet \grave{3}\acute{1}\acute{2}$$

Next we show that $\gg_\bullet$ has all the desired properties: it is a well-founded, monotone, partial order, provided that $>$ is a well-founded order on labels.

Lemma 3.15. *If $>$ is a strict partial order on labels, then $\gg_\bullet$ is a strict partial order on French strings. Furthermore the construction is incremental: If $> \subseteq >'$ then $\gg_\bullet \subseteq \gg'_\bullet$, where $s \gg'_\bullet t$ iff $\langle s \rangle^f \; ((>', \gg'_\bullet)_{lex})_{mul} \; \langle t \rangle^f$.*

Proof. Consider the map $\Lambda_>(\gg) = \{(s, t) \mid \langle s \rangle^f \; ((>, \gg)_{lex})_{mul} \; \langle t \rangle^f\}$. By the properties of the lexicographic product and multiset extension of partial orders, $\Lambda_>(\gg)$ is monotone in $\gg$ (with respect to $\subseteq$) and maps strict partial orders to strict partial orders. Therefore, and because the union of an increasing chain (w.r.t. $\subseteq$) of strict partial orders is again a strict partial order, the least fixed point of $\Lambda_>$ exists and is a strict partial order. Inspection of the definition shows that this least fixed point equals $\gg_\bullet$. Incrementality follows because $\Lambda_>(\gg)$ is monotone in $>$. □

Lemma 3.16. *The order $\gg_\bullet$ on French strings is monotone.*

Proof. First consider monotonicity of the inverse. We have to show that $s \gg_\bullet t$ implies $s^{-1} \gg_\bullet t^{-1}$. We proceed by induction on the length of s. Note that we can express $\langle s^{-1}\rangle^f$ as $\langle s^{-1}\rangle^f = [(\hat{\kappa}^{-1}, p^{-1}) \mid (\hat{\kappa}, p) \in \langle s\rangle^f]$. Now by assumption, $\langle s\rangle^f \gg_\bullet^\Lambda \langle t\rangle^f$, and we need to show $\langle s^{-1}\rangle^f \gg_\bullet^\Lambda \langle t^{-1}\rangle^f$, that is,

$$[(\hat{\kappa}^{-1}, p^{-1}) \mid (\hat{\kappa}, p) \in \langle s\rangle^f] \gg_\bullet^\Lambda [(\hat{\mu}^{-1}, q^{-1}) \mid (\hat{\mu}, q) \in \langle t\rangle^f] \tag{3.1}$$

Since $\hat{\kappa}^{-1} > \hat{\mu}^{-1}$ iff $\hat{\kappa} > \hat{\mu}$ by definition and $p^{-1} \gg_\bullet q^{-1}$ iff $p \gg_\bullet q$ for all proper substrings p of s by the induction hypothesis, the evaluation of $\langle s\rangle^f \gg_\bullet^\Lambda \langle t\rangle^f$ can be mirrored in the comparison (3.1), which therefore holds.

Next we show that concatenation is monotone. Assume that $s \gg_\bullet t$. We need to show that $ps \gg_\bullet pt$ for arbitrary French strings p. (Once we have proved that, we know that $s \gg_\bullet t$ implies $s^{-1} \gg_\bullet t^{-1}$, then $p^{-1}s^{-1} \gg_\bullet p^{-1}t^{-1}$, and finally $sp \gg_\bullet tp$ using the monotonicity of the inverse.) It suffices to show the claim if p has length 1; induction on the length of p will complete the proof. There are two cases, $p = \acute{\kappa}$ and $p = \grave{\kappa}$. We have:

$$\begin{aligned}
\langle \acute{\kappa} s\rangle^f &= [(\acute{\mu}, p) \mid (\acute{\mu}, p) \in \langle s\rangle^f] \cup [(\grave{\mu}, \acute{\kappa} p) \mid (\grave{\mu}, p) \in \langle s\rangle^f] \cup [(\acute{\kappa}, s)] \\
\langle \grave{\kappa} s\rangle^f &= [(\acute{\mu}, p) \mid (\acute{\mu}, p) \in \langle s\rangle^f] \cup [(\grave{\mu}, \grave{\kappa} p) \mid (\grave{\mu}, p) \in \langle s\rangle^f] \cup [(\grave{\kappa}, \epsilon)]
\end{aligned} \tag{3.2}$$

Now when comparing $\langle \acute{\kappa} s\rangle^f \gg_\bullet^\Lambda \langle \acute{\kappa} t\rangle^f$ (respectively $\langle \grave{\kappa} s\rangle^f \gg_\bullet^\Lambda \langle \grave{\kappa} t\rangle^f$), we have $(\acute{\kappa}, s)\ (>, \gg_\bullet)_{lex}\ (\acute{\kappa}, t)$ by assumption ($(\grave{\kappa}, \epsilon) = (\grave{\kappa}, \epsilon)$ trivially), while the other elements of the multisets originate in $\langle s\rangle^f$ and $\langle t\rangle^f$, and their comparisons carry over to that of $\langle \hat{\kappa} s\rangle^f \gg_\bullet \langle \hat{\kappa} t\rangle^f$. Note that in the lexicographic product, comparing $(\hat{\mu}, p)$ and $(\hat{\mu}', p')$ will only require comparing p to p' if $\hat{\mu} = \hat{\mu}'$, and then we know whether $\hat{\kappa}$ was prepended to p and p', in which case we apply the induction hypothesis to that comparison, or not. Hence we conclude that $\hat{\kappa} s \gg_\bullet \hat{\kappa} t$. □

Remark 3.17. None of the previously mentioned orders are monotone. For the order $\succ_{ilpo}$ from [21], we have $\succ_{ilpo} \backslash / / / \backslash \succ_{ilpo} / \backslash \backslash \backslash \backslash$, but appending $\backslash$ to both strings results in $/ \backslash \backslash \backslash \backslash \backslash \succ_{ilpo} \backslash / / / \backslash \backslash$, i.e., the order is reversed. For the order from [38], we have $\backslash \backslash \backslash \backslash \backslash \gg \backslash / / \backslash /$ but after appending $\backslash$, we get $\backslash / / \backslash / \backslash \gg \backslash \backslash \backslash \backslash \backslash \backslash$; for [36], $\backslash \backslash \backslash / \gg \backslash / \backslash /$ but $\backslash / \backslash / / \gg \backslash \backslash \backslash / /$ after appending $/$.

Remark 3.18. It is interesting to note that while we have not explicitly stated that $\langle\cdot\rangle^f$ is a homomorphism between involutive monoids, this is in fact the case. We can define the necessary operations as follows: Let $X = \langle s\rangle^f$ and $Y = \langle t\rangle^f$. The equations

$$\begin{aligned}
X^{-1} &= [(\hat{\kappa}^- 1, p^- 1) \mid (\hat{\kappa}) \in X] \\
XY &= [(\acute{\kappa}, pt) \mid (\acute{\kappa}, p) \in X] \cup [(\acute{\kappa}, p) \mid (\acute{\kappa}, p) \in Y] \cup \\
&\quad\ [(\grave{\kappa}, p) \mid (\grave{\kappa}, p) \in X] \cup [(\grave{\kappa}, sp) \mid (\grave{\kappa}, p) \in Y]
\end{aligned}$$

define an involutive monoid on such interpretations. See also (3.1) and (3.2). The definition of XY refers to s and t, but these strings can in fact be reconstructed from X and Y.

We still have to establish well-foundedness of $\gg_\bullet$. The proof is based on simple termination [50].

Theorem 3.19. *If the precedence $>$ is well-founded, then $\gg_\bullet$ is a well-founded, monotone, partial order on French strings.*

Proof. By Lemmas 3.15 and 3.16, $\gg_\bullet$ is a strict partial order and monotone. We have to show that $\gg_\bullet$ is well-founded as well. Because its construction is incremental by Lemma 3.15, we may assume w.l.o.g. that $>$ is a (partial) well-order. We can easily see that $\gg_\bullet$ is a simplification order ([50, Definition 5.2]), if we regard French strings a terms over a unary signature as usual: monotonicity means that $\gg_\bullet$ is a stable and monotone order, while $\hat\kappa \gg_\bullet \epsilon$ and $\hat\kappa \gg_\bullet \hat\mu$ if $\hat\kappa > \hat\mu$ ensure that $>_{\mathrm{emb}} \subseteq \gg_\bullet$. Therefore, $\gg_\bullet$ is well-founded by [50, Theorem 5.3]. □

Lemma 3.20. *Let $>$ be a strict partial order. Then (recall the notation from Theorem 3.12)*

1. *$\hat\kappa \gg_\bullet \{\kappa>\}$, and*
2. *$\acute\kappa\grave\mu \gg_\bullet \{\kappa>\}[\grave\mu]\{\kappa\mu>\}[\acute\kappa]\{\mu>\}$.*

Proof. 1. Let $t \in \{\kappa>\}$. We have to establish that

$$[(\hat\kappa, \epsilon)] \gg_\bullet^\Lambda \langle t\rangle^f$$

The elements of $\langle t\rangle^f$ are pairs $(\hat\mu, p)$ with $\kappa > \mu$, i.e., smaller than $(\hat\kappa, \epsilon)$ lexicographically, so the comparison holds.

2. Let $t \in \{\kappa>\}[\grave\mu]\{\kappa\mu>\}[\acute\kappa]\{\mu>\}$. We have to establish that

$$[(\acute\kappa, \grave\mu), (\grave\mu, \acute\kappa)] \gg_\bullet^\Lambda \langle t\rangle^f$$

Most elements of $\langle t\rangle^f$ are pairs $(\hat\nu, p)$ with $\kappa > \nu$ or $\mu > \nu$. There are up to two exceptions, depending on which of the letters $\acute\kappa$ and $\grave\mu$ are present in t: $(\acute\kappa, p)$ with $p \in \{\mu>\}$, for which we have $(\acute\kappa, \grave\mu)\ (>, \gg_\bullet)_{lex}\ (\acute\kappa, p)$ using the first case, and $(\grave\mu, p)$ with $p \in \{\kappa>\}$, for which we have $(\grave\mu, \acute\kappa)\ (>, \gg_\bullet)_{lex}\ (\grave\mu, p)$ likewise. Thus, every element of $\langle t\rangle^f$ is dominated by an element of $\langle \acute\kappa\grave\mu\rangle^f$, and the comparison succeeds. □

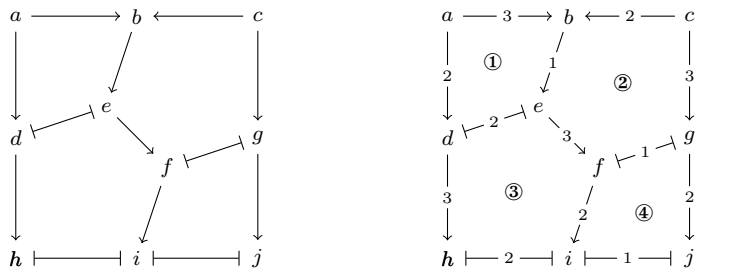
Consequently, French strings equipped with $\gg_\bullet$ are a well-founded involutive monoid for decreasing diagrams.

Theorem 3.21. *Let L be an alphabet equipped with a well-founded order $>$, and $(\to_\kappa)_{\kappa\in L}$ be a family of abstract rewrite relations. If*

$$\underset{\kappa}{\leftarrow} \cdot \underset{\mu}{\rightarrow} \subseteq \underset{\vee\kappa}{\overset{*}{\longleftrightarrow}} \cdot \underset{\mu}{\overset{=}{\rightarrow}} \cdot \underset{\vee\kappa\mu}{\overset{*}{\longleftarrow}} \cdot \underset{\kappa}{\overset{=}{\leftarrow}} \cdot \underset{\vee\mu}{\overset{*}{\longleftrightarrow}}$$

then $\to_L$ is confluent.

Proof. This follows directly from Theorem 3.12 and Lemma 3.20. □



(a) Rewrite relations $\to$, $\mapsto$.

(b) Labeled ARSs ($\to$), ($\mapsto$).

Figure 3.5: Decomposing relations with the Church-Rosser modulo property.

We have presented a well-founded, monotone order on French strings, $\gg_\bullet$. Let us briefly compare it to $\succ_{ilpo}$ from [21], which is defined in the same framework of involutive monoids. To compare two French strings by $\succ_{ilpo}$, we first map them to French *terms*, which are then compared by a lexicographic path order. The size of the French terms is linear in that of the strings, making the definition fairly lightweight. In contrast, the definition of $\gg_\bullet$ results in much bigger interpretations; unfolding it naively will result in an exponential number of comparisons. On the other hand, $\succ_{ilpo}$ is not monotone, which makes the proof of [21, Lemma 18] more tedious than that of Lemma 3.20. Also thanks to monotonicity, the validity of new rules like $\grave{1}\grave{2}\grave{2} \Rightarrow \grave{2}\acute{2}\grave{2}\grave{1}\acute{2}$ if $1 > 2$ is readily established by a direct comparison, $\grave{1}\grave{2}\grave{2} \gg_\bullet \grave{2}\acute{2}\grave{2}\grave{1}\acute{2}$. Without monotonicity, we would have to consider all possible prefixes and suffixes in the proof. Comparing strings that are longer than two symbols can be useful, as will be shown in Section 3.6.

3.4 Church-Rosser Modulo

In this section we derive a decreasing diagrams technique for Church-Rosser modulo property, in analogy to Section 3.3. In Section 3.3.1 we have seen how conversions correspond to French strings. In order to apply this idea to Church-Rosser modulo, we introduce Greek strings, an extension of French strings with self-inverse letters.

3.4.1 Decreasing Diagrams

Definition 3.22. Let L be an alphabet. For each $\kappa \in L$ there are three *Greek* letters, accented by acute, grave, or macron accents ($\acute{\kappa}$, $\grave{\kappa}$, or $\bar{\kappa}$). We use $\hat{\kappa}$ to denote a Greek letter with label κ. Mirroring letters is defined by $\acute{\kappa}^{-1} = \grave{\kappa}$ and $\bar{\kappa}^{-1} = \bar{\kappa}$. The *Greek* strings $\overline{L}$ are strings over Greek letters, which together with juxtaposition and mirroring form an involutive monoid. Any precedence $>$ on L is extended naturally to Greek letters by letting $\hat{\kappa} > \hat{\mu}$ iff $\kappa > \mu$.

The intended purpose of macron (self-inverse) letters is to represent equational steps in proofs, a natural extension of the interpretations (Definition 3.9) used

for confluence in Section 3.3.

Example 3.23. Consider the rewrite relations in Figure 3.5(b). There are several conversions proving the equivalence of d and g, using labels $L = \{\kappa, \mu, \nu\}$. We list some interpretations:

$$\underline{\acute{2}\grave{3}}\acute{2}\grave{3} \Rightarrow_{①} \bar{2}\acute{1}\underline{\acute{2}\grave{3}} \Rightarrow_{②} \bar{2}\underline{\acute{1}\grave{1}}\grave{3}\bar{1} \Rightarrow \underline{\bar{2}\grave{3}}\bar{1} \Rightarrow_{④} \grave{3}\underline{\bar{2}\acute{2}\bar{1}} \Rightarrow_{⑤} \grave{3}\bar{2}\bar{1}\acute{2}$$

We base our order on the monotone order from Section 3.3.2 (Definition 3.13).

Definition 3.24. Let L be an alphabet with precedence $>$. The order $\overline{\gg}_\bullet$ on Greek strings over L is defined by recursion as follows: $s \mathrel{\overline{\gg}_\bullet} t$ iff

$$\langle s \rangle^g \; ((>, \overline{\gg}_\bullet)_{lex})_{mul} \; \langle t \rangle^g$$

where

$$\langle s \rangle^g = [(\acute{\kappa}, q) \mid s = p\acute{\kappa}q] \cup [(\grave{\kappa}, p) \mid s = p\grave{\kappa}q] \cup [(\bar{\kappa}, \epsilon) \mid s = p\bar{\kappa}q]$$

collects acute letters together with their suffixes, grave letters together with their prefixes, and macron letters together with empty strings into a multiset. We also define $\overline{\gg}_\bullet^\Lambda = ((>, \overline{\gg}_\bullet)_{lex})_{mul}$.

Remark 3.25. We can regard any French string as a Greek string. If we do that, Definition 3.24 properly extends Definition 3.13: The map $\langle\cdot\rangle^g$ is an extension of $\langle\cdot\rangle^f$ that deals with self-inverse letters. One subtle difference is that $>$ is also extended: It compares French letters in Definition 3.13, but Greek letters in Definition 3.24.

Example 3.26. Continuing Example 3.23, we show that the second to last step is decreasing, using the order $3 > 2 > 1$ on L. In the resulting multiset comparison, it's easy to see that $(\grave{\mu}, \bar{\kappa})$ is larger than every element of the right-hand side multiset:

$$\langle \bar{2}\grave{3}\bar{1} \rangle^g = [(\bar{2}, \epsilon), (\grave{3}, \bar{2}), (\bar{1}, \epsilon)] \mathrel{\overline{\gg}_\bullet^\Lambda} [(\grave{3}, \epsilon), (\bar{2}, \epsilon), (\acute{2}, \bar{1}), (\bar{1}, \epsilon)] = \langle \grave{3}\bar{2}\acute{2}\bar{1} \rangle^g$$
$$\bar{2}\grave{3}\bar{1} \mathrel{\overline{\gg}_\bullet} \grave{3}\bar{2}\acute{2}\bar{1}$$

The order $\overline{\gg}_\bullet$ shares many properties with $\gg_\bullet$.

Theorem 3.27. *If the precedence $>$ on L is well-founded, then the order $\overline{\gg}_\bullet$ is a well-founded, monotone, partial order on Greek strings.*

Proof. The similarities between $\gg_\bullet$ and $\overline{\gg}_\bullet$ are so overwhelming that the proofs of Lemmas 3.15, 3.16 and Theorem 3.19 work with straight-forward modifications:

- Replace $\gg_\bullet$ by $\overline{\gg}_\bullet$, $\gg_\bullet^\Lambda$ by $\overline{\gg}_\bullet^\Lambda$ and $\langle\cdot\rangle^f$ by $\langle\cdot\rangle^g$ everywhere.
- In Lemma 3.15, define Λ by $\Lambda(\gg) = \{(s, t) \mid \langle s \rangle^g \; ((>, \gg)_{lex})_{mul} \; \langle t \rangle^g\}$.

- In Lemma 3.16, the expression for $\langle s^{-1}\rangle^f$ remains valid for $\langle s^{-1}\rangle^g$. For the monotonicity of concatenation, we have to consider three cases for p of length 1, $p = \acute{\kappa}$, $p = \grave{\kappa}$ and $p = \bar{\kappa}$, and we can express $\langle \hat{\kappa} s\rangle^g$ as follows:

  $$\langle \acute{\kappa} s\rangle^g = [(\hat{\mu}, p) \mid (\hat{\mu}, p) \in \langle s\rangle^g \text{ and } \hat{\mu} \neq \grave{\mu}] \cup [(\grave{\mu}, \acute{\kappa} p) \mid (\grave{\mu}, p) \in \langle s\rangle^g] \cup [(\acute{\kappa}, s)]$$
  $$\langle \grave{\kappa} s\rangle^g = [(\hat{\mu}, p) \mid (\hat{\mu}, p) \in \langle s\rangle^g \text{ and } \hat{\mu} \neq \grave{\mu}] \cup [(\grave{\mu}, \grave{\kappa} p) \mid (\grave{\mu}, p) \in \langle s\rangle^g] \cup [(\grave{\kappa}, \epsilon)]$$
  $$\langle \bar{\kappa} s\rangle^g = [(\hat{\mu}, p) \mid (\hat{\mu}, p) \in \langle s\rangle^g \text{ and } \hat{\mu} \neq \grave{\mu}] \cup [(\grave{\mu}, \bar{\kappa} p) \mid (\grave{\mu}, p) \in \langle s\rangle^g] \cup [(\bar{\kappa}, \epsilon)]$$

  When comparing $\langle \bar{\kappa} s\rangle^g$ and $\langle \bar{\kappa} t\rangle^g$, we have $(\bar{\kappa}, \epsilon) = (\bar{\kappa}, \epsilon)$, and the remaining elements of the multisets originate in $\langle s\rangle^g$ and $\langle t\rangle^g$, respectively. The remainder of the argument in Lemma 3.16 applies directly.

- Finally, the well-foundedness proof in Theorem 3.19 requires no further modifications. □

Lemma 3.28. *Let $>$ be a strict partial order. Then (recall the notation from Theorem 3.12)*

1. $\hat{\kappa} \overline{\gg}_\bullet \{\kappa>\}$ *and* $\acute{\kappa}\grave{\mu} \overline{\gg}_\bullet \{\kappa>\}[\grave{\mu}]\{\kappa\mu>\}[\acute{\kappa}]\{\mu>\}$,
2. $\bar{\kappa}\grave{\mu} \overline{\gg}_\bullet (\{\kappa>\} \cap \{\mu>\})[\grave{\mu}]\{\mu>\}\bar{\kappa}\{\mu>\}$ *(the intersection works on sets of strings), and*
3. $\bar{\kappa}\grave{\mu} \overline{\gg}_\bullet \{\kappa>\}[\grave{\mu}]\{\kappa\mu>\}$.

Proof. 1. The first item is analogous to Lemma 3.20.

2. Let $t \in (\{\kappa>\} \cap \{\mu>\})[\grave{\mu}]\{\mu>\}\bar{\kappa}\{\mu>\}$. We have to show that
$$[(\bar{\kappa}, \epsilon), (\grave{\mu}, \bar{\kappa})] \overline{\gg}^\Lambda_\bullet \langle t\rangle^g$$
Note that $(\bar{\kappa}, \epsilon) \in \langle t\rangle^g$, so all other elements of $\langle t\rangle^g$ must be smaller than $(\grave{\mu}, \bar{\kappa})$. This is true for $(\grave{\mu}, p)$ with $p \in (\{\kappa>\} \cap \{\mu>\})$ by the first item of this lemma, and all remaining pairs $(\hat{\mu}, p) \in \langle t\rangle^g$ are smaller than $(\grave{\mu}, \bar{\kappa})$ because $\mu > \mu$. We conclude that $\bar{\kappa}\grave{\mu} \overline{\gg}_\bullet t$.

3. Let $t \in \{\kappa>\}[\grave{\mu}]\{\kappa\mu>\}$. We show that
$$[(\bar{\kappa}, \epsilon), (\grave{\mu}, \bar{\kappa})] \overline{\gg}^\Lambda_\bullet \langle t\rangle^g$$
All elements $(\hat{\mu}, p)$ of $\langle t\rangle^g$ have $\kappa > \mu$ or $r > \mu$ (and are thus smaller than one of $(\grave{\mu}, \bar{\kappa})$ or $(\bar{\kappa}, \epsilon)$), with one possible exception: $(\grave{\mu}, p)$ where $p \in \{\kappa>\}$, which is smaller than $(\grave{\mu}, \bar{\kappa})$ using the first item of this lemma. Therefore, $\bar{\kappa}\grave{\mu} \overline{\gg}_\bullet t$ is true. □

Theorem 3.29. *Let L be an alphabet equipped with a well-founded order $>$. Furthermore, let $(\to_\kappa)_{\kappa\in L}$ and $(\mapsto_\kappa)_{\kappa\in L}$ be families of abstract rewrite relations, where each $\mapsto_\kappa$ is symmetric. If*

$$\underset{\kappa}{\leftarrow} \cdot \underset{\mu}{\rightarrow} \subseteq \overset{*}{\underset{\vee\kappa}{\Leftrightarrow}} \cdot \overset{=}{\underset{\mu}{\rightarrow}} \cdot \overset{*}{\underset{\vee\kappa\mu}{\Leftrightarrow}} \cdot \overset{=}{\underset{\kappa}{\leftarrow}} \cdot \overset{*}{\underset{\vee\mu}{\Leftrightarrow}}$$

and

$$\underset{\kappa}{\mapsto} \cdot \underset{\mu}{\rightarrow} \subseteq \left(\overset{*}{\underset{\vee\kappa\cap\vee\mu}{\Longleftrightarrow}} \cdot \overset{=}{\underset{\mu}{\rightarrow}} \cdot \overset{*}{\underset{\vee\mu}{\Leftrightarrow}} \cdot \underset{\kappa}{\mapsto} \cdot \overset{*}{\underset{\vee\mu}{\Leftrightarrow}} \right) \cup \left(\overset{*}{\underset{\vee\kappa}{\Leftrightarrow}} \cdot \overset{=}{\underset{\mu}{\rightarrow}} \cdot \overset{*}{\underset{\vee\kappa\mu}{\Longleftrightarrow}} \right),$$

for all $\kappa, \mu \in L$, where $\Leftrightarrow_\kappa = {}_\kappa\!\leftarrow \cup \mapsto_\kappa \cup \to_\kappa$ (see Figure 3.6), then $\to_L$ is Church-Rosser modulo $\mapsto_L$.

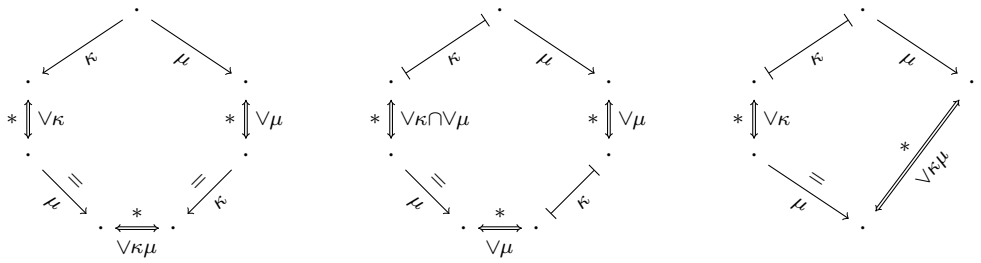


Figure 3.6: Locally decreasing diagrams for Church-Rosser modulo.

Proof. The proof follows that of Theorem 3.12. First we observe that if a conversion between two objects a and b is not a valley of shape $\to^* \cdot \mapsto^* \cdot {}^*\!\leftarrow$, then it must contain a local peak or cliff. By assumption, we can replace that peak or cliff by an alternative subproof. To show termination, we observe that the interpretation of the replacement proof is smaller than that of the peak or cliff w.r.t. $\overline{\gg}_\bullet$, by Lemma 3.28. Thanks to monotonicity this extends to the interpretations of the whole proofs. This implies termination, because $\overline{\gg}_\bullet$ is well-founded. □

The rewrite relations in Figure 3.5(a) have the Church-Rosser modulo property, because every local peak and cliff in Figure 3.5(b) can be joined in a decreasing diagram of the required shape. As an instance of Theorem 3.29 we obtain the following result by Jouannaud and Liu.

Corollary 3.30 ([36, Corollary 2.5.8])**.** *Let $(\to_\kappa)_{\kappa\in L}$ and $(\mapsto_\kappa)_{\kappa\in L}$ be families of abstract rewrite relations, where each $\mapsto_\kappa$ is symmetric. Then $\to_L$ is Church-Rosser modulo $\mapsto_L$ if for all $\kappa, \mu \in L$,*

$$\underset{\kappa}{\leftarrow} \cdot \underset{\mu}{\rightarrow} \subseteq \overset{*}{\underset{\vee\kappa}{\Longleftrightarrow}} \cdot \overset{=}{\underset{\mu}{\rightarrow}} \cdot \overset{*}{\underset{\vee\kappa\mu}{\Longleftrightarrow}} \cdot \overset{=}{\underset{\kappa}{\leftarrow}} \cdot \overset{*}{\underset{\vee\mu}{\Longleftrightarrow}} \qquad \textit{and} \qquad \underset{\kappa}{\mapsto} \cdot \underset{\mu}{\rightarrow} \subseteq \overset{=}{\underset{\mu}{\rightarrow}} \cdot \overset{*}{\underset{\vee\kappa\mu}{\Longleftrightarrow}}$$

Furthermore, Ohlebusch's main theorem of [56] is a consequence of Corollary 3.30 by labeling all $\mapsto$ steps with a minimal, fresh label $\bot$. As another instance of Theorem 3.29 we can obtain a key lemma for abstract Church-Rosser modulo from [5]:

Corollary 3.31 (Aoto and Toyama [5, Lemma 2.1])**.** *Let $(\to_\kappa)_{\kappa\in L}$ and $(\mapsto_\kappa)_{\kappa\in L}$ be families of abstract rewrite relations, where each $\mapsto_\kappa$ is symmetric. Then $\to_L$ is Church-Rosser modulo $\mapsto_L$ if for all $\kappa, \mu \in L$,*

$$\underset{\kappa}{\leftarrow} \cdot \underset{\mu}{\rightarrow} \subseteq \underset{\vee\kappa\mu}{\Longleftrightarrow} \qquad \textit{and} \qquad \underset{\kappa}{\mapsto} \cdot \underset{\mu}{\rightarrow} \subseteq \underset{\vee\kappa\mu}{\Longleftrightarrow}$$

3.4.2 Incompleteness

It is known that decreasing diagrams are complete for confluence of countable rewrite relations [70, Theorem 14.2.32]. In this section we show that no terminating proof rewrite system can be complete for proving Church-Rosser modulo. To this end, we exhibit a pair of rewrite relations $\to$, $\mapsto$ such that $\to$

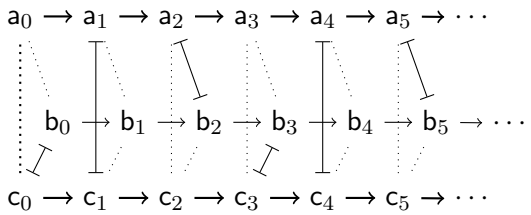


Figure 3.7: Incompleteness: The rewrite relations $\to$ and $\mapsto$.

is Church-Rosser modulo $\mapsto$, but there is no terminating proof rewrite system that has only valley proofs of shape $\to^* \cdot \mapsto^* \cdot {}^*\!\leftarrow$ as normal forms.

Remark 3.32. Terminating proof rewrite systems are exactly those which are compatible with some monotone well-founded order on proofs. We can also show termination of proof rewrite systems using non-monotone orders like $\succ_{ilpo}$ [21]. The incompleteness result of this section applies to such proofs as well.

Definition 3.33. On the set $A = \{\mathsf{a}_i, \mathsf{b}_i, \mathsf{c}_i \mid i \in \mathbb{N}\}$ we define the relations $\to$ and $\mapsto$ as follows:

1. $u_i \to u_{i+1}$ iff $u \in \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$;
2. $u_i \mapsto v_i$ iff $\{u, v\} = \{\mathsf{b}, \mathsf{c}\}$ if $i \equiv 0 \pmod 3$, $\{u, v\} = \{\mathsf{a}, \mathsf{c}\}$ if $i \equiv 1 \pmod 3$ and $\{u, v\} = \{\mathsf{a}, \mathsf{b}\}$ otherwise. (See also Figure 3.7.)

Remark 3.34. Definition 3.33 may be regarded as a simplified version of [36, Figure 1(a)]. Both examples would serve the purpose of this section, and Theorem 3.36 subsumes the incompleteness result of [36, Section 4.3].

Note that $\to$ is deterministic and that all $\mapsto^*$ equivalence classes have size 1 or 2. Together with the periodic and symmetric nature of the rewrite relations (consider mapping a_i, b_i and c_i to b_{i+1}, c_{i+1} and a_{i+1}, respectively), this restricts valley proofs to just a few possibilities:

Proposition 3.35. *1. The rewrite relation $\to$ is Church-Rosser modulo $\mapsto$.*

2. Any valley proof for a peak ${}^n\!\leftarrow \cdot \to^m$ has shape $\to^{l-n} \cdot \mapsto^* \cdot {}^{l-m}\!\leftarrow$ for some $l \geqslant n, m \geqslant 0$.

3. Any valley proof for a local cliff $\leftarrow \cdot \mapsto$ has shape $\to^{3n-1} \cdot \mapsto^+ \cdot {}^{3n}\!\leftarrow$ for some $n > 0$.

The following result establishes that no terminating proof rewrite system can be complete for Church-Rosser modulo of $\to$, $\mapsto$.

Theorem 3.36. *There is no terminating proof rewrite system for $\to$, $\mapsto$ that only rewrites local peaks and cliffs and always produces valley proofs as normal forms.*

Proof. By contradiction. Assume that we are given a terminating proof rewrite system whose normal forms are valley proofs. We show coinductively that any proof of shape

$$\mapsto \cdot \xrightarrow{n} \cdot \xleftarrow{m} \cdot \mapsto \tag{3.3}$$

with $n \not\equiv m \pmod 3$ allows an infinite proof rewrite sequence. Note that such proofs exist, for example, we have $\mathsf{b}_0 \mapsto \mathsf{c}_0 \to \mathsf{c}_1 \mapsto \mathsf{a}_1$. We may assume w.l.o.g. that $n > 0$ (if $n = 0$, then $m > 0$, and we can conclude symmetrically). Then we can rewrite the initial cliff $\mapsto \cdot \to$ to a normal form, which must be a valley proof. By Proposition 3.35, the resulting proof has shape $\to^{3k} \cdot \mapsto^{+} \cdot {}^{3k-1}\!\leftarrow \cdot \to^{n-1} \cdot {}^{m}\!\leftarrow \cdot \mapsto$ for some $k \in \mathbb{N}$. Similarly, we can reduce the new peak ${}^{3k-1}\!\leftarrow \cdot \to^{n-1}$ to a valley proof, which by Proposition 3.35 results in a proof

$$\xrightarrow{3k} \cdot \overset{+}{\mapsto} \cdot \xrightarrow{u} \cdot \overset{p}{\mapsto} \cdot \xleftarrow{v} \cdot \mapsto \tag{3.4}$$

with $u = l - 3k + 1$ and $v = l - n + 1 + m$ for some $l, p \in \mathbb{N}$. for some $l, p \in \mathbb{N}$. Let $u = l - 3k + 1$ and $v = l - n + 1 + m$. It is easy to see that $u \not\equiv v \pmod 3$. If $p = 0$, then (3.4) contains a subproof of shape (3.3), namely $\mapsto \cdot \to^{u} \cdot {}^{v}\!\leftarrow \cdot \mapsto$. If $p > 0$ and $u \not\equiv 0 \pmod 3$ then the subproof $\mapsto \cdot \to^{u} \cdot {}^{0}\!\leftarrow \cdot \mapsto$ of (3.4) has shape (3.3). Otherwise, $p > 0$ and $v \not\equiv 0 \pmod 3$, and the subproof $\mapsto \cdot \to^{0} \cdot {}^{v}\!\leftarrow \cdot \mapsto$ of (3.4) has shape (3.3). Continuing this process on the obtained subproof, we obtain an infinite proof rewrite sequence, contradicting our termination assumption. □

Remark 3.37. Note that by identifying u_i with u_{i+3} for all $i \in \mathbb{N}$ and $u \in \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ we obtain a pair of finite rewrite relations for which Theorem 3.36 still holds.

3.5 Point-Decreasing Diagrams

In this section we consider the point version of decreasing diagrams proposed by Bognar in [10], and show how it follows from ordinary decreasing diagrams. In this version, the objects (i.e., the points) rather than the steps of an abstract rewrite system are labeled. We will refer to van Oostrom's decreasing diagrams result as the *step version* of decreasing diagrams in this section. We can restate Bognar's version of local decreasingness as follows.

Definition 3.38. Let $\langle \mathcal{A}, \to \rangle$ be an abstract rewrite system and $>$ be a well-founded order on L, and $\ell : \mathcal{A} \to L$ be a function labeling the objects. We annotate steps by the labels of their target in square brackets, that is, we write $s \to_{[\ell(t)]} t$. If every local peak $t \,{}_{[\kappa]}\!\leftarrow s \to_{[\nu]} u$ with $\mu = \ell(s)$ has a joining valley

$$t \xrightarrow[[\vee\kappa\mu]]{*} \cdot \xrightarrow[[\nu]]{=} \cdot \xrightarrow[[\vee\kappa\mu\nu]]{*} \cdot \xleftarrow[[\vee\kappa\mu\nu]]{*} \cdot \xleftarrow[[\kappa]]{=} \cdot \xleftarrow[[\vee\mu\nu]]{*} u$$

then $\langle \mathcal{A}, \to \rangle$ is *locally point-decreasing*.

Bognar's main result [10, Corollary 8] states that any locally point-decreasing labeled ARS is confluent.

Theorem 3.39. *Any locally point-decreasing labeled ARS is confluent.*

Proof. Because any well-founded order can be extended to a well-order, and because locally point-decreasing diagram are preserved when the order $>$ on W is extended, we may assume that $>$ is a well-order. We label steps by pairs from $L \times \{\bot, \top\}$, ordered lexicographically, with the order $\top > \bot$ on the second component. Each step $s \to t$ is labeled by $\max((\ell(s), \bot), (\ell(t), \top))$. In particular, the peak $t \mathrel{{}_{[\kappa]}\!\leftarrow} s \mathrel{\to_{[\nu]}} u$ with $\mu = \ell(s)$ is labeled by $A = \max((\mu, \bot), (\kappa, \top))$ to the left and $B = \max((\mu, \bot), (\nu, \top))$ to the right. We claim that using this labeling, the point-decreasing diagrams of Definition 3.38 become decreasing diagrams. Consider a step $v \to w$ of the valley, with label $C = \max((\ell(v), \bot), (\ell(w), \top))$. We consider three cases.

1. Let $v \to w$ be from the $t \to^*_{[\vee\kappa\mu]} \cdot$ subderivation of the valley. The source v of such a step satisfies $\kappa \geqslant \ell(v)$ (hence $(\kappa, \top) > (\ell(v), \bot)$) or $\mu > \ell(v)$ (hence $(\mu, \bot) > (\ell(v), \bot)$), while the target w satisfies $\kappa > \ell(w)$ (hence $(\kappa, \top) > (\ell(w), \top)$)) or $\mu > \ell(w)$ (hence $(\mu, \bot) > (\ell(w), \top)$). Therefore, $A > C$.

2. Assume that $v \to w$ corresponds to the $\cdot \to^=_{[\nu]} \cdot$ step of the valley. Then $\kappa \geqslant \ell(v)$ or $\mu > \ell(v)$, and $\ell(w) = \nu$. We have $(\nu, \top) = (\ell(w), \top)$, $(\kappa, \top) > (\ell(v), \bot)$ and $(\mu, \bot) > (\ell(v), \bot)$. Consequently, $B = C$ or $A > C$.

3. Let $v \to w$ be from the $\cdot \to^*_{[\vee\kappa\mu\nu]} \cdot$ part of the valley. Then $\kappa \geqslant \ell(v)$, $\mu \geqslant \ell(v)$ or $\nu > \ell(v)$, and $\kappa > \ell(w)$, $\mu > \ell(w)$ or $\nu > \ell(w)$. Consequently, $(\kappa, \top) > (\ell(v), \bot)$, $(\mu, \top) > (\ell(v), \bot)$ or $(\nu, \bot) > (\ell(v))$, and $(\kappa, \top) > (\ell(w), \top)$, $(\mu, \top) > (\ell(w), \top)$ or $(\nu, \bot) > (\ell(w), \top)$. Consequently, $A > C$ or $B > C$ follows.

A symmetrical argument applies to steps $w \leftarrow v$ on the left side of the valley. Therefore it follows that

$$t \xrightarrow[\vee A]{*} \cdot \xrightarrow[B]{=} \cdot \xrightarrow[\vee AB]{*} \cdot \xleftarrow[\vee AB]{*} \cdot \xleftarrow[A]{=} \cdot \xleftarrow[\vee B]{*} u$$

This is a decreasing diagram. Since we started from an arbitrary peak and because the order on the set of labels $L \times \{\bot, \top\}$ is well-founded, we conclude that the ARS $\to$ is decreasing by Theorem 3.21. □

Remark 3.40. Definition 3.38 differs from Bognar's [10], which defines decreasing diagrams for peaks and valleys of arbitrary size, based on van Oostrom's *lexicographic path measure* [59]. Furthermore [10] assumes that $>$ is a total, well-founded order. Using the notation of [10], a locally decreasing diagram

$$j \xleftarrow[[j]]{} i \xrightarrow[[k]]{} k \qquad\qquad j \xrightarrow[[\tau']]{*} l \xleftarrow[[\sigma']]{*} k$$

(where σ' and τ' are strings of labels) satisfies

$$|i; j; \tau'| \preceq_\# [i] \cup_{\#\ i\leqslant} |j| \cup_{\#\ i\leqslant} |k| \tag{DCR1}$$

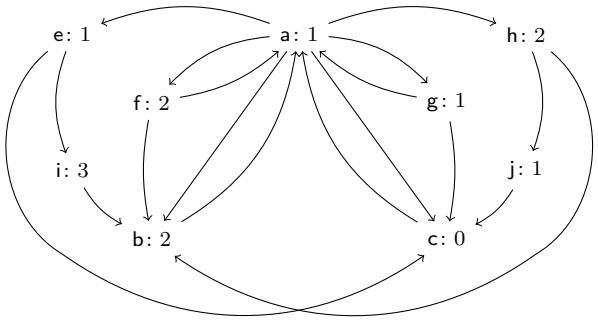


Figure 3.8: Labeling the "Maja the Bee"-example from [11].

and the symmetric property (DCR2) which is obtained by swapping the roles of j, σ' and k, τ'. The condition (DCR1) can be simplified as follows.

$$[i] \cup_{\#\ i\leqslant} |j| \cup_{\#\ \max(i,j)\leqslant} |\tau'| \preceq_{\#} [i] \cup_{\#\ i\leqslant} |j| \cup_{\#\ i\leqslant} |k|$$
$$\max(i,j)\leqslant |\tau'| \preceq_{\#\ i\leqslant} |k|$$

The right-hand side is an empty multiset if $i > k$, in which case τ' must consist of labels all smaller than $\max(i, j)$ (including labels smaller than or equal to k). If $k \geqslant i$, then the right-hand side is the singleton multiset $[k]$, and τ' must consist of some labels smaller than $\max(i, j)$, optionally followed by k, followed by further labels smaller than $\max(i, j, k)$. Definition 3.38 arises from these observations and the fact that a comparison by $\max(i, j)$ (or $\max(i, j, k)$) can be performed by comparing to each of i, j (or i, j, k) and taking the disjunction of the comparison results. It is therefore a generalization of Bognar's definition of local decreasingness.

To conclude the section, we briefly consider the question of completeness of the point version of decreasing diagrams.

Theorem 3.41. *The point version of decreasing diagrams is complete for confluence of finite ARSs.*

Proof. Let $\langle A, \to\rangle$ be a confluent, finite ARS. The relation $\leftrightarrow^*$ is an equivalence relation that partitions A into equivalence classes, the *components* of A. Because A is finite, there are only finitely many components of A and each component contains finitely many objects from A. Let $C \subseteq A$ be a component of A. For all $s, t \in C$, we have $s \leftrightarrow^* t$. Therefore, by finiteness of C and confluence, we can choose an object $f_C \in A$ that is reachable from all elements of C. Furthermore, because $\to^* \subseteq \leftrightarrow^*$, $f_C \in C$. Let

$$F = \{f_C \mid C \text{ is a component of } A\}$$

By this construction, every object $s \in A$ reaches exactly one element of F, namely f_{C_s}, where C_s denotes the component which contains s. Let

$$\ell(a) = \min\{n \in \mathbb{N} \mid a \xrightarrow{n} f_{C_a}\}$$

Note that for any $s \in A$ with $n = \ell(s)$, we have

$$s \xrightarrow[[\vee n]]{n} f_{C_s}$$

We claim that the labeling function ℓ makes $\langle A, \to \rangle$ point-decreasing. To see why, it suffices to consider a local peak $t \,{}_{[n]}\!\leftarrow s \to_{[m]} u$, and note that

$$t \xrightarrow[[\vee n]]{*} f_{C_t} = f_{C_u} \xleftarrow[[\vee m]]{*} u$$

□

Example 3.42. We consider the "Maja the Bee" example by Bognar et al. [11], which has been presented as a counterexample to the completeness of the point-version of decreasing diagrams. The example is reproduced in Figure 3.8. There is only a single component $C = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{e}, \mathsf{f}, \mathsf{g}, \mathsf{h}\}$, and we pick $f_C = \mathsf{c}$, which is reachable from all objects in C. (In fact, C is strongly connected, and we could pick any element of C.) Consider the local peak $\mathsf{e} \,{}_{[1]}\!\leftarrow \mathsf{a} \to_{[2]} \mathsf{f}$. We obtain the joining valley $\mathsf{e} \to_{[0]} \mathsf{c} \,{}_{[0]}\!\leftarrow \mathsf{a} \,{}_{[1]}\!\leftarrow \mathsf{f}$, which passes through a.

This particular peak is of interest because in [11], it is argued that any conversion between e and f that passes through a cannot result in a point-decreasing diagram. Evidently, that is not the case with our labeling, due to the fact that $\ell(\mathsf{f}) > \ell(\mathsf{a})$.

The question whether the point version of decreasing diagrams is complete for countable ARSs remains open.

3.6 Point-Step Decreasing Diagrams

In this section we present a unified result that encompasses both step- and point-decreasing diagrams results, and is strictly more general than both. The key idea is to use a different representation of conversions as Greek (rather than French) strings, a representation that alternates between steps and objects, mapping objects to macron letters and steps to accented letters.

Definition 3.43. Let $\langle \mathcal{A}, (\to_\kappa)_{\kappa \in L} \rangle$ be a labeled ARS. Furthermore let $\ell : \mathcal{A} \to L$ be a function labeling the objects. Then each conversion

$$s_0 \xleftrightarrow[\kappa_1]{} \cdots \xleftrightarrow[\kappa_n]{} s_n$$

has a *point-step interpretation* $\hat{\kappa}_1 \bar{\ell}(s_1) \dots \bar{\ell}(s_{n-1}) \hat{\kappa}_n$, where

$$\hat{\kappa}_i = \begin{cases} \acute{\kappa}_i & \text{if } s_{i-1} \,{}_{\kappa_i}\!\leftarrow s_i \\ \grave{\kappa}_i & \text{if } s_{i-1} \to_{\kappa_i} s_i \end{cases}$$

Note that the initial and final objects are omitted from the interpretation.

S	$\hat{\kappa} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}$
D	$\acute{\kappa}\grave{\mu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}[\grave{\mu}]\{\kappa\mu>\}[\acute{\kappa}]\{\mu>\}$
M1	$\acute{\kappa}\bar{\mu} \mathrel{\overline{\gg}_\bullet} \{\kappa\mu>\}[\acute{\kappa}]\{\mu>\}$
M2	$\acute{\kappa}\bar{\mu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}\bar{\mu}\{\kappa>\}[\acute{\kappa}](\{\kappa>\} \cap \{\mu>\})$
F1	$\bar{\kappa}\bar{\mu} \mathrel{\overline{\gg}_\bullet} \{\kappa\mu>\}$
F2	$\bar{\kappa}\bar{\mu} \mathrel{\overline{\gg}_\bullet} \{\mu>\}\bar{\kappa}\{\mu>\}$
P1	$\acute{\kappa}\bar{\mu}\grave{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa\mu>\}[\acute{\kappa}]\{\mu>\}[\grave{\nu}]\{\mu\nu>\}$
P2	$\acute{\kappa}\bar{\mu}\grave{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa\mu>\}[\grave{\nu}]\{\kappa\mu\nu>\}[\acute{\kappa}]\{\mu\nu>\}$
P3	$\acute{\kappa}\bar{\mu}\grave{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}\bar{\mu}\{\kappa>\}[\acute{\kappa}](\{\kappa>\} \cap \{\mu>\})[\grave{\nu}]\{\nu>\}$
P4	$\acute{\kappa}\bar{\mu}\grave{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}\bar{\mu}\{\kappa>\}[\grave{\nu}]\{\kappa\nu>\}[\acute{\kappa}]\{\nu>\}$
P5	$\acute{\kappa}\bar{\mu}\grave{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}[\grave{\nu}]\{\kappa\nu>\}\bar{\mu}\{\kappa\nu>\}[\acute{\kappa}]\{\nu>\}$
C1	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa\mu\nu>\}[\acute{\kappa}]\{\mu\nu>\}$
C2	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa\nu>\}[\acute{\kappa}]\{\nu>\}\bar{\mu}\{\nu>\}$
C2′	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa\mu>\}[\acute{\kappa}]\{\mu>\}\bar{\nu}\{\mu>\}$
C3	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa\nu>\}\bar{\mu}\{\kappa\nu>\}[\acute{\kappa}](\{\kappa\nu>\} \cap \{\mu\nu>\})$
C3′	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa\mu>\}\bar{\nu}\{\kappa\mu>\}[\acute{\kappa}](\{\kappa\mu>\} \cap \{\mu\nu>\})$
C4	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}\bar{\mu}\{\kappa>\}\bar{\nu}\{\kappa>\}[\acute{\kappa}](\{\kappa>\} \cap \{\mu\nu>\})$
C4′	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}\bar{\nu}\{\kappa>\}\bar{\mu}\{\kappa>\}[\acute{\kappa}](\{\kappa>\} \cap \{\mu\nu>\})$
C5	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}\bar{\mu}\{\kappa>\}[\acute{\kappa}](\{\kappa>\} \cap \{\mu>\})\bar{\nu}(\{\kappa>\} \cap \{\mu>\})$
C5′	$\acute{\kappa}\bar{\mu}\bar{\nu} \mathrel{\overline{\gg}_\bullet} \{\kappa>\}\bar{\nu}\{\kappa>\}[\acute{\kappa}](\{\kappa>\} \cap \{\nu>\})\bar{\mu}(\{\kappa>\} \cap \{\nu>\})$

Table 3.1: Comparing short Greek strings.

In order to obtain a point-step decreasing diagrams result, we map each conversion to its point-step interpretation, and then compare the resulting interpretations before and after pasting a local diagram by $\mathrel{\overline{\gg}_\bullet}$. Obviously, pasting a local diagram corresponds to replacing a substring $\acute{\kappa}\bar{\mu}\grave{\nu}$ (i.e., the interpretation of a local peak without its endpoints) by some other Greek string corresponding to the joining conversion (again, without endpoints). We omit the endpoints because their labels do not change.

Lemma 3.44. *Recall the notation from Theorem 3.12 and Lemma 3.28. All comparisons from Table 3.1 are true.*

Proof. Properties S, D, M1 and M2 have been established earlier. The proof of the remaining properties follow the same general process as seen before in the proofs of Theorem 3.12 and Lemma 3.28: Consider the elements of $\langle\cdot\rangle^g$ applied to the right-hand side of each comparison. In each case, for most of the elements of the resulting multiset, the first component is already strictly smaller than the first component of some element of $\langle\cdot\rangle^g$ applied to the left-hand side. For the remaining elements, we can establish the required comparisons by using the established comparisons on shorter strings.

We will only establish P3 here, leaving the rest as an exercise to the interested reader. We have

$$\langle\acute{\kappa}\bar{\mu}\grave{\nu}\rangle^g = [(\acute{\kappa}, \bar{\mu}\bar{\nu}), (\bar{\mu}, \epsilon), (\grave{\nu}, \acute{\kappa}\bar{\mu})]$$

Let $p \in \{\kappa>\}\bar{\mu}\{\kappa>\}[\acute{\kappa}](\{\kappa>\} \cap \{\mu>\})[\grave{\nu}]\{\nu>\}$. Almost all elements of $\langle p\rangle^g$ have a first component below κ or ν, and are therefore dominated by $(\acute{\kappa}, \bar{\mu}\bar{\nu})$ or

$(\grave{\nu}, \acute{\kappa}\bar{\mu})$, respectively. The exceptions are the elements corresponding to $\bar{\mu}$, $\acute{\kappa}$ and $\grave{\nu}$, if present. These elements are $(\bar{\mu}, \epsilon)$, which is present in $\langle \acute{\kappa}\bar{\mu}\grave{\nu}\rangle^g$, $(\acute{\kappa}, q)$ with $q \in (\{\kappa{>}\} \cap \{\mu{>}\})[\grave{\nu}]\{\nu{>}\}$, for which we have $\bar{\mu}\grave{\nu} \mathrel{\overline{\gg}_\bullet} q$ by the mirrored version of M1, and $(\grave{\nu}, q)$ with $q \in \{\kappa{>}\}\bar{\mu}\{\kappa{>}\}[\acute{\kappa}](\{\kappa{>}\} \cap \{\mu{>}\})$, whence $\acute{\kappa}\bar{\mu} \mathrel{\overline{\gg}_\bullet} q$ by M2. This establishes $\acute{\kappa}\bar{\mu}\grave{\nu} \mathrel{\overline{\gg}_\bullet} p$. □

The properties P1 to P5 from Table 3.1 translate into locally decreasing diagrams for local peaks, and properties C1 to C5′ correspond to locally decreasing diagrams for local cliffs.

Example 3.45. Let (M) denote an object whose label is in M, and $\leftrightarrow^*_M$ stand for a conversion with all steps and intermediate objects (not including the initial and final objects of the conversion) having labels in M. Property P1 corresponds to the following conversion joining a local peak $t \,{}_\kappa\!\leftarrow s \rightarrow_\nu u$, where $\mu = \ell(s)$:

$$t \underset{\vee\kappa\mu}{\overset{*}{\longleftrightarrow}} (\vee\kappa\mu) \underset{\nu}{\overset{=}{\rightarrow}} (\vee\kappa\mu\nu) \underset{\vee\kappa\mu\nu}{\overset{*}{\longleftrightarrow}} (\vee\kappa\mu\nu) \underset{\kappa}{\overset{=}{\leftarrow}} (\vee\mu\nu) \underset{\vee\mu\nu}{\longleftrightarrow} u$$

In the case of rewriting modulo, many properties give rise to several locally decreasing diagrams because macron letters can be labels of equational steps or objects. For example, for the cliff $t \,{}_\kappa\!\leftarrow s \mapsto_\nu u$ with $\mu = \ell(s)$, C2 corresponds to the following joining sequences, where we let $\Leftrightarrow_\kappa = \leftrightarrow_\kappa \cup \mapsto_\kappa$:

$$t \underset{\vee\kappa\nu}{\overset{*}{\Longleftrightarrow}} (\vee\kappa\nu) \underset{\kappa}{\overset{=}{\leftarrow}} (\vee\nu) \underset{\vee\nu}{\overset{*}{\Longleftrightarrow}} (\vee\nu) \underset{\mu}{\overset{}{\longmapsto}} (\vee\nu) \underset{\vee\nu}{\overset{*}{\Longleftrightarrow}} u$$

$$t \underset{\vee\kappa\nu}{\overset{*}{\Longleftrightarrow}} (\vee\kappa\nu) \underset{\kappa}{\overset{=}{\leftarrow}} (\vee\nu) \underset{\vee\nu}{\overset{*}{\Longleftrightarrow}} (\mu) \underset{\vee\nu}{\overset{*}{\Longleftrightarrow}} u \qquad t \underset{\vee\kappa\nu}{\overset{*}{\Longleftrightarrow}} (\vee\kappa\nu) \underset{\kappa}{\overset{=}{\leftarrow}} (\vee\nu) \underset{\vee\nu}{\overset{*}{\Longleftrightarrow}} u$$

$$t \underset{\vee\kappa\nu}{\overset{*}{\Longleftrightarrow}} (\vee\kappa\nu) \underset{\kappa}{\overset{=}{\leftarrow}} (\mu) \underset{\vee\nu}{\overset{*}{\Longleftrightarrow}} u \qquad t \underset{\vee\kappa\nu}{\overset{*}{\Longleftrightarrow}} (\vee\kappa\nu) \underset{\kappa}{\overset{=}{\leftarrow}} u$$

Theorem 3.46. *Let $\langle \mathcal{A}, (\rightarrow_\kappa)_{\kappa\in L}\rangle$ be a labeled ARS, and $\ell : \mathcal{A} \to L$ be a labeling function. Assume that every local peak $t \,{}_\kappa\!\leftarrow s \rightarrow_\nu u$, with $\mu = \ell(s)$ has a joining conversion $s \leftrightarrow^* t$ whose interpretation matches a right-hand side of P1, P2, P3, P4 or P5 from Table 3.1. Then $\rightarrow$ is confluent.*

Proof. We closely follow the proof of Theorem 3.12. By Lemma 3.16 and Theorem 3.19, $\gg_\bullet$ is well-founded and monotone. We show that every conversion $t \leftrightarrow^* u$ has an equivalent valley proof $t \rightarrow^* \cdot \,{}^*\!\leftarrow u$ by well-founded induction on $t \leftrightarrow^* u$, measured by the interpretation of the conversion according to Definition 3.43 and ordered according to $\gg_\bullet$. If $t \leftrightarrow^* u$ is a valley proof then we are done. Otherwise, there must be a local peak, say

$$t \overset{*}{\leftrightarrow} t' \underset{\kappa}{\leftarrow} s' \underset{\nu}{\rightarrow} u' \overset{*}{\leftrightarrow} u \tag{P}$$

Let $\mu = \ell(s')$. Then the interpretation of (P) can be written as $p\acute{\kappa}\bar{\mu}\grave{\nu}r$, where p is the interpretation of $t \leftrightarrow^* t'$ followed by $\bar{\ell}(t')$ and r is $\bar{\ell}(u')$ followed by the interpretation of $u' \leftrightarrow^* u$. By assumption, the local peak $t \,{}_\kappa\!\leftarrow s \rightarrow_\nu u$ has a joining conversion $t \leftrightarrow^* u$ whose interpretation q satisfies $\acute{\kappa}\bar{\mu}\grave{\nu} \gg_\bullet q$ by Lemma 3.44. By monotonicity, this implies $p\acute{\kappa}\bar{\mu}\grave{\nu}r \gg_\bullet pqr$. Consequently, we can apply the induction hypothesis to the resulting conversion $t \leftrightarrow^* t' \leftrightarrow^* u' \leftrightarrow^* u$, whose interpretation is pqr, to conclude. □

Remark 3.47. Theorem 3.46 entails both decreasing diagrams and point-decreasing diagrams. To obtain decreasing diagrams, simply label all objects by a fresh label $\bot$ that is minimal with respect to $>$. Then case D of Table 3.44 (which encodes a decreasing diagram) corresponds to case P2, noting that the sets $\{\kappa\mu{>}\}$, $\{\kappa\mu\nu{>}\}$ and $\{\mu\nu{>}\}$ contain $\bar{\bot}$.

In order to obtain point-decreasing diagrams, let ℓ_p be the labeling function for establishing point-decreasingness. We label steps $s \to t$ by $(\ell_p(t), \top)$ and objects s by $(\ell_p(s), \bot)$, with the tuples ordered lexicographically by $>$ on the original L and $\top > \bot$. Now if we consider the joining conversion from Definition 3.38, we easily see that it corresponds to case P2 of Table 3.44.

Analogously we obtain a theorem for Church-Rosser modulo:

Theorem 3.48. *Let $\langle \mathcal{A}, (\to_\kappa)_{\kappa\in L}\rangle$ be a labeled ARS, $\langle \mathcal{A}, (\mapsto_\kappa)_{\kappa\in L}\rangle$ a labeled ARS whose rewrite relations are symmetric, and let $\ell : \mathcal{A} \to L$ be a labeling function. Let $\Leftrightarrow_\kappa = \leftrightarrow_\kappa \cup \mapsto_\kappa$. If every local peak $t \,{}_\kappa\!\leftarrow s \to_\nu u$ with $\mu = \ell(s)$ has a joining conversion $s \Leftrightarrow^* t$ whose interpretation matches a right-hand side of P1–P5 from Table 3.1, and every local cliff $t \,{}_\kappa\!\leftarrow s \mapsto_\nu u$ with $\mu = \ell(s)$ has a joining conversion whose interpretation matches a right-hand side of C1–C5′ in Table 3.1, then $\to$ is Church-Rosser modulo $\mapsto$.*

Corollary 3.49. *Let $\langle \mathcal{A}, \to\rangle$ be an ARS, and $\ell : \mathcal{A} \to L$ a labeling function. If for ever peak $t \leftarrow s \to u$, either*

$$t \to v \leftarrow u \qquad \text{or} \qquad t \leftrightarrow v_1 \cdots v_n \leftrightarrow u$$

such that $\ell(s) = \ell(v)$ or $\ell(s) > \ell(v_i)$ for $1 \leqslant i \leqslant n$, then $\to$ is confluent.

Proof. Let $L_\bot = L \cup \{\bot\}$, where $\bot$ is a fresh label. We extend $>$ to $L_\bot$ by letting $\alpha > \bot$ for all $\alpha \in L$. Let $(\to_\alpha)_{\alpha\in L_\bot}$ be the labeled ARS given by $\to_\bot = \to$ and $\to_\alpha = \varnothing$. Then since both

$$\acute{\bot}\bar{\ell}(s)\grave{\bot} \gg_\bullet \grave{\bot}\bar{\ell}(v)\acute{\bot}$$

by P5 from Table 3.1 and

$$\acute{\bot}\bar{\ell}(s)\grave{\bot} \gg_\bullet \hat{\bot}\bar{\ell}(v_1)\ldots\bar{\ell}(v_n)\hat{\bot}$$

by P2, we conclude that the local diagrams are point-step decreasing and therefore $\to$ is confluent. □

Remark 3.50. Corollary 3.49 is interesting because to our knowledge, neither decreasing diagrams nor point-decreasing diagrams can prove it directly, i.e., without changing the joining conversions for the local peaks, therefore indicating that point-step decreasing diagrams strictly generalize decreasing diagrams and point-decreasing diagrams.

For the point-decreasing diagrams, the main obstacle presents itself as follows: In order to obtain a point-decreasing diagram (with labeling function ℓ') for joining $t \leftarrow s \to u$ as $t \leftrightarrow v_1 \ldots v_n \leftrightarrow u$ for $n > 1$, since we cannot assume anything about the labels of t and u, we will have to ensure that $\ell'(s) > \ell'(v_i)$

whenever $\ell(s) > \ell(v_i)$. This suggests using $\ell' = \ell$. But then the case of joining $t \to v \leftarrow u$ with $\ell(s) = \ell(v)$ does not result in a point-decreasing diagram if neither $\ell(t) > \ell(s)$ nor $\ell(u) > \ell(s)$ hold. So the proof attempt fails.

For decreasing diagrams, the picture is less clear. Let us assume that the order $>$ on labels is total. The simplest labeling that makes the joining conversions $t \leftrightarrow v_1 \dots v_n \leftrightarrow u$ decreasing labels each step $s \to t$ by $\max(\ell(s), \ell(t))$. Then for the $t \to v \leftarrow u$ join, we have to join the peak $t \,{}_\kappa\!\leftarrow s \to_\mu u$ (with $\kappa = \max(\ell(s), \ell(t))$ and $\mu = \max(\ell(s), \ell(u))$) by $t \to_\kappa s \,{}_\mu\!\leftarrow u$. However, this is only a decreasing diagram if $\kappa = \mu$, and we cannot ensure this in general.

3.7 Commutation and Extended Decreasingness

This section is devoted to two common generalizations of decreasing diagrams. Neither extension is new. The extension to commutation appears in [60, Theorem 2.3.5], whereas extended decreasingness is introduced in [31].

The first extension concerns commutation of two rewrite relations. In order to show commutation of two ARSs $\rightharpoonup$ and $\rightharpoondown$, we adopt the convention that $\rightharpoonup$ steps are always leftward steps. That is, we consider conversions of the form $\overset{*}{\leftrightharpoons}$, where $\leftrightharpoons = \leftharpoonup \cup \rightharpoondown$. These conversions possess a partial monoidal structure much like ordinary conversions, but without the involution. If we label all rewrite steps, then any conversion has a corresponding French string, and we can compare those French strings using the order from Section 3.3.2. We arrive at the following result.

Theorem 3.51. *Let $(\rightharpoonup_\kappa)_{\kappa\in L}$ and $(\rightharpoondown_\kappa)_{\kappa\in L}$ be labeled ARSs. Then $\rightharpoonup$ and $\rightharpoondown$ commute if for all $\kappa, \mu \in L$,*

$$\underset{\kappa}{\leftharpoonup} \cdot \underset{\mu}{\rightharpoondown} \subseteq \underset{\vee\kappa}{\overset{*}{\leftrightharpoons}} \cdot \underset{\mu}{\overset{=}{\rightharpoondown}} \cdot \underset{\vee\kappa\mu}{\overset{*}{\leftrightharpoons}} \cdot \underset{\kappa}{\overset{=}{\leftharpoonup}} \cdot \underset{\vee\mu}{\overset{*}{\leftrightharpoons}}$$

Corollary 3.52. *Let $\to$ be strongly confluent, i.e., $\leftarrow \cdot \to \subseteq \to^* \cdot {}^=\!\leftarrow$. Then $\to_L$ is confluent.*

Proof. We show that $\to$ commutes with itself. To that end, let $L = \{1, 2\}$ with $2 > 1$, $\rightharpoonup_2 = \rightharpoondown_1 = \to$ and $\rightharpoonup_1 = \rightharpoondown_2 = \varnothing$, thereby labeling leftward steps by 1 and rightward steps by 2. We immediately obtain the locally decreasing diagram

$$\underset{2}{\leftharpoonup} \cdot \underset{1}{\rightharpoondown} \subseteq \underset{1}{\overset{*}{\rightharpoondown}} \cdot \underset{2}{\overset{=}{\leftharpoonup}}$$

and no other local peaks ${}_\kappa\!\leftharpoonup \cdot \rightharpoondown_\mu$ exist. Therefore, $\rightharpoonup$ and $\rightharpoondown$ commute by Theorem 3.51. This concludes the proof because $\rightharpoonup = \rightharpoondown = \to$. □

Corollary 3.52 illustrates the use of the commutation version of decreasing diagrams. In fact, it appears that the fact that strong confluence implies confluence cannot be shown by ordinary decreasing diagrams that do not distinguish between left and right steps.

Our second extension concerns the order on labels: rather than requiring a single order $>$ on labels, we consider a pair $(>, \geqslant)$ of a well-founded order $>$ and a quasi-order $\geqslant$ on L that are compatible: $\geqslant \cdot > \cdot \geqslant \subseteq >$. In other words,

$(>, \geqslant)$ is a reduction pair on L, where we treat all elements of L as constants, which makes monotonicity and stability trivial.

Definition 3.53. Let L be a set of labels equipped with a reduction pair $(>, \geqslant)$. A labeled ARS $(\to_\kappa)_{\kappa\in L}$ is *extended locally decreasing* if for all $\kappa, \mu \in L$,

$$\underset{\kappa}{\leftarrow} \cdot \underset{\mu}{\rightarrow} \subseteq \underset{\vee\kappa}{\overset{*}{\longleftrightarrow}} \cdot \underset{\vee\!\!/\mu}{\overset{=}{\longrightarrow}} \cdot \underset{\vee\kappa\mu}{\overset{*}{\longleftrightarrow}} \cdot \underset{\vee\!\!/\kappa}{\overset{=}{\longleftarrow}} \cdot \underset{\vee\mu}{\overset{*}{\longleftrightarrow}}$$

Theorem 3.54. *Let $(\to_\kappa)_{\kappa\in L}$ be an extended locally decreasing labeled ARS. Then $\to_L$ is confluent.*

Proof. We define a new labeled ARS $(\Rightarrow_\kappa)_{\kappa\in L}$ by $\Rightarrow_\kappa = \to_{\vee\!\!/\kappa}$. Consider a local peak $t \ {}_\kappa\!\Leftarrow s \Rightarrow_\mu u$. Then there exist labels $\kappa' \leqslant \kappa$ and $\mu' \leqslant \mu$ such that $t \ {}_{\kappa'}\!\leftarrow s \to_{\mu'} u$. By assumption, there is a joining conversion

$$t \underset{\vee\kappa'}{\overset{*}{\longleftrightarrow}} \cdot \underset{\vee\!\!/\mu'}{\overset{=}{\longrightarrow}} \cdot \underset{\vee\kappa'\mu'}{\overset{*}{\longleftrightarrow}} \cdot \underset{\vee\!\!/\kappa'}{\overset{=}{\longleftarrow}} \cdot \underset{\vee\mu'}{\overset{*}{\longleftrightarrow}} u$$

Note that each label κ'' from $\vee\kappa'$ satisfies $\kappa \geqslant \kappa' > \kappa''$, hence $\kappa > \kappa''$ by compatibility. Furthermore, each label κ'' from $\vee\!\!/\kappa'$ satisfies $\kappa \geqslant \kappa' \geqslant \kappa''$, hence $\kappa \geqslant \kappa''$ by transitivity. By these observations, which analogously hold for labels from $\vee\kappa'\mu'$, $\vee\mu'$ and $\vee\!\!/\mu'$, we find that

$$t \underset{\vee\kappa}{\overset{*}{\Longleftrightarrow}} \cdot \underset{\mu}{\overset{=}{\Longrightarrow}} \cdot \underset{\vee\kappa\mu}{\overset{*}{\Longleftrightarrow}} \cdot \underset{\kappa}{\overset{=}{\Longleftarrow}} \cdot \underset{\vee\mu}{\overset{*}{\Longleftrightarrow}} u$$

Consequently, $(\Rightarrow_\kappa)_{\kappa\in L}$ is locally decreasing, and hence $\Rightarrow_L$ is confluent by Theorem 3.21. This concludes the proof, because $\Rightarrow_L = \to_L$. □

As can be seen from the proof of Theorem 3.54, any confluence proof using extended local decreasingness can be rephrased in terms of local decreasingness. The main benefit of extended local decreasingness is that it allows labeling rewrite steps by functions instead of allowing a step to have several labels. We will exploit this fact extensively in Chapter 4.

3.8 Conclusion

We have presented a well-founded monotone order on French strings that entails the decreasing diagrams technique. Generalizing the monotone order to work on Greek strings that include self-inverse letters, we have obtained a new result for Church–Rosser modulo. Furthermore, we have shown that no complete criterion for Church–Rosser modulo can be obtained by considering proof transformations alone; at least some sort of strategy for applying proof rewrite rules must be incorporated to obtain completeness. Using the same extension to Greek strings, we have also demonstrated an extension to point-step decreasing diagrams that labels both steps and objects of an ARS, and generalizes both point-decreasingness and decreasing diagrams. Finally, we have presented commonly used extensions to the decreasing diagrams technique to commutation, and to reduction pairs.

Chapter 4

Labeling Diagrams Decreasingly

In this chapter we develop a framework for labeling the rewrite steps of a TRS $\mathcal{R}$ such that all local peaks arising from rewriting by $\mathcal{R}$ are decreasing, thereby establishing confluence of $\mathcal{R}$. The main challenge is to obtain a finite criterion that covers all local peaks. To this end, we take inspiration from the critical pair lemma for establishing local confluence (Lemma 2.47). We restrict labeling functions such that local peaks that do not arise from critical overlaps are automatically decreasing, while we demand that decreasing diagrams exist for critical pairs.

The contents of this chapter has appeared previously in [79].

4.1 Introduction

Confluence is an important property of rewrite systems since it ensures unique normal forms. It is decidable in the presence of termination [44] and implied by orthogonality [67] or restricted joinability conditions on the critical pairs [34, 76, 61, 65, 58]. Recently, there is a renewed interest in confluence research, with a strong emphasis on automation. As one application we mention [68], where automated confluence tools are employed for proving soundness of abstract forms of reduction in solving the typing problem.

The decreasing diagrams technique of van Oostrom [59] is a complete method for showing confluence of countable abstract rewrite systems. The main idea of the approach is to show confluence by establishing local confluence under the side condition that rewrite steps of the joining sequences must *decrease* with respect to some well-founded order. For term rewrite systems however, the main problem for automation of decreasing diagrams is that in general infinitely many local peaks must be considered. To reduce this problem to a finite set of local peaks one can label rewrite steps with functions that satisfy special properties. In [62] van Oostrom presented the rule labeling that allows to conclude confluence of *linear* rewrite systems by checking decreasingness of the critical peaks (those emerging from critical overlaps). The rule labeling has been implemented by Aoto [1] and Hirokawa and Middeldorp [32]. Already in [62] van Oostrom presented constraints that allow to apply the rule labeling to *left-linear* systems. This approach has been implemented and extended by Aoto [1]. Our framework subsumes the above ideas.

The contributions of this chapter comprise the extraction of abstract constraints on a labeling such that for a (left-)linear rewrite system decreasingness

of the (parallel) critical peaks ensures confluence. We show that the rule labeling adheres to our constraints and present additional labeling functions. Furthermore such labeling functions can be combined lexicographically to obtain new labeling functions satisfying our constraints. This approach allows the formulation of an abstract criterion that makes virtually every labeling function for linear rewrite systems also applicable to left-linear systems. Consequently, confluence of the TRS in Example 4.1 can be established automatically, e.g., by the rule labeling, while current approaches based on the decreasing diagrams technique [1, 32] as well as other confluence criteria like Knuth and Bendix' criterion or orthogonality (and its refinements) fail.

Example 4.1. Consider the TRS $\mathcal{R}$ (Cops #60)[1] consisting of the rules

$$
\begin{array}{lll}
1\colon & x + (y + z) \to (x + y) + z & 6\colon \quad x \times y \to y \times x \\
2\colon & (x + y) + z \to x + (y + z) & 7\colon \quad \mathsf{s}(x) + y \to x + \mathsf{s}(y) \\
3\colon & \mathsf{sq}(x) \to x \times x & 8\colon \quad x + \mathsf{s}(y) \to \mathsf{s}(x) + y \\
4\colon & \mathsf{sq}(\mathsf{s}(x)) \to (x \times x) + \mathsf{s}(x + x) & 9\colon \quad x \times \mathsf{s}(y) \to x + (x \times y) \\
5\colon & x + y \to y + x & 10\colon \quad \mathsf{s}(x) \times y \to (x \times y) + y
\end{array}
$$

This system is locally confluent since all its 34 critical pairs are joinable.

The remainder of this chapter is organized as follows. We present constraints (on a labeling) such that decreasingness of the critical peaks ensures confluence for (left-)linear rewrite systems in Section 4.2. Three of these constraints are based on relative termination while the fourth employs persistence. We focus on parallel rewriting in Section 4.3. The merits of these approaches are assessed in Section 4.4 by discussing the relationship to the recent literature. Implementation issues are addressed in Section 4.5. Section 4.6 concludes.

4.2 Labeling Plain Rewrite Steps

In this section we present constraints (on a labeling) such that decreasingness of the critical peaks ensures confluence of linear (Section 4.2.1) and left-linear (Section 4.2.2) TRSs. Furthermore, we show that if two labelings satisfy these conditions then also their lexicographic combination satisfies them.

For a local peak

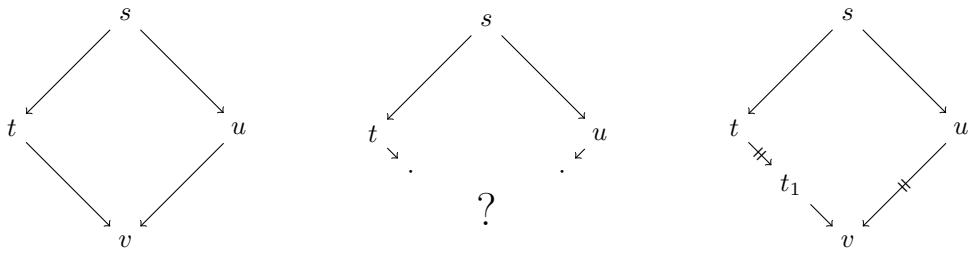
$$t = s[r_1\sigma]_p \leftarrow s[l_1\sigma]_p = s = s[l_2\sigma]_q \to s[r_2\sigma]_q = u \tag{4.1}$$

there are three possibilities (modulo symmetry):

(a) $p \parallel q$ (parallel),

(b) $q \leqslant p$ and $p\backslash q \in \mathcal{P}os_{\mathcal{F}}(l_2)$ (critical overlap),

(c) $q < p$ and $p\backslash q \notin \mathcal{P}os_{\mathcal{F}}(l_2)$ (variable overlap).



(a) (parallel) (b) (critical overlap) (c) (variable overlap)

Figure 4.1: Three kinds of local peaks.

These cases are visualized in Figure 4.1. Figure 4.1(a) shows the shape of a local peak where the steps take place at parallel positions. Here we have $s \to^{p,l_1\to r_1} t$ and $u \to^{p,l_1\to r_1} v$ as well as $s \to^{q,l_2\to r_2} u$ and $t \to^{q,l_2\to r_2} v$, i.e., the steps drawn at opposing sides in the diagram are due to the same rules. The question mark in Figure 4.1(b) conveys that joinability of critical overlaps may depend on auxiliary rules. Variable overlaps (Figure 4.1(c)) can again be joined by the rules involved in the diverging step. More precisely, if q' is the unique position in $\mathcal{P}os_{\mathcal{V}}(l_2)$ such that $qq' \leqslant p$, $x = l_2|_{q'}$, $|l_2|_x = m$, and $|r_2|_x = n$ then we have $t \to^{m-1}_{l_1\to r_1} t_1$, $t_1 \to_{l_2\to r_2} v$, and $u \to^{n}_{l_1\to r_1} v$.

Labelings are used to compare rewrite steps. In the sequel we denote the set of all rewrite steps for a TRS $\mathcal{R}$ by $\Lambda_{\mathcal{R}}$ and elements from this set by capital Greek letters Γ and Δ. Furthermore if $\Gamma = s \to^{p,l\to r} t$ then $C[\Gamma\sigma]$ denotes the rewrite step $C[s\sigma] \to^{p'p,l\to r} C[t\sigma]$ for any substitution σ and context C with $C|_{p'} = \Box$.

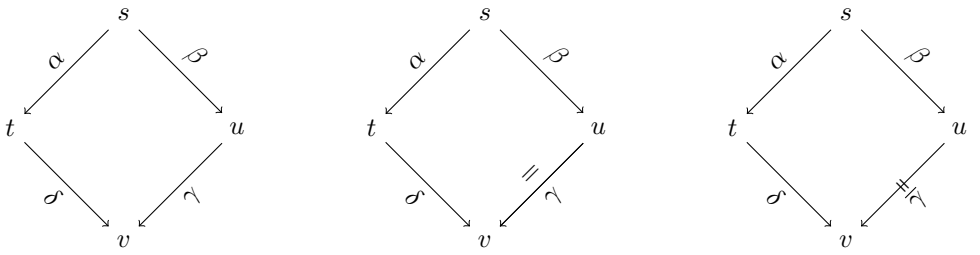
Definition 4.2. Let $\mathcal{R}$ be a TRS. A *labeling function* $\ell\colon \Lambda_{\mathcal{R}} \to W$ is a mapping from rewrite steps into some set W. A *labeling* $(\ell, \geqslant, >)$ for $\mathcal{R}$ consists of a labeling function ℓ, a preorder $\geqslant$, and a well-founded order $>$ such that $\geqslant$ and $>$ are compatible and for all rewrite steps $\Gamma, \Delta \in \Lambda_{\mathcal{R}}$, contexts C and substitutions σ:

1. $\ell(\Gamma) \geqslant \ell(\Delta)$ implies $\ell(C[\Gamma\sigma]) \geqslant \ell(C[\Delta\sigma])$, and
2. $\ell(\Gamma) > \ell(\Delta)$ implies $\ell(C[\Gamma\sigma]) > \ell(C[\Delta\sigma])$.

All labelings we present satisfy $> \subseteq \geqslant$, which allows to avoid tedious case distinctions, and we assume this property henceforth. We do so without loss of generality, because $((> \cup \geqslant)^*, >)$ satisfies the conditions of Definition 4.2 if $(\geqslant, >)$ does.

In the sequel W, $\geqslant$, and $>$ are left implicit when clear from the context and a labeling is identified with the labeling function ℓ. We use the terminology that a labeling ℓ is *monotone* and *stable* if properties 1 and 2 of Definition 4.2 hold. Abstract labels, i.e., labels that are unknown, are represented by lowercase Greek letters α, β, γ, and δ. We write $s \to^{\pi}_{\alpha} t$ (or simply $s \to_{\alpha} t$) if $\ell(s \to^{\pi} t) = \alpha$. Often we leave the labeling ℓ implicit and just attach labels to arrows. A local

[1] COnfluence ProblemS, see `http://coco.nue.riec.tohoku.ac.jp/problems/`.



(a) (parallel) (b) (variable-linear) (c) (variable-left-linear)

Figure 4.2: Labeled local peaks.

peak $t \leftarrow s \rightarrow u$ is called *decreasing for* ℓ if there are labels α and β such that $t\ {}_{\alpha}\!\leftarrow s \rightarrow_{\beta} u$, and $\rightarrow_{\alpha}$ and $\rightarrow_{\beta}$ are decreasing with respect to $\geqslant$ and $>$. To employ Theorem 3.21 for TRSs, decreasingness of the ARS $\langle \mathcal{T}(\mathcal{F},\mathcal{V}), \{\rightarrow_w\}_{w\in W}\rangle$ must be shown.

In this chapter we investigate conditions on a labeling such that local peaks according to (parallel) and (variable overlap) are decreasing automatically. This is desirable since in general there are infinitely many local peaks corresponding to these cases (even if the underlying TRS has finitely many rules). There are also infinitely many local peaks according to (critical overlap) in general, but for a finite TRS they are captured by the finitely many critical overlaps. Still, it is undecidable if they are decreasingly joinable [32].

For later reference, Figure 4.2 shows labeled local peaks for the case (parallel) (Figure 4.2(a)) and (variable overlap) if the rule $l_2 \rightarrow r_2$ in local peak (4.1) is linear (Figure 4.2(b)) and left-linear (Figure 4.2(c)), respectively. In Figure 4.2(c) the expression $\overline{\gamma}$ denotes a sequence of labels $\vec{\gamma}$. In the subsequent analysis we will always use the fact that the local peaks in Figure 4.2 can be closed by the rules involved in the peak (applied at opposing sides in the diagram).

4.2.1 Linear TRSs

The next definition presents sufficient abstract conditions on a labeling such that local peaks according to the cases (parallel) and (variable-linear) in Figure 4.2 are decreasing. We use the observation that for linear TRSs the (parallel) case can be seen as an instance of the (variable-linear) case to shorten proofs.

Definition 4.3. Let ℓ be a labeling for a TRS $\mathcal{R}$. We call ℓ an *L-labeling (for* $\mathcal{R}$*)* if for local peaks according to (parallel) and (variable-linear) we have $\alpha \geqslant \gamma$ and $\beta \geqslant \delta$ in Figures 4.2(a) and 4.2(b), respectively.

The local diagram in Figure 4.3(a) visualizes the conditions on an L-labeling more succinctly. We will use L-labelings also for left-linear TRSs, where no conditions are required for local peaks different from (parallel) and (variable-linear). We call the critical peaks of a TRS $\mathcal{R}$ Φ-*decreasing* if there exists a Φ-labeling ℓ for $\mathcal{R}$ such that the critical peaks of $\mathcal{R}$ are decreasing for ℓ. In the sequel we will introduce further labelings, e.g., LL-labelings and weak

LL-labelings. The placeholder Φ avoids the need for repeating the definition of decreasingness for these labelings.

The next theorem states that L-labelings may be used to show confluence of linear TRSs.

Theorem 4.4. *Let $\mathcal{R}$ be a linear TRS. If the critical peaks of $\mathcal{R}$ are L-decreasing then $\mathcal{R}$ is confluent.*

Proof. By assumption there is an L-labeling ℓ that makes the critical peaks of $\mathcal{R}$ decreasing. We establish confluence of $\mathcal{R}$ by Theorem 3.21, i.e., show decreasingness of the ARS $\langle \mathcal{T}(\mathcal{F},\mathcal{V}), \to_{\mathcal{R}} \rangle$ where rewrite steps are labeled according to ℓ. Since $\mathcal{R}$ is linear, local peaks have the shape (parallel), (variable-linear), or (critical overlap). By definition of an L-labeling the former two are decreasing. Now consider a local peak according to (critical overlap), i.e., for the local peak (4.1) we have $q \leqslant p$ and $p\backslash q \in \mathcal{P}os_{\mathcal{F}}(l_2)$. Let $p' = p\backslash q$. Then $t|_q \leftarrow s|_q \to u|_q$ must be an instance of a critical peak $l_2\mu[r_1\mu]_{p'} \leftarrow l_2[l_1\mu]_{p'} = l_2\mu \to r_2\mu$ which is decreasing by assumption. By monotonicity and stability of ℓ we obtain decreasingness of the local peak (4.1). □

We recall the rule labeling of van Oostrom [62], parametrized by a mapping $i\colon \mathcal{R} \to \mathbb{N}$. Often i is left implicit. The rule labeling satisfies the constraints of an L-labeling.

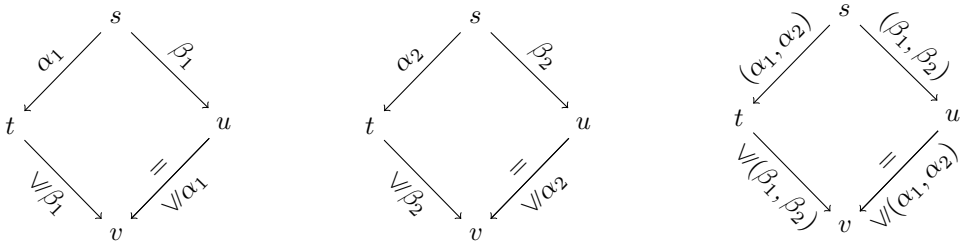
Lemma 4.5. *Let $\mathcal{R}$ be a TRS and $\ell^i_{\mathrm{rl}}(s \to^\pi t) = i(l_\pi \to r_\pi)$. Then $(\ell^i_{\mathrm{rl}}, \geqslant_{\mathbb{N}}, >_{\mathbb{N}})$ is an L-labeling for $\mathcal{R}$.*

Proof. First we show that $(\ell^i_{\mathrm{rl}}, \geqslant_{\mathbb{N}}, >_{\mathbb{N}})$ is a labeling. The preorder $\geqslant_{\mathbb{N}}$ and the well-founded order $>_{\mathbb{N}}$ are compatible. Furthermore $\ell^i_{\mathrm{rl}}(s \to^\pi t) = i(l_\pi \to r_\pi)$ which ensures monotonicity and stability of ℓ^i_{rl}. Hence $(\ell^i_{\mathrm{rl}}, \geqslant_{\mathbb{N}}, >_{\mathbb{N}})$ is a labeling. Next we show the properties demanded in Definition 4.3. For local peaks according to cases (parallel) and (variable-linear) we recall that the steps drawn at opposite sides in the diagram, e.g., the steps labeled with α and γ (β and δ) in Figures 4.2(a) and 4.2(b), are due to applications of the same rule. Hence $\alpha = \gamma$ and $\beta = \delta$ in Figures 4.2(a) and 4.2(b), which shows the result. □

Inspired by [32] we propose a labeling based on relative termination.

Lemma 4.6. *Let $\mathcal{R}$ be a TRS and $\ell_{\mathrm{rt}}(s \to t) = s$. Then $\ell^{\mathcal{S}}_{\mathrm{rt}} = (\ell_{\mathrm{rt}}, \to^*_{\mathcal{R}}, \to^+_{\mathcal{S}/\mathcal{R}})$ is an L-labeling for $\mathcal{R}$, provided $\to_{\mathcal{S}} \subseteq \to_{\mathcal{R}}$ and $\mathcal{S}/\mathcal{R}$ is terminating.*

Proof. Let $\geqslant = \to^*_{\mathcal{R}}$ and $> = \to^+_{\mathcal{S}/\mathcal{R}}$. First we show that $(\ell_{\mathrm{rt}}, \geqslant, >)$ is a labeling. By definition of relative rewriting, $\geqslant$ and $>$ are compatible and $>$ is well-founded by the termination assumption of $\mathcal{S}/\mathcal{R}$. Since rewriting is closed under contexts and substitutions, $\ell^{\mathcal{S}}_{\mathrm{rt}}$ is monotone and stable and hence a labeling. Next we show the properties demanded in Definition 4.3. The assumption $\to_{\mathcal{S}} \subseteq \to_{\mathcal{R}}$ yields $> \subseteq \geqslant$. Combining $\alpha = s = \beta$, $\gamma = u$, and $\delta = t$ with $s \to_{\mathcal{R}} t$ and $s \to_{\mathcal{R}} u$ yields $\alpha = \beta \geqslant \gamma, \delta$ for local peaks according to (parallel) and (variable-linear) in Figures 4.2(a) and 4.2(b). □



(a) Labeling ℓ_1. (b) Labeling ℓ_2. (c) Labeling $\ell_1 \times \ell_2$.

Figure 4.3: Lexicographic combination of L-labelings.

The L-labeling from the previous lemma allows to establish a decrease with respect to some steps of $\mathcal{R}$. The next lemma allows to combine L-labelings. Let $\ell_1 \colon \Lambda_\mathcal{R} \to W_1$ and $\ell_2 \colon \Lambda_\mathcal{R} \to W_2$. Then $(\ell_1, \geqslant_1, >_1) \times (\ell_2, \geqslant_2, >_2)$ is defined as $(\ell_1 \times \ell_2, \geqslant_{12}, >_{12})$ where $\ell_1 \times \ell_2 \colon \Lambda_\mathcal{R} \to W_1 \times W_2$ with $(\ell_1 \times \ell_2)(\Gamma) = (\ell_1(\Gamma), \ell_2(\Gamma))$. Furthermore $(x_1, x_2) \geqslant_{12} (y_1, y_2)$ if and only if $x_1 >_1 y_1$ or $x_1 \geqslant_1 y_1$ and $x_2 \geqslant_2 y_2$ and $(x_1, x_2) >_{12} (y_1, y_2)$ if and only if $x_1 >_1 y_1$ or $x_1 \geqslant_1 y_1$ and $x_2 >_2 y_2$.

Lemma 4.7. *Let ℓ_1 and ℓ_2 be L-labelings. Then $\ell_1 \times \ell_2$ is an L-labeling.*

Proof. First we show that $\ell_1 \times \ell_2$ is monotone and stable whenever ℓ_1 and ℓ_2 are labelings. Indeed if $(\ell_1 \times \ell_2)(\Gamma) \geqslant (\ell_1 \times \ell_2)(\Delta)$ then $\ell_1(\Gamma) > \ell_1(\Delta)$ or $\ell_1(\Gamma) \geqslant \ell_1(\Delta)$ and $\ell_2(\Gamma) \geqslant \ell_2(\Delta)$, which for all contexts C and substitutions σ implies $\ell_1(C[\Gamma\sigma]) > \ell_1(C[\Delta\sigma])$ or $\ell_1(C[\Gamma\sigma]) \geqslant \ell_1(C[\Delta\sigma])$ and $\ell_2(C[\Gamma\sigma]) \geqslant \ell_2(C[\Delta\sigma])$ by stability and monotonicity of ℓ_1 and ℓ_2, which is equivalent to $(\ell_1 \times \ell_2)(C[\Gamma\sigma]) \geqslant (\ell_1 \times \ell_2)(C[\Delta\sigma])$. Showing stability and monotonicity of $>$ is similar. Since the lexicographic product satisfies $>_{12} \subseteq \geqslant_{12}$ if ℓ_1 and ℓ_2 are labelings we conclude that $\ell_1 \times \ell_2$ is a labeling.

Next we show that $\ell_1 \times \ell_2$ satisfies the requirements of Definition 4.3. If ℓ_1 and ℓ_2 are L-labelings then the diagram of Figure 4.2(b) has the shape as in Figure 4.3(a) and 4.3(b), respectively. It is easy to see that the lexicographic combination is again an L-labeling (cf. Figure 4.3(c)). □

4.2.2 Left-linear TRSs

For left-linear TRSs the notion of an LL-labeling is introduced. The following definition exploits that Figure 4.2(b) is an instance of Figure 4.2(c).

Definition 4.8. A labeling ℓ for a TRS $\mathcal{R}$ is an *LL-labeling (for $\mathcal{R}$)* if

1. in Figure 4.2(a), $\alpha \geqslant \gamma$ and $\beta \geqslant \delta$,
2. in Figure 4.2(c), $\alpha \geqslant \overline{\gamma}$ and $\beta \geqslant \delta$ for all permutations of the rewrite steps of $u \mathrel{+\!\!\!\twoheadrightarrow} v$, where $\alpha \geqslant \overline{\gamma}$ means $\alpha \geqslant \gamma_i$ for $1 \leqslant i \leqslant n$, and
3. in Figure 4.2(c), $\alpha > \overline{\gamma}$ for some permutation of the rewrite steps of $u \mathrel{+\!\!\!\twoheadrightarrow} v$, where $\alpha > \overline{\gamma}$ means $\alpha \geqslant \gamma_1$ and $\alpha > \gamma_i$ for $2 \leqslant i \leqslant n$.

A labeling ℓ is a *weak LL-labeling* if the first two conditions are satisfied.

Considering *all* permutations in case 2 of Definition 4.8 is necessary to ensure that the lexicographic combination of two weak LL-labelings again is a weak LL-labeling (cf. Lemma 4.12). Furthermore, this condition facilitates their use for parallel rewriting (Section 4.3).

Remark 4.9. The L-labelings presented so far (cf. Lemmata 4.5 and 4.6) are weak LL-labelings.

The next theorem states that LL-labelings allow to show confluence of left-linear TRSs.

Theorem 4.10. *Let $\mathcal{R}$ be a left-linear TRS. If the critical peaks of $\mathcal{R}$ are LL-decreasing then $\mathcal{R}$ is confluent.*

Proof. By assumption the critical peaks of $\mathcal{R}$ are decreasing for some LL-labeling ℓ. We establish confluence of $\mathcal{R}$ by Theorem 3.21, i.e., we show decreasingness of the ARS $\langle \mathcal{T}(\mathcal{F},\mathcal{V}), \to_{\mathcal{R}} \rangle$ by labeling rewrite steps according to ℓ. By definition of an LL-labeling local peaks according to (parallel) and (variable-left-linear) are decreasing. The reasoning for local peaks according to (critical overlap) is the same as in the proof of Theorem 4.4. □

The rule labeling from Lemma 4.5 is a weak LL-labeling but not an LL-labeling since in Figure 4.2(c) we have $\alpha = \gamma_i$ for $1 \leqslant i \leqslant n$ which does not satisfy $\alpha > \overline{\gamma}$ if $n > 1$. (See also [32, Example 9].) We return to this problem and propose two solutions (based on persistence of confluence and on parallel steps) after presenting simpler (weak) LL-labelings based on measuring duplicating steps, the context above the contracted redex, and the contracted redex.

Measuring Duplicating Steps

The L-labeling from Lemma 4.6 can be adapted to an LL-labeling.

Lemma 4.11. *Let $\mathcal{R}$ be a TRS. Then $\ell_{\mathrm{rt}}^{\mathcal{R}_\mathsf{d}}$ is an LL-labeling, provided $\mathcal{R}_\mathsf{d}/\mathcal{R}_\mathsf{nd}$ is terminating.*

Proof. By Theorem 2.41 the relative TRS $\mathcal{R}_\mathsf{d}/\mathcal{R}_\mathsf{nd}$ is terminating if and only if $\mathcal{R}_\mathsf{d}/\mathcal{R}$ is terminating. Hence $(\ell_{\mathrm{rt}}^{\mathcal{R}_\mathsf{d}}, \geqslant, >)$ is a labeling by Lemma 4.6. Here $\geqslant = \to_{\mathcal{R}}^*$ and $> = \to_{\mathcal{R}_\mathsf{d}/\mathcal{R}}^+$. Since $\ell_{\mathrm{rt}}(s \to t) = s$, we have $\alpha = \beta$ in Figures 4.2(a) and 4.2(c). We have $> \subseteq \geqslant$. Hence $\alpha \geqslant \gamma$ and $\alpha \geqslant \delta$ in Figure 4.2(a) and, if $l_2 \to r_2$ in local peak (4.1) is linear, also in Figure 4.2(c) as $\overline{\gamma}$ is empty or $\overline{\gamma} = \gamma$ in this case. If $l_2 \to r_2$ is not linear then it must be duplicating and hence $\alpha > \gamma_i$ for $1 \leqslant i \leqslant n$. Because $\alpha \geqslant \delta$, $\ell_{\mathrm{rt}}^{\mathcal{R}_\mathsf{d}}$ is an LL-labeling for $\mathcal{R}$. □

To combine the previous lemma with the rule labeling we study how different labelings can be combined.

Lemma 4.12. *Let ℓ_1 be an LL-labeling and let ℓ_2 be a weak LL-labeling. Then $\ell_1 \times \ell_2$ and $\ell_2 \times \ell_1$ are LL-labelings.*

Proof. By the proof of Lemma 4.7 $\ell_1 \times \ell_2$ and $\ell_2 \times \ell_1$ are labelings. The only interesting case of (variable-left-linear) is when $l_2 \to r_2$ in local peak (4.1) is non-linear, i.e., $\overline{\gamma}$ contains more than one element. First we show that $\ell_1 \times \ell_2$ is an LL-labeling. Here labels according to ℓ_1 are suffixed with the subscript 1 and similarly for ℓ_2. Recall Figure 4.2(c). Let us first deal with Definition 4.8(2). We have $\alpha_1 \geqslant \overline{\gamma}_1$, $\beta_1 \geqslant \delta_1$, $\alpha_2 \geqslant \overline{\gamma}_2$ and $\beta_2 \geqslant \delta_2$, which yields $(\beta_1, \beta_2) \geqslant (\delta_1, \delta_2)$, $(\alpha_1, \alpha_2) \geqslant (\gamma_{1i}, \gamma_{2i})$ for all $1 \leqslant i \leqslant n$, by the definition of the lexicographic product. Next we consider Definition 4.8(3). By assumption we have $\alpha_1 > \overline{\gamma}_1$, and $\alpha_2 \geqslant \overline{\gamma}_2$, which yields the desired $(\alpha_1, \alpha_2) \geqslant (\gamma_{11}, \gamma_{21})$, $(\alpha_1, \alpha_2) > (\gamma_{1i}, \gamma_{2i})$ for $2 \leqslant i \leqslant n$. In the proof for $\ell_2 \times \ell_1$ the assumptions yield $(\beta_2, \beta_1) \geqslant (\delta_2, \delta_1)$ and $(\alpha_2, \alpha_1) \geqslant (\gamma_{2i}, \gamma_{1i})$ for $1 \leqslant i \leqslant n$ for Definition 4.8(2) and additionally $(\alpha_2, \alpha_1) > (\gamma_{2i}, \gamma_{1i})$ for $2 \leqslant i \leqslant n$ for Definition 4.8(3). □

Remark 4.13. If ℓ_1 and ℓ_2 are weak LL-labelings then so are $\ell_1 \times \ell_2$ and $\ell_2 \times \ell_1$. Furthermore, LL-labelings are also weak LL-labelings by definition. In particular LL-labelings can be composed lexicographically.

From Theorem 4.10 and Lemmata 4.11 and 4.12 we obtain the following result.

Corollary 4.14. *Let $\mathcal{R}$ be a left-linear TRS. If $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ is terminating and all critical peaks of $\mathcal{R}$ are weakly LL-decreasing then $\mathcal{R}$ is confluent.*

Proof. By Lemma 4.11 $\ell_{\mathrm{rt}}^{\mathcal{R}_{\mathsf{d}}}$ is an LL-labeling. By assumption the critical peaks of $\mathcal{R}$ are decreasing for some weak LL-labeling ℓ. By Lemma 4.12 also $\ell_{\mathrm{rt}}^{\mathcal{R}_{\mathsf{d}}} \times \ell$ is an LL-labeling. It remains to show decreasingness of the critical peaks for $\ell_{\mathrm{rt}}^{\mathcal{R}_{\mathsf{d}}} \times \ell$. This is obvious since for terms s, t, u with $s \to_{\mathcal{R}} t \to_{\mathcal{R}} u$ we have $\ell_{\mathrm{rt}}^{\mathcal{R}_{\mathsf{d}}}(s \to t) \geqslant \ell_{\mathrm{rt}}^{\mathcal{R}_{\mathsf{d}}}(t \to u)$. Hence decreasingness for ℓ implies decreasingness for $\ell_{\mathrm{rt}}^{\mathcal{R}_{\mathsf{d}}} \times \ell$. Confluence of $\mathcal{R}$ follows from Theorem 4.10. □

We revisit the example from the introduction.

Example 4.15. Recall the TRS $\mathcal{R}$ from Example 4.1. The polynomial interpretation

$$+_{\mathbb{N}}(x,y) = x + y \quad \mathsf{s}_{\mathbb{N}}(x) = x + 1 \quad \times_{\mathbb{N}}(x,y) = x^2 + xy + y^2 \quad \mathsf{sq}_{\mathbb{N}}(x) = 3x^2 + 1$$

shows termination of $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$. It is easy to check that ℓ_{rl}^{i} with $i(3) = i(6) = 2$, $i(4) = i(10) = 1$, and $i(l \to r) = 0$ for all other rules $l \to r \in \mathcal{R}$ establishes decreasingness of the 34 critical peaks. We consider two selected critical peaks (where the applied rewrite rule is indicated above the arrow in parentheses). The peaks

$$t_1 = x + ((y + z) + w) \underset{0}{\overset{(1)}{\leftarrow}} x + (y + (z + w)) \underset{0}{\overset{(1)}{\rightarrow}} (x + y) + (z + w) = u_1$$

$$t_2 = \mathsf{s}(x) \times \mathsf{s}(x) \underset{2}{\overset{(3)}{\leftarrow}} \mathsf{sq}(\mathsf{s}(x)) \underset{1}{\overset{(4)}{\rightarrow}} (x \times x) + \mathsf{s}(x + x) = u_2$$

can be joined decreasingly as follows:

$$
\begin{aligned}
t_1 &\xrightarrow[0]{(2)} x + (y + (z + w)) \xleftarrow[0]{(2)} u_1 \\
t_2 &\xrightarrow[1]{(10)} (x \times \mathsf{s}(x)) + \mathsf{s}(x) \xrightarrow[0]{(9)} (x + (x \times x)) + \mathsf{s}(x) \xrightarrow[0]{(2)} x + ((x \times x) + \mathsf{s}(x)) \\
&\xrightarrow[0]{(8)} x + (\mathsf{s}(x \times x) + x) \xleftarrow[0]{(2)} (x + \mathsf{s}(x \times x)) + x \xleftarrow[0]{(5)} (\mathsf{s}(x \times x) + x) + x \\
&\xleftarrow[0]{(1)} \mathsf{s}(x \times x) + (x + x) \xleftarrow[0]{(8)} u_2
\end{aligned}
$$

The next example is concise and constitutes a minimal example to familiarize the reader with Corollary 4.14.

Example 4.16. Consider the TRS $\mathcal{R}$ consisting of the three rules

$$1\colon\ \mathsf{b} \to \mathsf{a} \qquad 2\colon\ \mathsf{a} \to \mathsf{b} \qquad 3\colon\ \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to \mathsf{g}(\mathsf{f}(x), \mathsf{f}(x))$$

We have $\mathcal{R}_{\mathsf{d}} = \{3\}$ and $\mathcal{R}_{\mathsf{nd}} = \{1, 2\}$. Termination of $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ can be established by LPO with precedence $\mathsf{a} \sim \mathsf{b}$ and $\mathsf{f} > \mathsf{g}$. The rule labeling that takes the rule numbers as labels shows the only critical peak decreasing, i.e., $\mathsf{f}(\mathsf{g}(x, \mathsf{b}))\ {}_2\!\leftarrow \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_3 \mathsf{g}(\mathsf{f}(x), \mathsf{f}(x))$ and $\mathsf{f}(\mathsf{g}(x, \mathsf{b})) \to_1 \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_3 \mathsf{g}(\mathsf{f}(x), \mathsf{f}(x))$. Hence we obtain the confluence of $\mathcal{R}$ by Corollary 4.14.

Remark 4.17. Using $\ell^i_{\mathrm{rl}}(\cdot) = 0$ as weak LL-labeling, Corollary 4.14 gives a condition (termination of $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$) such that $t \to^{=} u$ or $u \to^{=} t$ for all critical pairs $t \leftarrow\!\rtimes\!\rightarrow u$ implies confluence of a left-linear TRS $\mathcal{R}$. This partially answers one question in the RTA list of open problems #13.[2]

Measuring the Context above the Contracted Redex

In [62, Example 20] van Oostrom suggests to count function symbols above the contracted redex, demands that this measurement decreases for variables that are duplicated, and combines this with the rule labeling. Consequently local peaks according to Figure 4.2(c) are decreasing. Below we exploit this idea but incorporate the following beneficial generalizations. First, we do not restrict to counting function symbols (which has been adopted and extended by Aoto in [1]) but represent the constraints as a relative termination problem. This abstract formulation allows to strictly subsume the criteria from [62, 1] (see Section 4.4) because more advanced techniques than counting symbols can be applied for proving termination. Additionally, our setting also allows to weaken these constraints significantly (cf. Lemma 4.25).

The next example motivates the need for an LL-labeling that does not require termination of $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$.

Example 4.18. Consider the TRS $\mathcal{R}$ consisting of the six rules

$$
\begin{array}{lll}
\mathsf{f}(\mathsf{h}(x)) \to \mathsf{h}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) & \mathsf{f}(x) \to \mathsf{a} & \mathsf{a} \to \mathsf{b} \\
\mathsf{h}(x) \to \mathsf{c} & \mathsf{b} \to \bot & \mathsf{c} \to \bot
\end{array}
$$

[2] http://www.cs.tau.ac.il/~nachum/rtaloop/problems/13.html

Since the duplicating rule admits an infinite sequence, Corollary 4.14 cannot succeed.

In the sequel we let $\mathcal{G}$ be the signature consisting of unary function symbols $\vec{f}$ for every n-ary function symbol $f \in \mathcal{F}$.

Definition 4.19. Let $x \in \mathcal{V}$. We define a partial mapping $\star$ from terms in the original signature and positions $\mathcal{T}(\mathcal{F},\mathcal{V}) \times \mathbb{N}_+^*$ to terms in $\mathcal{T}(\mathcal{G},\mathcal{V})$ as follows:

$$\star(f(\vec{t}),p) = \begin{cases} f_i(\star(t_i,q)) & \text{if } p = iq \\ x & \text{if } p = \epsilon \end{cases}$$

For a TRS $\mathcal{R}$ we abbreviate $\mathcal{R}^\star_> / \mathcal{R}^\star_=$ by $\star(\mathcal{R})$. Here, for $\gtrsim \in \{>,=\}$, $\mathcal{R}^\star_\gtrsim$ consists of all rules $\star(l,p) \to \star(r,q)$ such that $l \to r \in \mathcal{R}$, $l|_p = r|_q = y \in \mathcal{V}$, and $|r|_y \gtrsim 1$.

The next example illustrates the transformation $\star(\cdot)$.

Example 4.20. Consider the TRS $\mathcal{R}$ from Example 4.18. The relative TRS $\star(\mathcal{R}) = \mathcal{R}^\star_> / \mathcal{R}^\star_=$ consists of the TRS $\mathcal{R}^\star_>$ with rules

$$\mathsf{f}_1(\mathsf{h}_1(x)) \to \mathsf{h}_1(\mathsf{g}_1(\mathsf{f}_1(x))) \qquad\qquad \mathsf{f}_1(\mathsf{h}_1(x)) \to \mathsf{h}_1(\mathsf{g}_2(x))$$

and the TRS $\mathcal{R}^\star_=$ which is empty.

Due to the next lemma a termination proof of $\star(\mathcal{R})$ yields an LL-labeling.

Lemma 4.21. *Let $\mathcal{R}$ be a TRS and $\ell_\star(s \to^\pi t) = \star(s,p_\pi)$. Then $(\ell_\star,\geqslant,>)$ is an LL-labeling, provided $(\geqslant,>)$ is a monotone reduction pair, $\mathcal{R}^\star_> \subseteq >$, and $\mathcal{R}^\star_> \cup \mathcal{R}^\star_= \subseteq \geqslant$.*

Proof. Because $(\geqslant,>)$ is a monotone reduction pair, $(\ell_\star,\geqslant,>)$ is a labeling for $\mathcal{R}$. Note that monotonicity and stability are with respect to the signature $\mathcal{G}$. To see that the constraints of Definition 4.8 are satisfied we argue as follows. For Figure 4.2(a) we have $\alpha = \gamma$ and $\beta = \delta$ because the steps drawn at opposing sides in the diagram take place at the same positions and the function symbols above these positions stay the same. Next we consider Figure 4.2(b), i.e., the right-linear case. Recall the local peak (4.1). Again we have $\beta = \delta$ because $q < p$. To see $\alpha \geqslant \gamma$ consider the step $s \to^{q,l_2\to r_2} u$ and let q' be the unique position in $\mathcal{P}os_\mathcal{V}(l_2)$ such that $qq'r = p$ with $x = l_2|_{q'}$ for some position r. If $|r_2|_x = 0$ then there is no step and we are done. Otherwise let q'' be the position in r_2 with $|r_2|_{q''} = x$. By construction $\mathcal{R}^\star_=$ contains the rule $\star(l_2,q') \to \star(r_2,q'')$. Combining the assumption $\mathcal{R}^\star_= \subseteq \geqslant$ with monotonicity and stability of $\ell_\star$ yields $\star(s,p) \geqslant \star(u,qq''r)$, i.e., $\alpha \geqslant \gamma$. Next we consider Figure 4.2(c) for the duplicating case. Recall the local peak (4.1). Again we have $\beta = \delta$ because $q < p$. To see $\alpha > \overline{\gamma}$ (for any permutation of the steps) consider the step $s \to^{q,l_2\to r_2} u$ and let q' be the unique position in $\mathcal{P}os_\mathcal{V}(l_2)$ such that $qq'r = p$ for some position r. Let $x = l_2|_{q'}$ and $Q = \{\vec{q'}\}$ with $r_2|_{q'_i} = x$. Then $P = \{qq'_ir \mid q'_i \in Q\}$ is the set of descendants of p. By construction $\mathcal{R}^\star_>$ contains all rules $\star(l_2,q') \to \star(r_2,q'_i)$ for $1 \leqslant i \leqslant n$. Combining the assumption $\mathcal{R}^\star_> \subseteq >$ with monotonicity and stability of $\ell_\star$ yields $\star(s,p) > \star(u,p'_i)$ for $p'_i \in P$. Since $u \twoheadrightarrow^P v$ we obtain $\alpha > \gamma_i$ for $1 \leqslant i \leqslant n$ and hence the desired $\alpha > \overline{\gamma}$. □

Remark 4.22. It is also possible to formulate Lemma 4.21 as a relative termination criterion without the use of a monotone reduction pair. However, the monotone reduction pair may admit more labels to be comparable (in the critical diagrams) because of the inclusions $\mathcal{R}^\star_> \subseteq >$ and $\mathcal{R}^\star_> \cup \mathcal{R}^\star_= \subseteq \geqslant$.

From Lemma 4.21 we obtain the following corollary.

Corollary 4.23. *Let $\mathcal{R}$ be a left-linear TRS and let ℓ be a weak LL-labeling. Let $\ell_\star\ell$ denote $\ell \times \ell_\star$ or $\ell_\star \times \ell$. Let $(\geqslant, >)$ be a monotone reduction pair showing termination of $\star(\mathcal{R})$. If the critical peaks of $\mathcal{R}$ are decreasing for $\ell_\star\ell$ then $\mathcal{R}$ is confluent.*

Proof. The function $\ell_\star$ is an LL-labeling by Lemma 4.21. Lemma 4.12 yields that $\ell_\star\ell$ is an LL-labeling. By assumption the critical peaks are decreasing for $\ell_\star\ell$ and hence Theorem 4.10 yields the confluence of $\mathcal{R}$. □

The next example illustrates the use of Corollary 4.23.

Example 4.24. We show confluence of the TRS $\mathcal{R}$ from Example 4.18. Termination of $\star(\mathcal{R})$ (cf. Example 4.20) is easily shown, e.g., the polynomial interpretation

$$\mathsf{f}_{1\mathbb{N}}(x) = 2x \qquad \mathsf{g}_{1\mathbb{N}}(x) = \mathsf{g}_{2\mathbb{N}}(x) = x \qquad \mathsf{h}_{1\mathbb{N}}(x) = x + 1$$

orients both rules in $\mathcal{R}^\star_>$ strictly. To show decreasingness of the three critical peaks (two of which are symmetric) we use the labeling $\ell_\star \times \ell^i_{\mathrm{rl}}$ with $i(\mathsf{f}(\mathsf{h}(x)) \to \mathsf{h}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a}))))) = 1$ and all other rules receive label 0. For the moment we label a step $s \to^\pi t$ with the interpretation of $\star(s, p_\pi)$. E.g., a step $\mathsf{f}(\mathsf{h}(\mathsf{b})) \to \mathsf{f}(\mathsf{h}(\bot))$ is labeled $2x + 2$ since $\star(\mathsf{f}(\mathsf{h}(\mathsf{b})), 11) = \mathsf{f}_1(\mathsf{h}_1(x))$ and $[\mathsf{f}_1(\mathsf{h}_1(x))]_\mathbb{N} = 2x+2$. The critical peak $\mathsf{h}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) \;{}_{x,1}\!\leftarrow \mathsf{f}(\mathsf{h}(x)) \to_{x,0} \mathsf{a}$ is closed decreasingly by

$$\mathsf{h}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) \underset{x,0}{\longrightarrow} \mathsf{c} \underset{x,0}{\longrightarrow} \bot \underset{x,0}{\longleftarrow} \mathsf{b} \underset{x,0}{\longleftarrow} \mathsf{a}$$

and the critical peak $\mathsf{h}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) \;{}_{x,1}\!\leftarrow \mathsf{f}(\mathsf{h}(x)) \to_{2x,0} \mathsf{f}(\mathsf{c})$ is closed decreasingly by

$$\mathsf{h}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) \underset{x,0}{\longrightarrow} \mathsf{c} \underset{x,0}{\longrightarrow} \bot \underset{x,0}{\longleftarrow} \mathsf{b} \underset{x,0}{\longleftarrow} \mathsf{a} \underset{x,0}{\longleftarrow} \mathsf{f}(\mathsf{c})$$

which allows to prove confluence of $\mathcal{R}$ by Corollary 4.23.

By definition of $\alpha > \overline{\gamma}$ (cf. Definition 4.8) we observe that the definition of $\star(\mathcal{R})$ can be relaxed. If $l_2 \to r_2$ with $l_2|_{q'} = x \in \mathcal{V}$ and $\{\vec{q'}\}$ are the positions of the variable x in r_2 then it suffices if $n-1$ instances of $\star(l_2, q') \to \star(r_2, q'_i)$ are put in $\mathcal{R}^\star_>$ while one $\star(l_2, q') \to \star(r_2, q'_j)$ can be put in $\mathcal{R}^\star_=$ (since the steps labeled $\overline{\gamma}$ in Figure 4.2(c) are at parallel positions we can choose the first closing step such that $\alpha \geqslant \gamma_1$). This improved version of $\star(\mathcal{R})$ is denoted by $\overset{\star}{\star}(\mathcal{R}) = \mathcal{R}^{\star\star}_>/\mathcal{R}^{\star\star}_=$. We obtain the following variant of Lemma 4.21.

Lemma 4.25. *Let $\mathcal{R}$ be a TRS. Then $(\ell_\star, \geqslant, >)$ is an LL-labeling, provided $(\geqslant, >)$ is a monotone reduction pair, $\mathcal{R}^{\star\star}_> \subseteq >$, and $\mathcal{R}^{\star\star}_> \cup \mathcal{R}^{\star\star}_= \subseteq \geqslant$.* □

Obviously any $\overset{\star}{\star}(\mathcal{R})$ inherits termination from $\star(\mathcal{R})$. The next example shows that the reverse statement does not hold. In Section 4.5 we show how the intrinsic indeterminacy of $\overset{\star}{\star}(\mathcal{R})$ is eliminated in the implementation.

Example 4.26. Consider the TRS $\mathcal{R}$ from Example 4.1. The TRS $\mathcal{R}^\star_>$ consists of the rules

$$\begin{array}{lll}
\mathsf{sq}_1(x) \to \times_1(x) & \mathsf{sq}_1(\mathsf{s}_1(x)) \to +_1(\times_1(x)) & \times_1(x) \to +_1(x) \\
\mathsf{sq}_1(x) \to \times_2(x) & \mathsf{sq}_1(\mathsf{s}_1(x)) \to +_1(\times_2(x)) & \dagger\colon \times_1(x) \to +_2(\times_1(x)) \\
 & \mathsf{sq}_1(\mathsf{s}_1(x)) \to +_2(\mathsf{s}_1(+_1(x))) & \dagger\colon \times_2(y) \to +_1(\times_2(y)) \\
 & \mathsf{sq}_1(\mathsf{s}_1(x)) \to +_2(\mathsf{s}_1(+_2(x))) & \times_2(y) \to +_2(y)
\end{array}$$

while $\mathcal{R}^\star_=$ consists of the rules

$$\begin{array}{lll}
+_1(x) \to +_1(+_1(x)) & +_1(x) \to +_2(x) & +_1(x) \to +_1(\mathsf{s}_1(x)) \\
+_2(+_1(y)) \to +_1(+_2(y)) & +_2(y) \to +_1(y) & +_2(\mathsf{s}_1(y)) \to +_2(y) \\
+_2(+_2(z)) \to +_2(z) & \times_1(x) \to \times_2(x) & \times_2(\mathsf{s}_1(y)) \to +_2(\times_2(y)) \\
+_1(+_1(x)) \to +_1(x) & \times_2(y) \to \times_1(y) & \times_1(\mathsf{s}_1(x)) \to +_1(\times_1(x)) \\
+_1(+_2(y)) \to +_2(+_1(y)) & +_1(\mathsf{s}_1(x)) \to +_1(x) & \\
+_2(z) \to +_2(+_2(z)) & +_2(y) \to +_2(\mathsf{s}_1(y)) &
\end{array}$$

Let $\mathcal{R}^\star_\dagger$ denote the rules in $\mathcal{R}^\star_>$ marked with $\dagger$. Termination of $\star(\mathcal{R})$ cannot be established (because $\mathcal{R}^\star_\dagger$ is non-terminating) but we stress that moving these rules into $\mathcal{R}^\star_=$ yields a valid $\overset{\star}{\star}(\mathcal{R})$ which can be proved terminating by the polynomial interpretation with

$$\mathsf{sq}_{1\mathbb{N}}(x) = x + 2 \qquad \times_{1\mathbb{N}}(x) = \times_{2\mathbb{N}}(x) = x + 1$$

that interprets the remaining function symbols by the identity function. We remark that Corollary 4.23 with the labeling from Lemma 4.25 establishes confluence of $\mathcal{R}$. Since all reductions in the 34 joining sequences have only $+$ above the redex and $+_{1\mathbb{N}}(x) = +_{2\mathbb{N}}(x) = x$, the $\ell_\star$ labeling attaches x to any of these steps. The rule labeling that assigns $i(3) = i(6) = 2$, $i(4) = i(10) = 1$, and 0 to all other rules shows the 34 critical peaks decreasing.

Measuring the Contracted Redex

Instead of the labeling $\ell_\star$, which is based on the context above the contracted redex, one can also use the contracted redex itself for labeling.

Lemma 4.27. *Let $\mathcal{R}$ be a TRS and $\ell_\triangle(s \to^\pi t) = s|_{p_\pi}$. Then $(\ell_\triangle, \geqslant, >)$ is a weak LL-labeling, provided $(\geqslant, >)$ is a monotone reduction pair with $\mathcal{R} \subseteq \geqslant$.*

Proof. Because $(\geqslant, >)$ is a monotone reduction pair, $(\ell_\triangle, \geqslant, >)$ is a labeling for $\mathcal{R}$. To see that the constraints of Definition 4.8 are satisfied we argue as follows. For Figure 4.2(a) we have $\alpha = \gamma$ and $\beta = \delta$. For Figure 4.2(c) we have $\alpha = \gamma_1 = \cdots = \gamma_n$ (since the same redex is contracted) and $\beta \geqslant \delta$ by the assumption $\mathcal{R} \subseteq \geqslant$ and monotonicity and stability of $\geqslant$. □

The following definition collects the constraints, such that variable overlaps can be made decreasing.

Definition 4.28. For a TRS $\mathcal{R}$ let $\mathcal{R}^{\triangle} = \{l \to x \mid l \to r \in \mathcal{R} \text{ and } |r|_x > 1\}$.

Due to the next result a termination proof of $\mathcal{R}^{\triangle}/\mathcal{R}$ enables a weak LL-labeling to establish confluence.

Corollary 4.29. *Let $\mathcal{R}$ be a left-linear TRS and let ℓ be a weak LL-labeling. Let $(\geqslant, >)$ be a simple monotone reduction pair showing termination of $\mathcal{R}^{\triangle}/\mathcal{R}$. If the critical peaks of $\mathcal{R}$ are decreasing for $\ell_{\triangle} \times \ell$ then $\mathcal{R}$ is confluent.*

Proof. Note that $\ell_{\triangle} \times \ell$ is a weak LL-labeling (cf. Remark 4.13), which shows the peaks in Figure 4.2(a) and Figure 4.2(b) decreasing. For the duplicating case of Figure 4.2(c) we inspect the labels with regard to $\ell_{\triangle}$. Consider the local peak (4.1). Clearly, $\beta = l_2\sigma$ and $\alpha = l_1\sigma$. Since $\gamma_i = \alpha$, we want to establish $\beta > \alpha$. To this end let $q' \in \mathcal{P}os_{\mathcal{V}}(l_2)$ such that $qq'r = p$ and $x = l_2|_{q'}$. Note that $l_2 \to x \in \mathcal{R}^{\triangle}$ because we are in the duplicating case. Hence the relative termination assumption gives $l_2 > x$, and $l_2\sigma > x\sigma$ is obtained by stability. Now as $x\sigma|_r = l_1\sigma$ the desired $\beta > \alpha$ follows from simplicity of the reduction pair since $l_2\sigma > x\sigma \geqslant l_1\sigma$. Combining $\ell_{\triangle}$ lexicographically with a weak LL-labeling ℓ into $\ell_{\triangle} \times \ell$ maintains decreasingness. □

Remark 4.30. Note that the labeling $\ell_{\triangle} \times \ell$ from Corollary 4.29 is not an LL-labeling. The point is that there are multiple ways of ensuring decreasingness of Figure 4.2(c). For LL-labelings, we use $\alpha > \overline{\gamma}$, while in Corollary 4.29, $\beta > \gamma_i$ for $1 \leqslant i \leqslant n$ does the job. This is also the reason why $\ell \times \ell_{\triangle}$ cannot be used in Corollary 4.29. Consider the TRS with the rules $1 : \mathsf{f}(x) \to \mathsf{g}(x, x)$ and $2 : \mathsf{a} \to \mathsf{b}$. Let ℓ_{rl} be the rule labeling attaching the rule numbers as labels. Then the variable overlap is not decreasing for $\ell_{\mathrm{rl}} \times \ell_{\triangle}$.

We demonstrate Corollary 4.29 on the TRS from Example 4.16.

Example 4.31. Consider the TRS from Example 4.16. The polynomial interpretation

$$\mathsf{g}_{\mathbb{N}}(x, y) = 2x + 2y + 1 \qquad \mathsf{a}_{\mathbb{N}} = \mathsf{b}_{\mathbb{N}} = 0 \qquad \mathsf{f}_{\mathbb{N}}(x) = x^2$$

establishes relative termination of $\{\mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to x\}/\mathcal{R}$ and shows the critical peak decreasing when labeling steps with the pair obtained by the interpretation of the redex and the rule labeling, i.e., $t = \mathsf{f}(\mathsf{g}(x, \mathsf{b}))\ {}_{0,2}\!\leftarrow \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_{(2x+1)^2,3} \mathsf{g}(\mathsf{f}(x), \mathsf{f}(x)) = u$ for the peak and $t \to_{0,1} \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_{(2x+1)^2,3} u$ for the join.

Exploiting Persistence

In this section we show how to exploit persistence of confluence [4, 19] to enhance the applicability of L-labelings to certain duplicating left-linear TRSs. Compared to the previous labelings (based on duplicating steps, or context or redex of the rewrite steps), where variable overlaps were closed decreasingly by a relative termination criterion, here persistence arguments are employed to avoid reasoning about variable overlaps at duplicating variable positions at all. To this end we recall order-sorted TRSs.

Definition 4.32. Let S be a set of sorts equipped with a partial order $\leqslant$. A signature $\mathcal{F}$ and a set of variables $\mathcal{V}$ are S-sorted if every n-ary function symbol $f \in \mathcal{F}$ is equipped with a sort declaration $\alpha_1 \times \cdots \times \alpha_n \to \alpha$ where $\vec{\alpha}, \alpha \in S$ and every variable $x \in \mathcal{V}$ has exactly one sort $\alpha \in S$. We write $S(f) = \alpha$, $S(f,i) = \alpha_i$ for $1 \leqslant i \leqslant n$, and $S(x) = \alpha$, respectively. We let $\mathcal{V}_\alpha = \{x \in \mathcal{V} \mid S(x) = \alpha\}$ and require that $\mathcal{V}_\alpha$ is infinite for all $\alpha \in S$. The set of S-sorted terms, $\mathcal{T}_S(\mathcal{F},\mathcal{V})$, is the union of the sets $\mathcal{T}_\alpha(\mathcal{F},\mathcal{V})$ for $\alpha \in S$ that are inductively defined as follows: $\mathcal{V}_\alpha \subseteq \mathcal{T}_\alpha(\mathcal{F},\mathcal{V})$ and $f(\vec{t}) \in \mathcal{T}_\alpha(\mathcal{F},\mathcal{V})$ whenever $f \in \mathcal{F}$ has sort declaration $\alpha_1 \times \cdots \times \alpha_n \to \alpha$ and $t_i \in \mathcal{T}_{\leqslant\alpha_i}(\mathcal{F},\mathcal{V})$ for all $1 \leqslant i \leqslant n$. Here $\mathcal{T}_{\leqslant\alpha}(\mathcal{F},\mathcal{V})$ is the union of all $\mathcal{T}_\beta(\mathcal{F},\mathcal{V})$ for $\beta \leqslant \alpha$.

The notion of S-sorted terms properly extends many-sorted terms. Indeed, if we let $\leqslant$ be the identity relation then $\mathcal{T}_{\leqslant\alpha}(\mathcal{F},\mathcal{V}) = \mathcal{T}_\alpha(\mathcal{F},\mathcal{V})$, which means that the i-th argument of f in an S-sorted term must have sort $S(f,i)$.

Definition 4.33. We extend $S(\cdot)$ and $S(\cdot,\cdot)$ to S-sorted terms t and non-root positions of t. If $t = f(\vec{t})$ then $S(t) = S(f)$, $S(t,i) = S(f,i)$, and $S(t,ip) = S(t_i,p)$ for $p \neq \epsilon$. If $t = x \in \mathcal{V}$ then $S(t) = S(x)$.

Example 4.34. Let $S = \{0,1,2\}$ with $0 \leqslant 1$ and consider the sort declarations $\mathsf{f} : 1 \to 2$ and $x : 0$. Then $t = \mathsf{f}(x) \in \mathcal{T}_S(\{\mathsf{f}\},\{x\})$, $S(t) = 2$, $S(t,1) = 1$, and $S(t|_1) = 0 \leqslant 1$.

One easily observes that $S(t,p)$ defines the maximal sort induced by the context $t[\Box]_p$: a term $t[u]_p$ is S-sorted if and only if $u \in \mathcal{T}_{\leqslant S(t,p)}(\mathcal{F},\mathcal{V})$. Consequently, we have $S(t|_p) \leqslant S(t,p)$ for all non-root positions p of t.

We are particularly interested in the case where rewriting restricted to S-sorted terms coincides with ordinary rewriting with initial terms restricted to S-sorted ones. This property is captured by S-compatible TRSs.

Definition 4.35. A TRS $\mathcal{R}$ is *S-compatible* if for every rule $l \to r \in \mathcal{R}$ there exists a sort $\alpha \in S$ such that $l \in \mathcal{T}_\alpha(\mathcal{F},\mathcal{V})$ and $r \in \mathcal{T}_{\leqslant\alpha}(\mathcal{F},\mathcal{V})$, and $S(l,p) = S(l|_p)$ for all $p \in \mathcal{P}os_\mathcal{V}(l)$.

The following lemma is well-known (e.g. [77]) and easy to prove.

Lemma 4.36. *If $\mathcal{R}$ is S-compatible then $\mathcal{T}_S(\mathcal{F},\mathcal{V})$ and $\mathcal{T}_{\leqslant\alpha}(\mathcal{F},\mathcal{V})$ for every $\alpha \in S$ are closed under rewriting by $\mathcal{R}$.* □

The following result is a generalization of persistency of modularity [3] to ordered sorts, and will be proved in Chapter 5 (cf. Theorem 5.65).

Theorem 4.37. *An S-compatible left-linear TRS $\mathcal{R}$ is confluent on $\mathcal{T}(\mathcal{F},\mathcal{V})$ if and only if it is confluent on $\mathcal{T}_S(\mathcal{F},\mathcal{V})$.* □

Example 4.38. Consider the duplicating TRS $\mathcal{R}$ with rules

$$1\colon\ \mathsf{f}(\mathsf{a}) \to \mathsf{f}(\mathsf{b}) \qquad\qquad 2\colon\ \mathsf{f}(x) \to \mathsf{g}(\mathsf{f}(x),\mathsf{f}(x))$$

Recall that L-labelings (in particular, rule labelings) that are not LL-labelings are not applicable to non-linear TRSs because the variable overlap diagram (Figure 4.2(c)) is not decreasing. Let $S = \{0, 1\}$ with the following sort declarations:

$$x : 0 \qquad \mathsf{a} : 0 \qquad \mathsf{b} : 0 \qquad \mathsf{f} : 0 \to 1 \qquad \mathsf{g} : 1 \times 1 \to 1$$

The TRS $\mathcal{R}$ is S-compatible and hence we may restrict rewriting to S-sorted terms without affecting confluence by Theorem 4.37. This has the beneficial effect that variable overlaps are ruled out. To see how, note that no subterms of sort 1 can appear inside terms of sort 0. Consider the left-hand side $\mathsf{f}(x)$ of $\mathcal{R}$. We have $S(\mathsf{f}(x), 1) = 0$, so that any term substituted for x must have sort 0. Further note that both left-hand sides have sort 1. Consequently, no rule application may be nested below $\mathsf{f}(x) \to \mathsf{g}(\mathsf{f}(x), \mathsf{f}(x))$ and hence variable overlaps are ruled out. Therefore, we may use L-labelings to show confluence of $\mathcal{R}$ even though $\mathcal{R}$ is not linear, and in fact the rule labeling which takes the rule numbers as labels allows us to join the sole (modulo symmetry) critical peak $t = \mathsf{f}(\mathsf{b}) \,{}_1\!\leftarrow \mathsf{f}(\mathsf{a}) \to_2 \mathsf{g}(\mathsf{f}(\mathsf{a}), \mathsf{f}(\mathsf{a})) = u$ decreasingly: $t \to_2 \mathsf{g}(\mathsf{f}(\mathsf{b}), \mathsf{f}(\mathsf{b})) \,{}_1\!\leftarrow \mathsf{g}(\mathsf{f}(\mathsf{b}), \mathsf{f}(\mathsf{a})) \,{}_1\!\leftarrow u$.

Formally, we define $\mathcal{T}_{\trianglelefteq\alpha}(\mathcal{F}, \mathcal{V}) = \{t \mid t \trianglelefteq t' \text{ for some } t' \in \mathcal{T}_{\leqslant\alpha}(\mathcal{F}, \mathcal{V})\}$, to capture which terms may occur as subterms of terms of sort α or below.

Theorem 4.39. *Let $\mathcal{R}$ be a left-linear S-compatible TRS such that the variable $l|_p$ occurs at most once in r whenever $l \to r \in \mathcal{R}$ and $l' \to r' \in \mathcal{R}$ with $l' \in \mathcal{T}_{\trianglelefteq S(l,p)}(\mathcal{F}, \mathcal{V})$ for some $p \in \mathcal{P}os_{\mathcal{V}}(l)$. Then $\mathcal{R}$ is confluent if all its critical peaks are L-decreasing.*

Proof. By Theorem 4.37 we may restrict rewriting to S-sorted terms. The proof follows that of Theorem 4.4, except in the analysis of local peaks, where right-linearity of $\mathcal{R}$ is used, which is not among our assumptions. Instead, we argue as follows: Since $\mathcal{R}$ is left-linear, any local peak has the shape (parallel), (critical overlap), or (variable-left-linear). In the latter case, the step $s \to^{q,l'\to r'} t$ is nested below $s \to^{p,l\to r} u$, and it is easy to see that this implies $l' \in \mathcal{T}_{\trianglelefteq S(l,q')}(\mathcal{F}, \mathcal{V})$ for some variable position q' of l such that $pq' \leqslant q$. Consequently the variable $x = l|_{q'}$ occurs at most once in r by assumption, and the parallel step (which contains one rewrite step for every occurrence of x in r) is empty or a single step, resulting in a decreasing diagram. □

As a refinement of Theorem 4.39, instead of ruling out duplicating (variable-left-linear) overlaps completely, we can also add additional constraints on the labeling for the remaining variable overlaps.

Definition 4.40. Let ℓ be a weak LL-labeling for an S-compatible TRS $\mathcal{R}$. We call ℓ *persistent* if whenever rules $l \to r, l' \to r' \in \mathcal{R}$ satisfy $l' \in \mathcal{T}_{\trianglelefteq S(l,p)}(\mathcal{F}, \mathcal{V})$ for some $p \in \mathcal{P}os_{\mathcal{V}}(l)$, either $|r|_{l|_p} \leqslant 1$ or $\beta > \overline{\gamma}$ in Figure 4.2 for all resulting variable overlaps with $l' \to r'$ below $l \to r$. We call $\mathcal{R}$ persistent LL-decreasing if there is a persistent, weak LL-labeling ℓ such that all critical peaks of $\mathcal{R}$ are decreasing with respect to ℓ.

Theorem 4.41. *Let $\mathcal{R}$ be a left-linear TRS. If the critical peaks of $\mathcal{R}$ are persistent LL-decreasing then $\mathcal{R}$ is confluent.*

Proof. The proof follows along the lines of the proof of Theorem 4.39. In the case of a duplicating variable-left-linear overlap, the additional constraints ensure that the resulting diagram is decreasing. □

Example 4.42. Suppose we extend the TRS from Example 4.38 with the rule $\mathsf{a} \to \mathsf{b}$, using the same sorts:

$$1\colon\ \mathsf{f}(x) \to \mathsf{g}(\mathsf{f}(x), \mathsf{f}(x)) \qquad 2\colon\ \mathsf{f}(\mathsf{a}) \to \mathsf{f}(\mathsf{b}) \qquad 3\colon\ \mathsf{a} \to \mathsf{b}$$

Theorem 4.39 is no longer applicable, because rule 3 may be nested below rule 1, which is duplicating. However, by the preceding remark, any rule labeling with $\ell^i_{\mathrm{rl}}(1) > \ell^i_{\mathrm{rl}}(3)$ will make the corresponding variable overlaps decreasing.

Remark 4.43. Note that Theorem 4.41 does not subsume Theorem 4.39, because the former demands a weak LL-labeling whereas the latter requires only an L-labeling. If we were to restrict the L-labeling and weak LL-labeling conditions to those variable overlaps that are consistent with the sort declarations, then Theorem 4.41 would subsume Theorem 4.39. We chose not to do so because all our labelings are weak LL-labelings.

The following example shows that considering order-sorted instead of many-sorted signatures is beneficial.

Example 4.44. Consider the duplicating TRS $\mathcal{R}$ given by the rules

$$1\colon\ \mathsf{h}(\mathsf{a}, \mathsf{a}) \to \mathsf{f}(\mathsf{a}) \qquad 2\colon\ \mathsf{f}(\mathsf{a}) \to \mathsf{a} \qquad 3\colon\ \mathsf{f}(x) \to \mathsf{h}(x, x)$$

Furthermore, let $\mathcal{S} = \{0, 1\}$ with $1 > 0$ and take the sort declarations

$$\mathsf{h} : 0 \times 0 \to 1 \qquad \mathsf{f} : 0 \to 1 \qquad \mathsf{a} : 0$$

Considering only $\mathcal{S}$-sorted terms, no rule can be nested below the duplicating rule $\mathsf{f}(x) \to \mathsf{h}(x, x)$. Basically, there is one critical peak, $\mathsf{h}(\mathsf{a}, \mathsf{a}) \,{}^{3}\!\leftarrow \mathsf{f}(\mathsf{a}) \to^2 \mathsf{a}$, which is decreasingly joinable as $\mathsf{h}(\mathsf{a}, \mathsf{a}) \to^1 \mathsf{f}(\mathsf{a}) \to^2 \mathsf{a}$ by the rule labeling (using rule numbers as labels), and confluence follows by Theorem 4.39. Due to the rule $\mathsf{f}(\mathsf{a}) \to \mathsf{a}$, any many-sorted sort declaration for $\mathcal{R}$ must assign the same sorts to a and the argument and result types of f. Therefore, $\mathsf{f}(x) \to \mathsf{h}(x, x)$ may be nested below itself, and Theorems 4.39 and 4.41 would fail in connection with the rule labeling.

4.3 Labeling Parallel Rewrite Steps

In this section, rather than labeling individual rewrite steps, we will label parallel rewrite steps instead. This is inspired by the parallel moves lemma, which says that any peak $t \mathrel{\mathrlap{\Leftarrow}\Leftarrow} s \Rightarrow\!\!\!\!\Rightarrow u$ of two non-overlapping parallel rewrite steps can be joined in a diamond as $t \Rightarrow\!\!\!\!\Rightarrow \cdot \mathrel{\mathrlap{\Leftarrow}\Leftarrow} u$, and diamonds are comparatively easy to label decreasingly, as we saw in Section 4.2.1.

The main problem is to label parallel steps such that variable overlaps are decreasing. The multiset of the single steps' labels does not work since $\{\alpha\} \not\geqslant_{\mathsf{mul}} \{\alpha, \ldots, \alpha\}$. Hence we use sets to label parallel steps which we denote by capital Greek letters. Sets of labels are ordered by the Hoare preorder of $(\geqslant, >)$, which we denote by $(\geqslant_H, >_H)$ and is defined by

$$\begin{aligned} \Gamma >_H \Delta \quad &\iff \quad \Gamma \neq \varnothing \wedge \forall \beta \in \Delta\, \exists\, \alpha \in \Gamma\, (\alpha > \beta) \\ \Gamma \geqslant_H \Delta \quad &\iff \quad \forall \beta \in \Delta\, \exists\, \alpha \in \Gamma\, (\alpha \geqslant \beta) \end{aligned}$$

For readability we drop the subscript H when attaching labels to rewrite steps as in $\twoheadrightarrow_{\bigvee \Gamma}$.

Example 4.45. Let $\geqslant$ denote the natural order on $\mathbb{N}$. Then $\{1\} \geqslant_H \{0, 1\}$ and $\{1\} \geqslant_H \{1, 1, 1\} = \{1\}$ but $\{5, 4\} \not> _H \{5, 3\}$.

The following lemma states obvious properties of Hoare preorders which we implicitly use in the sequel.

Lemma 4.46. *Let $(\geqslant_H, >_H)$ be a Hoare preorder.*

1. *If $(\geqslant, >)$ is a monotone reduction pair then $(\geqslant_H, >_H)$ is a monotone reduction pair.*
2. *If $\Gamma \supseteq \Gamma'$ then $\Gamma \geqslant_H \Gamma'$.*
3. *If $\Gamma >_H \Gamma'$ and $\Delta >_H \Delta'$ then $\Gamma \cup \Delta >_H \Gamma' \cup \Delta'$.*
4. *If $\Gamma \geqslant_H \Gamma'$ and $\Delta \geqslant_H \Delta'$ then $\Gamma \cup \Delta \geqslant_H \Gamma' \cup \Delta'$.* □

As we have seen in Section 4.2.2, constructing LL-labelings is quite a bit harder than constructing L-labelings, because of the duplicated steps in the (variable-left-linear) case (Figure 4.2(c)). Here, we use weak LL-labelings for labeling single and parallel rewrite steps. Throughout this section we assume a given left-linear TRS $\mathcal{R}$, and a weak LL-labeling ℓ with corresponding labeling function for parallel steps $\ell^{\parallel}$, as introduced in the following definition.

Definition 4.47. We lift a weak LL-labeling ℓ to parallel steps $t \twoheadrightarrow^P t'$ as follows. For each $\pi \in P$, we have a rewrite step $t \to^\pi t^\pi$. We label $t \twoheadrightarrow^P t'$ by $\ell^{\parallel}(t \twoheadrightarrow^P t') = \{\ell(t \to^\pi t^\pi) \mid \pi \in P\}$.

So a parallel rewrite step is labeled by the set of the labels of the single steps making up the parallel step. We indicate labels along with the step, writing $t \twoheadrightarrow^P_\Gamma t'$.

The next example shows that the labels change when decomposing a parallel step into a sequence of single steps, i.e., the label of the parallel step may be different from the union of labels of the single steps. However, the proof of Lemma 4.49 reveals that for weak LL-labelings the labels never increase when sequencing a parallel step.

Example 4.48. Consider the rule $\mathsf{a} \to \mathsf{b}$ and the extension of the source labeling $\ell(s \to t) = s$ to parallel steps. Then $\mathsf{f}(\mathsf{a}, \mathsf{a}) \twoheadrightarrow_{\{\mathsf{f}(\mathsf{a},\mathsf{a})\}} \mathsf{f}(\mathsf{b}, \mathsf{b})$ but $\mathsf{f}(\mathsf{a}, \mathsf{a}) \twoheadrightarrow_{\{\mathsf{f}(\mathsf{a},\mathsf{a})\}} \mathsf{f}(\mathsf{b}, \mathsf{a}) \twoheadrightarrow_{\{\mathsf{f}(\mathsf{b},\mathsf{a})\}} \mathsf{f}(\mathsf{b}, \mathsf{b})$. Clearly $\{\mathsf{f}(\mathsf{a}, \mathsf{a})\} \neq \{\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{f}(\mathsf{b}, \mathsf{a})\}$.

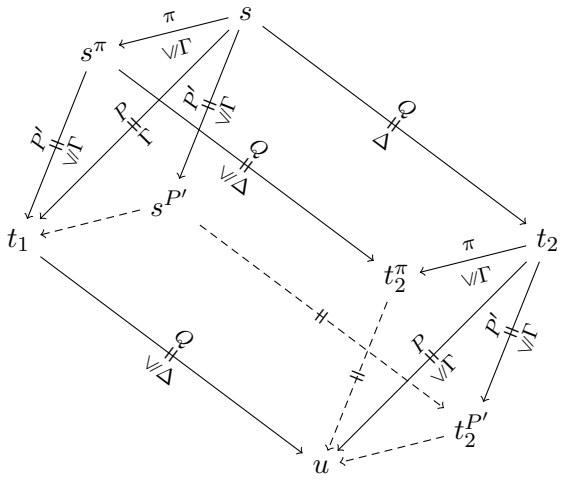


Figure 4.4: Weak LL-labeling applied to parallel steps.

This effect is intrinsic to labelings that take the context of the rewrite step into account. On the other hand, the rule labeling gives $\mathsf{f}(\mathsf{a},\mathsf{a}) \mathrel{+\!\!\!\twoheadrightarrow}_{\{1\}} \mathsf{f}(\mathsf{b},\mathsf{b})$ and $\mathsf{f}(\mathsf{a},\mathsf{a}) \mathrel{+\!\!\!\twoheadrightarrow}_{\{1\}} \mathsf{f}(\mathsf{b},\mathsf{a}) \mathrel{+\!\!\!\twoheadrightarrow}_{\{1\}} \mathsf{f}(\mathsf{b},\mathsf{b})$ with $\{1\} = \{1,1\}$, because the labels are independent of the context.

The following lemma is the key to show that even for parallel rewriting overlaps due to Figure 4.2(a) (parallel) and Figure 4.2(c) (variable-left-linear) are decreasing.

Lemma 4.49.

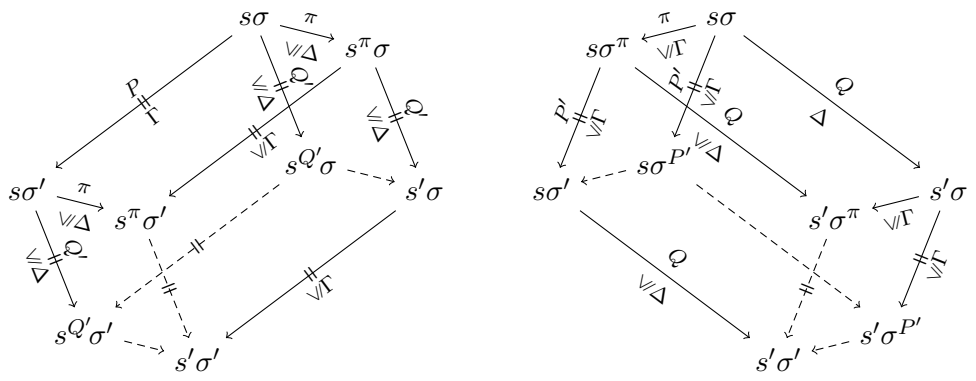
1. *Let $t_1 \mathrel{{}^{P}_{\Gamma}\!\!+\!\!\!\twoheadleftarrow} s \mathrel{+\!\!\!\twoheadrightarrow}^{Q}_{\Delta} t_2$ with $P \parallel Q$. Then there is a term u such that $s \mathrel{+\!\!\!\twoheadrightarrow}^{P\cup Q}_{\Gamma\cup\Delta} u$ and $t_1 \mathrel{+\!\!\!\twoheadrightarrow}^{Q}_{\Delta'} u \mathrel{{}^{P}_{\Gamma'}\!\!+\!\!\!\twoheadleftarrow} t_2$, where $\Gamma \geqslant_H \Gamma'$ and $\Delta \geqslant_H \Delta'$.*

2. *Let $s \mathrel{+\!\!\!\twoheadrightarrow} s'$ and $\sigma(x) \mathrel{+\!\!\!\twoheadrightarrow} \sigma'(x)$ for all $x \in \mathcal{V}$, so that there are parallel rewrite steps $s\sigma' \mathrel{{}^{P}_{\Gamma}\!\!+\!\!\!\twoheadleftarrow} s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{Q}_{\Delta} s'\sigma$. Then $s\sigma' \mathrel{+\!\!\!\twoheadrightarrow}^{Q}_{\Delta'} s'\sigma' \mathrel{{}_{\Gamma'}\!\!+\!\!\!\twoheadleftarrow} s'\sigma$ and $\Gamma \geqslant_H \Gamma'$, $\Delta \geqslant_H \Delta'$. Furthermore, if $\sigma(x) = \sigma'(x)$ for all $x \in \mathcal{V}ar(s'|_Q)$ then $s\sigma \mathrel{+\!\!\!\twoheadrightarrow}_{\Sigma} s'\sigma'$ for some $\Sigma \subseteq \Gamma \cup \Delta$.*

Proof. 1. First note that since $P \parallel Q$, a term u with $s \mathrel{+\!\!\!\twoheadrightarrow}^{P\cup Q} u$ exists. We have

$$\begin{aligned}\ell^{\parallel}(s \mathrel{\overset{P\cup Q}{+\!\!\!\twoheadrightarrow}} u) &= \{\ell(s \xrightarrow{\pi} s^\pi) \mid \pi \in P \cup Q\}\\ &= \{\ell(s \xrightarrow{\pi} s^\pi) \mid \pi \in P\} \cup \{\ell(s \xrightarrow{\pi} s^\pi) \mid \pi \in Q\}\\ &= \ell^{\parallel}(s \mathrel{\overset{P}{+\!\!\!\twoheadrightarrow}} t_1) \cup \ell^{\parallel}(s \mathrel{\overset{Q}{+\!\!\!\twoheadrightarrow}} t_2) = \Gamma \cup \Delta\end{aligned}$$

by definition. To establish $t_1 \mathrel{+\!\!\!\twoheadrightarrow}^{Q}_{\Delta'} u \mathrel{{}^{P}_{\Gamma'}\!\!+\!\!\!\twoheadleftarrow} t_2$, we use induction on $|P| + |Q|$. We consider several base cases. If $|P| = 0$ or $|Q| = 0$ then the result follows by definition of parallel rewriting. If $|P| = |Q| = 1$ the result follows from the fact that ℓ is a weak LL-labeling, Definition 4.8(1) (Figure 4.2(a)). For the induction step, assume without loss of generality that $|P| > 1$ and let $P = \{\pi\} \uplus P'$. The proof is illustrated in Figure 4.4. The parallel P-step can be decomposed into a



(a) Split base step. (b) Split substitution.

Figure 4.5: Weak LL-labeling applied to nested parallel steps.

π-step and a P'-step. Since $\{\pi\}, P' \subseteq P$, the labels are less than or equal to Γ. Then we apply the induction hypothesis to the peaks

i. $s^{P'} \overset{P'}{\curlyvee\!\Gamma}\!\!\twoheadleftarrow s \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Gamma} s^\pi$ yielding $s^\pi \twoheadrightarrow^{P'}_{\curlyvee\!\Gamma} t_1$,

ii. $s^\pi \overset{\{\pi\}}{\curlyvee\!\Gamma}\!\!\twoheadleftarrow s \twoheadrightarrow^{Q}_{\Delta} t_2$ yielding $t_2 \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Gamma} t_2^\pi$ and $s^\pi \twoheadrightarrow^{Q}_{\curlyvee\!\Delta} t_2^\pi$,

iii. $s^{P'} \overset{P'}{\curlyvee\!\Gamma}\!\!\twoheadleftarrow s \twoheadrightarrow^{Q}_{\Delta} t_2$ yielding $t_2 \twoheadrightarrow^{P'}_{\curlyvee\!\Gamma} t_2^{P'}$, which we merge with $t_2 \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Gamma} t_2^\pi$ to obtain $t_2 \twoheadrightarrow^{P}_{\curlyvee\!\Gamma} u$, noting that the union of two sets from $\curlyvee\Gamma$ is again in $\curlyvee\Gamma$, and finally

iv. $t_1 \overset{P'}{\curlyvee\!\Gamma}\!\!\twoheadleftarrow s^\pi \twoheadrightarrow^{Q}_{\curlyvee\!\Delta} t_2^\pi$ yielding $t_1 \twoheadrightarrow^{Q}_{\curlyvee\!\Delta} u$.

2. The existence of parallel rewrite steps $s\sigma' \twoheadrightarrow s'\sigma'$ and $s'\sigma \twoheadrightarrow s'\sigma'$ follows easily from the definition of parallel steps. We establish $\Gamma \geqslant_H \Gamma'$ and $\Delta \geqslant_H \Delta'$ by induction on $|Q|$. The reasoning for the induction step ($|Q| > 1$) is very similar to the induction step in item 1, cf. Figure 4.5(a): Taking $Q = \{\pi\} \uplus Q'$, we split $s\sigma \twoheadrightarrow^{Q}_{\Delta} s'\sigma$ into $s\sigma \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Delta} s^\pi\sigma$ and $s\sigma \twoheadrightarrow^{Q'}_{\curlyvee\!\Delta} s^{Q'}\sigma$. We apply the induction hypothesis to the peaks

i. $s\sigma' \overset{P}{\Gamma}\!\!\twoheadleftarrow s\sigma \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Delta} s^\pi\sigma$ yielding $s\sigma' \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Delta} s^\pi\sigma'$ and $s^\pi\sigma \twoheadrightarrow_{\curlyvee\!\Gamma} s^\pi\sigma'$,

ii. $s\sigma' \overset{P}{\Gamma}\!\!\twoheadleftarrow s\sigma \twoheadrightarrow^{Q'}_{\curlyvee\!\Delta} s^{Q'}\sigma$ yielding $s\sigma' \twoheadrightarrow^{Q'}_{\curlyvee\!\Delta} s^{Q'}\sigma'$, which can be merged with $s\sigma' \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Delta} s^\pi\sigma'$ to obtain $s\sigma' \twoheadrightarrow^{Q}_{\curlyvee\!\Delta} s'\sigma'$, and finally

iii. $s^\pi\sigma' \overset{}{\curlyvee\!\Gamma}\!\!\twoheadleftarrow s^\pi\sigma \twoheadrightarrow^{Q'}_{\curlyvee\!\Delta} s'\sigma$ yielding $s'\sigma \twoheadrightarrow_{\curlyvee\!\Gamma} s'\sigma'$, where $s^\pi\sigma \twoheadrightarrow^{Q'}_{\curlyvee\!\Delta} s'\sigma$ is obtained from part 1 of this lemma applied to $s^{Q'}\sigma \overset{Q'}{\curlyvee\!\Delta}\!\!\twoheadleftarrow s\sigma \twoheadrightarrow^{\{\pi\}}_{\curlyvee\!\Delta} s^\pi\sigma$.

This concludes the induction step. If $|Q| = 0$, there is nothing to show, so only the base case $|Q| = 1$ remains. Note that because $\mathcal{R}$ is left-linear, we may assume without loss of generality that s is linear. Therefore, every rewrite step of $s\sigma \twoheadrightarrow^{P} s\sigma'$ can be performed by modifying σ. For $P' \subseteq P$, we write $\sigma^{P'}$ for the substitution τ that satisfies $s\sigma \twoheadrightarrow^{P'} s\tau$, and proceed by induction on $|P|$. For

the induction step ($|P| > 1$), the argument is again almost the same as before, cf. Figure 4.5(b). Let $P = \{\pi\} \uplus P'$. We split $s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P}_{\Gamma} s\sigma'$ into $s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{\{\pi\}}_{\Vdash\Gamma} s\sigma^{\pi}$ and $s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P'}_{\Vdash\Gamma} s\sigma^{P'}$. Next we apply the induction hypothesis to the peaks

i. $s\sigma^{\pi} \mathrel{{}^{\{\pi\}}_{\Vdash\Gamma}\!\!+\!\!\!\twoheadleftarrow} s\sigma \to^{Q}_{\Delta} s'\sigma$ yielding $s\sigma^{\pi} \to^{Q}_{\Vdash\Delta} s'\sigma^{\pi}$ and $s'\sigma \mathrel{+\!\!\!\twoheadrightarrow}_{\Vdash\Gamma} s'\sigma^{\pi}$,

ii. $s\sigma^{P'} \mathrel{{}^{P'}_{\Vdash\Gamma}\!\!+\!\!\!\twoheadleftarrow} s\sigma \to^{Q}_{\Delta} s'\sigma$ yielding $s'\sigma \mathrel{+\!\!\!\twoheadrightarrow}_{\Vdash\Gamma} s'\sigma^{P'}$, which can be merged with $s'\sigma \mathrel{+\!\!\!\twoheadrightarrow}_{\Vdash\Gamma} s'\sigma^{\pi}$ to obtain $s'\sigma \mathrel{+\!\!\!\twoheadrightarrow}_{\Vdash\Gamma} s'\sigma'$, and finally

iii. $s\sigma' \mathrel{{}^{P'}_{\Vdash\Gamma}\!\!+\!\!\!\twoheadleftarrow} s\sigma^{\pi} \to^{Q}_{\Vdash\Delta} s'\sigma^{\pi}$ yielding $s\sigma' \to^{Q}_{\Vdash\Delta} s'\sigma'$, where $s\sigma^{\pi} \mathrel{+\!\!\!\twoheadrightarrow}^{P'}_{\Vdash\Gamma} s\sigma$ is obtained from part 1 of this lemma applied to $s\sigma^{\pi} \mathrel{{}^{\{\pi\}}_{\Vdash\Gamma}\!\!+\!\!\!\twoheadleftarrow} s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P'}_{\Vdash\Gamma} s\sigma^{P'}$.

This concludes the induction step. If $|P| = 0$ then there is nothing to show. Finally, if $|P| = |Q| = 1$, then we are left with a parallel or variable overlap, and we conclude by Definition 4.8(1) or 4.8(2), respectively. This concludes the proof that $\Gamma \geqslant_H \Gamma'$ and $\Delta \geqslant_H \Delta'$. Now if $\sigma(x) = \sigma'(x)$ for all $x \in \mathcal{V}ar(s'|_Q)$, then $s'\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P'} s'\sigma'$ satisfies $P' \parallel Q$. Performing the same rewrite steps on $s\sigma$, we obtain a parallel rewrite step $s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P'} s''$ with $P' \subseteq P$ and therefore $\Gamma'' = \ell^{\parallel}(s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P'} s'') \subseteq \ell^{\parallel}(s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P} s\sigma') = \Gamma$. Finally, using the first part of this lemma, we can combine the two parallel steps from $s\sigma$ into a single one, $s\sigma \mathrel{+\!\!\!\twoheadrightarrow}^{P'\cup Q}_{\Gamma''\cup\Delta} s'\sigma'$ with $\Sigma = \Gamma'' \cup \Delta \subseteq \Gamma \cup \Delta$ as claimed. □

Only Definition 4.8(1) was used in the proof of Lemma 4.49(1). This fact can be exploited for an alternative characterization of weak LL-labelings.

Corollary 4.50. *Let ℓ be a labeling. Then ℓ is a weak LL-labeling if and only if*

1. *in Figure 4.2(a), $\alpha \geqslant \gamma$ and $\beta \geqslant \delta$, and*
2. *in Figure 4.2(c), $\beta \geqslant \delta$ and $\{\alpha\} \geqslant_H \ell^{\parallel}(u \mathrel{+\!\!\!\twoheadrightarrow} v)$.*

Proof. Assume that ℓ is a weak LL-labeling. The first condition of this lemma is identical to Definition 4.8(1). For the second condition, $\beta \geqslant \delta$ follows from Definition 4.8(2). To establish $\{\alpha\} \geqslant_H \ell^{\parallel}(u \mathrel{+\!\!\!\twoheadrightarrow}^{P} v)$, we need to show that $\alpha \geqslant \ell(u \mathrel{+\!\!\!\twoheadrightarrow}^{\pi} u^{\pi})$ for all $\pi \in P$. For each π, we can arrange that $\ell(u \mathrel{+\!\!\!\twoheadrightarrow}^{\pi} u^{\pi}) = \gamma_1$ by choosing $u \mathrel{+\!\!\!\twoheadrightarrow}^{\pi} u^{\pi}$ as the first step in the permutation of $u \mathrel{+\!\!\!\twoheadrightarrow} v$, and then $\alpha \geqslant \gamma_1$ follows from Definition 4.8(2), establishing the claim.

Next assume that ℓ satisfies the conditions of this lemma. Then the condition of Definition 4.8(1) holds. To show the conditions of Definition 4.8(2), note that $\beta \geqslant \delta$ holds by assumption. Consider the parallel rewrite step $u \mathrel{+\!\!\!\twoheadrightarrow}^{P} v$ and a permutation $\vec{\pi}$ of P. We can decompose $u \mathrel{+\!\!\!\twoheadrightarrow}^{P} v$ into $u = u_0 \to^{\pi_1}_{\gamma_1} u_1 \to^{\pi_2}_{\gamma_2} \cdots \to^{\pi_n}_{\gamma_n} u_n = v$. By Lemma 4.49(1) applied to the peaks

$$\cdot \mathrel{{}^{\{\pi_i\}}_{\Vdash\{\alpha\}}\!\!+\!\!\!\twoheadleftarrow} u \mathrel{{}^{\{\pi_1,\ldots,\pi_{i-1}\}}_{\Vdash\{\alpha\}}\!\!+\!\!\!\twoheadrightarrow} u_{i-1}$$

we obtain $u_{i-1} \mathrel{+\!\!\!\twoheadrightarrow}^{\{\pi_i\}}_{\Vdash\{\alpha\}} u_i$, i.e., $\{\alpha\} \geqslant_H \{\gamma_i\}$, which is equivalent to $\alpha \geqslant \gamma_i$. Hence $\alpha \geqslant \overline{\gamma}$. □

The following lemma is used to reduce the number of parallel peaks that have to be considered in the proof of Theorem 4.54.

Lemma 4.51. *Let* $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma} \cdot \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}_{\vee\!\!\!/\Delta} \cdot \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma\Delta} t$ *and* $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma} \cdot \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}_{\vee\!\!\!/\Delta} \cdot \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma\Delta} u$ *be two rewrite sequences such that all rewrite steps in the sequence to* t *are at or below a position* p *and the rewrite steps in the sequence to* u *are parallel to* p*. Then the two rewrite sequences can be merged into* $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma} \cdot \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}_{\vee\!\!\!/\Delta} \cdot \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma\Delta} u[t|_p]_p$*.*

Proof. Let the two sequences be $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma} t_1 \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}_{\vee\!\!\!/\Delta} t_2 \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma\Delta} t$ and $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma} u_1 \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}_{\vee\!\!\!/\Delta} u_2 \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^*_{\vee\Gamma\Delta} u$. Using Lemma 4.49(1) repeatedly, we can derive a sequence

$$s \xrightarrow[\vee\Gamma]{+\!\!\!+\,*} u_1[t_1|_p]_p \xrightarrow[\vee\!\!\!/\Delta]{+\!\!\!+} u_2[t_2|_p]_p \xrightarrow[\vee\Gamma\Delta]{+\!\!\!+\,*} u[t|_p]_p$$

which establishes the claim. □

In order to perform a critical pair analysis for parallel rewrite steps, we need parallel critical pairs [74, 30].

Definition 4.52. Let $l \to r$ be a rule in a TRS $\mathcal{R}$ and P be a non-empty set of pairwise parallel redex patterns such that every $\pi \in P$ critically overlaps with l. By choosing variants of rules from $\mathcal{R}$ appropriately, we may assume that the sets $\mathcal{V}ar(l_\pi)$ for $\pi \in P$ and $\mathcal{V}ar(l)$ are pairwise disjoint. Assume that the unification problem $\{l|_{p_\pi} \approx l_\pi \mid \pi \in P\}$ has a solution and let σ be a most general unifier. Then there is a unique term l_P such that $l\sigma \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^P l_P$. We call $l_P \twoheadleftarrow\!\!\!\!\!\!+\rtimes\!\!\to r\sigma$ a *parallel critical pair*, and $l_P \twoheadleftarrow\!\!\!\!\!\!+\; l\sigma \to r\sigma$ a *parallel critical peak*.

Note that every standard critical pair also is a parallel critical pair. The following lemma states how critical pair analysis for a peak consisting of a parallel and a root rewrite step is done. It is a straightforward extension of [30, Lemma 4.7].

Lemma 4.53. *Let* $\mathcal{R}$ *be a left-linear TRS and* $t \;{}^P\!\!\twoheadleftarrow\!\!\!\!\!\!+\; s \to^\pi u$ *with* $p_\pi = \epsilon$*. Then either* $P \perp \pi$ *or there are substitutions* $\sigma \mathrel{+\!\!\!\!\!\!\twoheadrightarrow} \sigma'$ *and a parallel critical pair* $t' \twoheadleftarrow\!\!\!\!\!\!+\rtimes\!\!\to u'$ *such that* $t = t'\sigma' \;{}^{P\setminus P'}\!\!\twoheadleftarrow\!\!\!\!\!\!+\; t'\sigma \;{}^{P'}\!\!\twoheadleftarrow\!\!\!\!\!\!+\; s \to u'\sigma = u$ *with* $P' \subseteq P$*.* □

Note that left-linearity is essential for the substitutions σ and σ' to exist in Lemma 4.53. We are now ready to state and prove the main theorem of this section.

Theorem 4.54. *A left-linear TRS* $\mathcal{R}$ *is confluent if all its parallel critical peaks* $t \;{}^P_\Gamma\!\!\twoheadleftarrow\!\!\!\!\!\!+\; s \to_\Delta u$ *can be joined decreasingly as*

$$t \xrightarrow[\vee\Gamma]{*} \cdot \xrightarrow[\vee\!\!\!/\Delta]{+\!\!\!+} \cdot \xrightarrow[\vee\Gamma\Delta]{*} \cdot \xleftarrow[\vee\Gamma\Delta]{*} v \xleftarrow[\vee\!\!\!/\Gamma]{+\!\!\!+\,Q} \cdot \xleftarrow[\vee\Delta]{*} u$$

such that $\mathcal{V}ar(v|_Q) \subseteq \mathcal{V}ar(s|_P)$*.*

Proof. We show that $\mathrel{+\!\!\!\!\!\!\twoheadrightarrow}$ is decreasing, which implies confluence of $\mathcal{R}$. Let $t \;{}^P_\Gamma\!\!\twoheadleftarrow\!\!\!\!\!\!+\; s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^Q_\Delta u$. It suffices to show that

$$t \xrightarrow[\vee\Gamma]{*} \cdot \xrightarrow[\vee\!\!\!/\Delta]{+\!\!\!+} \cdot \xrightarrow[\vee\Gamma\Delta]{*} \cdot \xleftarrow[\vee\Gamma\Delta]{*} \cdot \xleftarrow[\vee\!\!\!/\Gamma]{+\!\!\!+} \cdot \xleftarrow[\vee\Delta]{*} u \tag{4.2}$$

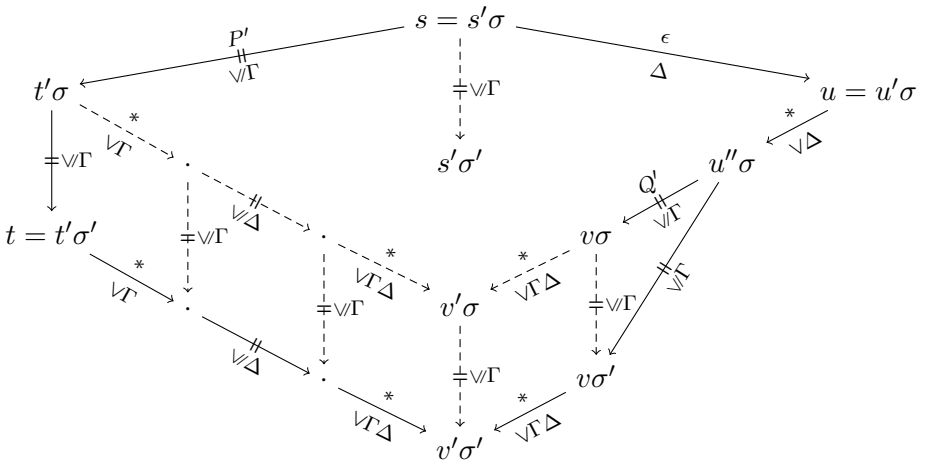


Figure 4.6: Part of the proof of Theorem 4.54.

Below we show that (4.2) holds whenever $P = \{\pi\}$ or $Q = \{\pi\}$ with $p_\pi = \epsilon$. Then for all $p \in \min\{p_\pi \mid \pi \in P \cup Q\}$, $t \overset{P}{\Gamma}\!\!\twoheadleftarrow s \twoheadrightarrow^{Q}_{\Delta} u$ induces a peak $t|_p \overset{P'}{\Gamma_0}\!\!\twoheadleftarrow s|_p \twoheadrightarrow^{Q'}_{\Delta_0} u|_p$, where $P' = \{\pi\}$ or $Q' = \{\pi\}$ for some π with $p_\pi = \epsilon$. So for each p, we obtain a joining sequence for $t|_p$ and $u|_p$ of shape (4.2). By the monotonicity of labelings, this results in joining sequences

$$s[t|_p]_p \xrightarrow[\vee\Gamma]{*} \cdot \xrightarrow[\vee\!\!/\Delta]{\shortparallel} \cdot \xrightarrow[\vee\Gamma\Delta]{*} \cdot \xleftarrow[\vee\Gamma\Delta]{*} \cdot \xleftarrow[\vee\!\!/\Gamma]{\shortparallel} \cdot \xleftarrow[\vee\Delta]{*} s[u|_p]_p$$

which are mutually parallel since the positions $p \in \min(P \cup Q)$ are mutually parallel. By repeated application of Lemma 4.51 those sequences can be combined into a single sequence of the same shape.

In order to show (4.2) for $P = \{\pi\}$ or $Q = \{\pi\}$ with $p_\pi = \epsilon$, assume without loss of generality that $Q = \{\pi\}$. If $P \perp \pi$ then $s = l_\pi\sigma$ and, because l_π is linear, there is a substitution σ' with $t = l_\pi\sigma'$ and $\sigma(x) \twoheadrightarrow \sigma'(x)$ for all variables $x \in \mathcal{V}$. We conclude by Lemma 4.49(2). Otherwise P and π overlap, and by Lemma 4.53, there are a parallel critical peak $t' \overset{P'}{\twoheadleftarrow} s' \to u'$ and substitutions σ, σ' such that $\sigma \twoheadrightarrow \sigma'$ and $t = t'\sigma' \overset{P\setminus P'}{\vee\!\!/\Gamma}\!\!\twoheadleftarrow t'\sigma \overset{P'}{\vee\!\!/\Gamma}\!\!\twoheadleftarrow s'\sigma = s \to^{\epsilon}_{\Delta} u'\sigma = u$ with $P' \subseteq P$. This case is illustrated in Figure 4.6. By assumption there are u'', v and v' with $\mathcal{V}ar(v|_{Q'}) \subseteq \mathcal{V}ar(s|_{P'})$ such that we can join t' and u' decreasingly, and consequently, using the stability of labelings we obtain

$$t'\sigma \xrightarrow[\vee\Gamma]{*} \cdot \xrightarrow[\vee\!\!/\Delta]{\shortparallel} \cdot \xrightarrow[\vee\Gamma\Delta]{*} v'\sigma \xleftarrow[\vee\Gamma\Delta]{*} v\sigma \overset{Q'}{\underset{\vee\!\!/\Gamma}{\shortparallel}}\!\!\twoheadleftarrow u''\sigma \xleftarrow[\vee\Delta]{*} u'\sigma = u$$

Furthermore, making repeated use of Lemma 4.49(2),

$$t = t'\sigma' \xrightarrow[\vee\Gamma]{*} \cdot \xrightarrow[\vee\!\!/\Delta]{\shortparallel} \cdot \xrightarrow[\vee\Gamma\Delta]{*} v'\sigma' \xleftarrow[\vee\Gamma\Delta]{*} v\sigma' \xleftarrow[\vee\!\!/\Gamma]{\shortparallel} v\sigma$$

Notably, the step $v\sigma \twoheadrightarrow_{\vee\!\!/\Gamma} v\sigma'$ is obtained from $s'\sigma \twoheadrightarrow_{\vee\!\!/\Gamma} s'\sigma'$ by passing through the rewrite sequence $s'\sigma \to u'\sigma \to^* u''\sigma \twoheadrightarrow v\sigma$. We have $\sigma(x) = \sigma'(x)$ for $x \in \mathcal{V}ar(s|_{P'})$ for otherwise $s \twoheadrightarrow_\Gamma t$ would not be a parallel step. Together with

$\mathcal{V}ar(v|_{Q'}) \subseteq \mathcal{V}ar(s|_{P'})$, the parallel steps $u''\sigma \mathrel{+\!\!\!\twoheadrightarrow}_{\parallel\Gamma} v\sigma$ and $v\sigma \mathrel{+\!\!\!\twoheadrightarrow}_{\parallel\Gamma} v\sigma'$ can be combined into a single $\mathrel{+\!\!\!\twoheadrightarrow}_{\parallel\Gamma}$ step by Lemma 4.49(2). Thus we can join t and u decreasingly with common reduct $v'\sigma'$, completing the proof. □

To conclude the section we demonstrate Theorem 4.54 on two examples. Both are based on rule labeling.

Example 4.55. Consider the TRS $\mathcal{R}$ consisting of the following five rules with labels $2 > 1 > 0$:

$$\mathsf{a} \underset{1}{\to} \mathsf{b} \qquad \mathsf{b} \underset{0}{\to} \mathsf{a} \qquad \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{1}{\to} \mathsf{c} \qquad \mathsf{f}(\mathsf{b},\mathsf{b}) \underset{2}{\to} \mathsf{c} \qquad \mathsf{h}(x) \underset{0}{\to} \mathsf{h}(\mathsf{f}(x,x))$$

There are six parallel critical peaks that can all be joined decreasingly as required by Theorem 4.54:

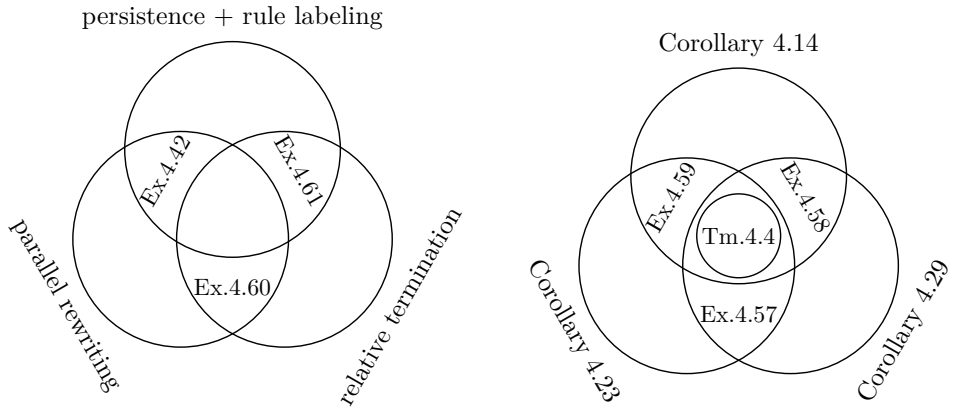
$$\begin{aligned}
\mathsf{f}(\mathsf{b},\mathsf{a}) &\underset{\{1\}}{\leftarrow\!\!\!+\!\!\!+} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} : & \mathsf{f}(\mathsf{b},\mathsf{a}) &\underset{\{0\}}{\longrightarrow} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} \\
\mathsf{f}(\mathsf{a},\mathsf{b}) &\underset{\{1\}}{\leftarrow\!\!\!+\!\!\!+} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} : & \mathsf{f}(\mathsf{a},\mathsf{b}) &\underset{\{0\}}{\longrightarrow} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} \\
\mathsf{f}(\mathsf{b},\mathsf{b}) &\underset{\{1\}}{\leftarrow\!\!\!+\!\!\!+} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} : & \mathsf{f}(\mathsf{b},\mathsf{b}) &\underset{\{0\}}{+\!\!\!\twoheadrightarrow} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} \\
\mathsf{f}(\mathsf{a},\mathsf{b}) &\underset{\{0\}}{\leftarrow\!\!\!+\!\!\!+} \mathsf{f}(\mathsf{b},\mathsf{b}) \underset{\{2\}}{\longrightarrow} \mathsf{c} : & \mathsf{f}(\mathsf{a},\mathsf{b}) &\underset{\{0\}}{\longrightarrow} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} \\
\mathsf{f}(\mathsf{b},\mathsf{a}) &\underset{\{0\}}{\leftarrow\!\!\!+\!\!\!+} \mathsf{f}(\mathsf{b},\mathsf{b}) \underset{\{2\}}{\longrightarrow} \mathsf{c} : & \mathsf{f}(\mathsf{b},\mathsf{a}) &\underset{\{0\}}{\longrightarrow} \mathsf{f}(\mathsf{a},\mathsf{a}) \underset{\{1\}}{\longrightarrow} \mathsf{c} \\
\mathsf{f}(\mathsf{a},\mathsf{a}) &\underset{\{0\}}{\leftarrow\!\!\!+\!\!\!+} \mathsf{f}(\mathsf{b},\mathsf{b}) \underset{\{2\}}{\longrightarrow} \mathsf{c} : & \mathsf{f}(\mathsf{a},\mathsf{a}) &\underset{\{1\}}{\longrightarrow} \mathsf{c}
\end{aligned}$$

Therefore, $\mathcal{R}$ is confluent.

Example 4.56. Let $\mathcal{R}$ be the TRS (Cops #62) consisting of the (labeled) rules

$$\begin{aligned}
x - 0 &\underset{0}{\to} x & 0 - x &\underset{0}{\to} 0 & \mathsf{s}(x) - \mathsf{s}(y) &\underset{0}{\to} x - y \\
0 < \mathsf{s}(x) &\underset{0}{\to} \mathsf{true} & x < 0 &\underset{0}{\to} \mathsf{false} & \mathsf{s}(x) < \mathsf{s}(y) &\underset{0}{\to} x < y \\
\mathsf{gcd}(x,0) &\underset{0}{\to} x & \mathsf{gcd}(0,x) &\underset{0}{\to} x & \mathsf{gcd}(x,y) &\underset{1}{\to} \mathsf{gcd}(y,\mathsf{mod}(x,y)) \\
\mathsf{if}(\mathsf{true},x,y) &\underset{0}{\to} x & \mathsf{if}(\mathsf{false},x,y) &\underset{0}{\to} y \\
\mathsf{mod}(x,0) &\underset{0}{\to} x & \mathsf{mod}(0,x) &\underset{0}{\to} 0 \\
\mathsf{mod}(x,\mathsf{s}(y)) &\underset{1}{\to} \mathsf{if}(x < \mathsf{s}(y), x, \mathsf{mod}(x - \mathsf{s}(y), \mathsf{s}(y)))
\end{aligned}$$

There are 12 critical pairs, 6 of which are trivial. One easily verifies that the remaining 6 pairs can be joined decreasingly, using the order $1 > 0$. Hence the confluence of $\mathcal{R}$ follows from Theorem 4.54. Even though $\mathcal{R}$ lacks proper parallel critical pairs, none of the other results in this chapter applies. Note that the preconditions for Corollaries 4.14, 4.23, and 4.29 are not satisfied as $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$, $\star(\mathcal{R})$, and $\mathcal{R}^{\triangle}$ are non-terminating (due to the rules with label 1). Finally, persistence cannot rule out variable overlaps (of the duplicating mod rule below the variable x) and hence Theorems 4.39 and 4.41 based on the rule labeling fail.



(a) All. (b) Relative Termination.

Figure 4.7: Interrelationships.

4.4 Assessment

In this section we relate the results from this chapter to each other (Section 4.4.1) and to the recent literature [1, 32] (Section 4.4.2).

4.4.1 Interrelationships

The main results for left-linear systems presented in this chapter can be divided into three classes. Those that require relative termination as a precondition (Corollaries 4.14, 4.23, and 4.29), those exploiting persistence (Theorems 4.39 and 4.41), and those considering parallel rewriting (Theorem 4.54). Figure 4.7(a) demonstrates that these three classes are incomparable. The same holds when focusing on the results relying on relative termination, cf. Figure 4.7(b). Note that the regions where only one class is applicable can be populated with examples using Toyama's celebrated modularity result [75], e.g., the disjoint union (after renaming function symbols) of the TRSs in Examples 4.60 and 4.61 can only be handled by the approach based on relative termination. We discuss the interrelationships in more detail below.

First we observe that Corollaries 4.14, 4.23, and 4.29 subsume Theorem 4.4 since the preconditions of the corollaries evaporate for linear systems. The inclusion is strict since Theorem 4.4 cannot deal with the rule $\mathsf{f}(x) \to \mathsf{g}(x,x)$, while all the corollaries can. Furthermore, Theorem 4.4 is subsumed by Theorem 4.39, which, if restricted to weak LL-labelings, is subsumed by Theorem 4.41.

The following three examples show that Corollaries 4.14, 4.23, and 4.29 are pairwise incomparable in power (for an overview see Figure 4.7(b)).

Example 4.57. Consider the TRS $\mathcal{R}$ consisting of the following rules

$$\begin{array}{lll} \mathsf{f}(\mathsf{h}(x)) \to \mathsf{k}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) & \mathsf{f}(x) \to \mathsf{a} & \mathsf{a} \to \mathsf{b} \\ \mathsf{k}(x) \to \mathsf{c} & \mathsf{b} \to \bot & \mathsf{c} \to \bot \end{array}$$

This TRS has one critical peak (modulo symmetry). Since $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ is non-terminating, Corollary 4.14 does not apply. For Corollary 4.23 observe that $\star(\mathcal{R})$

is terminating using the interpretation $\mathsf{h}_{1\mathbb{N}}(x) = x + 1$ and the identify function for all other function symbols. To show decreasingness we use the labeling $\ell_\star \times \ell^i_{\mathrm{rl}}$ with $i(\mathsf{f}(x) \to \mathsf{a}) = 1$ and all other rules receive label 0. The critical peak $t = \mathsf{a}\ {}_{x,1}\!\leftarrow \mathsf{f}(\mathsf{h}(x)) \to_{x,0} \mathsf{k}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) = u$ is closed decreasingly by $t \to_{x,0} \mathsf{b} \to_{x,0} \bot\ {}_{x,0}\!\leftarrow \mathsf{c}\ {}_{x,0}\!\leftarrow u$. Corollary 4.29 also applies since the polynomial interpretation with $\mathsf{h}_{\mathbb{N}}(x) = 3x + 1$ and interpreting all other function symbols by the sum of its arguments establishes termination of $\mathcal{R}^{\triangle}/\mathcal{R}$. When taking the identity for ℓ in Corollary 4.29 the critical peak $t = \mathsf{a}\ {}_{3x+1}\!\leftarrow \mathsf{f}(\mathsf{h}(x)) \to_{3x+1} \mathsf{k}(\mathsf{g}(\mathsf{f}(x), x, \mathsf{f}(\mathsf{h}(\mathsf{a})))) = u$ can be closed decreasingly by $t \to_0 \mathsf{b} \to_0 \bot\ {}_0\!\leftarrow \mathsf{c}\ {}_{2x+1}\!\leftarrow u$.

Example 4.58. It is easy to adapt the TRS from Example 4.16 such that $\star(\mathcal{R})$ becomes non-terminating. Consider the TRS $\mathcal{R}$

$$1\colon \mathsf{b} \to \mathsf{a} \qquad 2\colon \mathsf{a} \to \mathsf{b} \qquad 3\colon \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to \mathsf{g}(\mathsf{f}(x), \mathsf{f}(\mathsf{g}(x, \mathsf{c})))$$

for which termination of $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ is proved by LPO with precedence $\mathsf{f} > \mathsf{g}$ and $\mathsf{a} \sim \mathsf{b} > \mathsf{c}$. Corollary 4.14 applies since the rule labeling establishes decreasingness of the critical peak $t = \mathsf{f}(\mathsf{g}(x, \mathsf{b}))\ {}_2\!\leftarrow \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_3 \mathsf{g}(\mathsf{f}(x), \mathsf{f}(\mathsf{g}(x, \mathsf{c}))) = u$ by the join $t \to_1 \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_3 u$. Note that $\mathsf{f}_1(\mathsf{g}_1(x)) \to \mathsf{g}_2(\mathsf{f}_1(\mathsf{g}_1(x))) \in \mathcal{R}^{\star}_{>}$ is non-terminating and hence Corollary 4.23 does not apply.[3] For Corollary 4.29 the (above) termination proof establishes termination of $\mathcal{R}^{\triangle}/\mathcal{R}$ and $\ell_{\triangle}$ in combination with the rule labeling (taking rule numbers as labels) labels the critical peak $t = \mathsf{f}(\mathsf{g}(x, \mathsf{b}))\ {}_{\mathsf{a},2}\!\leftarrow \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_{\mathsf{f}(\mathsf{g}(x,\mathsf{a})),3} \mathsf{g}(\mathsf{f}(x), \mathsf{f}(\mathsf{g}(x, \mathsf{c}))) = u$ decreasingly since $t \to_{\mathsf{b},1} \mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to_{\mathsf{f}(\mathsf{g}(x,\mathsf{a})),3} u$.

Example 4.59. Consider the TRS consisting of the rules

$$\mathsf{a}(\mathsf{a}(\mathsf{c})) \to \mathsf{a}(\mathsf{b}(\mathsf{a}(\mathsf{c}))) \qquad \mathsf{b}(x) \to \mathsf{h}(x, x)$$

The TRS $\mathcal{R}$ has no critical peaks and is terminating by the following matrix interpretation over $\mathbb{N}^2$:

$$\mathsf{a}_{\mathbb{N}^2}(\vec{x}) = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}\vec{x} + \begin{pmatrix} 0 \\ 3 \end{pmatrix} \qquad \mathsf{h}_{\mathbb{N}^2}(\vec{x}, \vec{y}) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\vec{x} + \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\vec{y}$$

$$\mathsf{b}_{\mathbb{N}^2}(\vec{x}) = \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}\vec{x} + \begin{pmatrix} 2 \\ 0 \end{pmatrix} \qquad \mathsf{c}_{\mathbb{N}^2} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Hence also $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ is terminating, and by Corollary 4.14 the TRS $\mathcal{R}$ is confluent. Corollary 4.23 also applies since $\star(\mathcal{R})$ is terminating. The derivation $\mathsf{a}(\mathsf{a}(\mathsf{c})) \to \mathsf{a}(\mathsf{b}(\mathsf{a}(\mathsf{c}))) \to_{\mathcal{R}^{\triangle}} \mathsf{a}(\mathsf{a}(\mathsf{c})) \to \cdots$ shows that $\mathcal{R}^{\triangle}/\mathcal{R}$ is non-terminating, so Corollary 4.29 does not apply.

Note that any simple monotone reduction pair showing termination of $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ will also establish termination of $\mathcal{R}^{\triangle}/\mathcal{R}$, because if $l \to x \in \mathcal{R}^{\triangle}$ then there is a rule $l \to r \in \mathcal{R}_{\mathsf{d}}$ that duplicates x, whence $l > r \geqslant x$. Hence it is no surprise that Example 4.59 used a matrix interpretation of dimension 2.

[3] We remark that it is easy to extend this example such that also $\overset{\star}{\star}(\mathcal{R})$ is non-terminating; just consider the rule $\mathsf{f}(\mathsf{g}(x, \mathsf{a})) \to \mathsf{g}(\mathsf{f}(x), \mathsf{g}(\mathsf{f}(\mathsf{g}(x, \mathsf{c}), \mathsf{f}(\mathsf{g}(x, \mathsf{c})))))$.

Furthermore, the results on relative termination are incomparable with those on persistence and those based on parallel rewriting. To this end observe that the first rule of Example 4.42 violates all preconditions of Corollaries 4.14, 4.23, and 4.29 but Theorems 4.41 and 4.54 apply. Note that Theorem 4.41 based on arbitrary weak LL-labelings subsumes Corollaries 4.14 and 4.23, since they produce LL-labelings which may be used to close problematic variable peaks decreasingly even without persistence. However, if restricted to the rule labeling the following TRS cannot be handled using persistence while each of the Corollaries 4.14, 4.23, and 4.29 as well as Theorem 4.54 succeeds.

Example 4.60. Consider the TRS consisting of the rules

$$1\colon\ \mathsf{f}(x,y,\mathsf{a}) \to \mathsf{f}(x,x,\mathsf{b}) \qquad\qquad 2\colon\ \mathsf{f}(\mathsf{f}(x,y,\mathsf{b}),z,\mathsf{c}) \to x$$

which is orthogonal. Since a most general sort assignment cannot exclude variable overlaps of the first rule with itself, Theorem 4.41 can only succeed when used in combination with an LL-labeling. Note that all preconditions for Corollaries 4.14, 4.23, and 4.29 are satisfied and due to the lack of critical overlaps they are decreasing. For the same reason Theorem 4.54 applies.

The final example shows that Theorem 4.54 does not subsume the plain version for linear TRSs (because of the variable condition).

Example 4.61. Consider the linear TRS consisting of the single rule

$$(x+y)+z \to (z+y)+x$$

Note that all steps are labeled the same, because they use the same rule. There is only one (parallel) critical peak, $((z{+}y){+}x){+}u \leftarrow ((x{+}y){+}z){+}u \to (u{+}z){+}(x{+}y)$, which may be joined as $((z{+}y){+}x){+}u \to ((x{+}y){+}z){+}u \leftarrow (u{+}z){+}(x{+}y)$. Confluence of $\mathcal{R}$ can be established by Theorem 4.4 using the rule labeling from Lemma 4.5. On the other hand, trying to use Theorem 4.54 fails for this joining sequence, because $\mathcal{V}ar(((z+y)+x)+u) \not\subseteq \mathcal{V}ar((z+y)+x)$. All other ways of joining the critical peak fail to be decreasing because they require more than one parallel rewrite step from $((z+y)+x)+u$ or $(u+z)+(x+y)$, e.g. $((z+y)+x)+u \to ((x+y)+z)+y \to (y+z)+(x+y)$.

4.4.2 Related work

In this section we relate our results to [32, 1].

To compare our setting with the main result from [32] we define the *critical pair steps* $\mathsf{CPS}(\mathcal{R}) = \{s \to t, s \to u \mid t \leftarrow s \to u \text{ is a critical peak of } \mathcal{R}\}$. Furthermore let $\mathsf{CPS}'(\mathcal{R})$ be the critical pair steps which do not give rise to trivial critical pairs.

Theorem 4.62 ([32, Theorem 3])**.** *A left-linear locally confluent TRS $\mathcal{R}$ is confluent if $\mathsf{CPS}'(\mathcal{R})/\mathcal{R}$ is terminating.*

Using the weak LL-labeling $\ell_{\mathrm{rt}}^{\mathsf{PCPS}'(\mathcal{R})}$, from Theorem 4.54 we obtain the following corollary. Here $\mathsf{PCPS}'(\mathcal{R})$ are the parallel critical pair steps which do not give rise to trivial parallel critical pairs.

Corollary 4.63. *A left-linear TRS $\mathcal{R}$ whose parallel critical pairs are joinable is confluent if $\mathsf{PCPS}'(\mathcal{R})/\mathcal{R}$ is terminating.*

Proof. We need to show that the relative termination assumption eliminates the variable condition in Theorem 4.54. If $\mathsf{PCPS}'(\mathcal{R})/\mathcal{R}$ is terminating then for any (non-trivial) parallel critical peak $t \overset{P}{\Gamma}\!\!\twoheadleftarrow s \to_\Delta u$ we obtain $t \to^*_{\vee\Gamma} \cdot {}_{\vee\Delta}{}^*\!\leftarrow u$, hence Q can be chosen to be empty and $\varnothing = \mathcal{V}ar(v|_\varnothing) \subseteq \mathcal{V}ar(s|_P)$ trivially holds.[4] □

We stress that despite the fact that the preconditions in Corollary 4.63 require more (implementation) effort to check than those in Theorem 4.62, in theory Corollary 4.63 subsumes Theorem 4.62. To this end observe that termination of $\mathsf{PCPS}'(\mathcal{R})/\mathcal{R}$ is equivalent to termination of $\mathsf{CPS}'(\mathcal{R})/\mathcal{R}$. Furthermore joinability of the parallel critical pairs is a necessary condition for confluence just as local confluence is.

Due to the flexibility of the $\ell^{\mathcal{S}}_{\mathrm{rt}}$ labeling we can also choose $\mathcal{S}$ to be (a subset of) the *critical diagram steps* $\mathsf{CDS}(\mathcal{R}) = \{s \to t_i, s \to u_j \mid\ t_0 \leftarrow s \to u_0$ is a critical peak in $\mathcal{R}$, $t_0 \to^* t_n = u_m\ {}^*\!\leftarrow u_0$, $0 \leqslant i \leqslant n$, and $0 \leqslant j \leqslant m\}$. Using $\mathsf{CDS}(\mathcal{R})$ allows to detect a possible decrease also somewhere in the joining part of the diagrams. This incorporates (and generalizes) the idea of critical valleys [64]. However, we remark that our setting does not (yet) follow another recent trend, i.e., to drop development closed critical pairs (see [64, 33]). We leave this for future work.

Next we show that Corollary 4.23 generalizes the results from [1, Sections 5 and 6]. It is not difficult to see that the encoding presented in [1, Theorem 5.4] can be mimicked by Corollary 4.23 where linear polynomial interpretations over $\mathbb{N}$ of the shape as in (1)

$$(1)\quad f_{i\mathbb{N}}(x) = x + c_f \qquad\qquad (2)\quad f_{i\mathbb{N}}(x) = x + c_{f_i}$$

are used to prove termination of $\star(\mathcal{R})$ and $\ell_\star \times \ell_{\mathrm{rl}}$ is employed to show LL-decreasingness of the critical peaks. In contrast to [1, Theorem 5.4], which explicitly encodes these constraints in a single formula of linear arithmetic, our abstract formulation has the following advantages. First, we do not restrict to weight functions but allow powerful machinery for proving relative termination and second our approach allows to combine arbitrarily many labelings lexicographically (cf. Lemma 4.12). Furthermore we stress that our abstract treatment of $\star(\mathcal{R})$ allows to implement Corollary 4.23 based on $\overset{\star}{\star}(\mathcal{R})$ (cf. Section 4.5) which admits further gains in power (cf. Example 4.1).

The idea of the extension presented in [1, Example 6.1] amounts to using $\ell_{\mathrm{rl}} \times \ell_\star$ instead of $\ell_\star \times \ell_{\mathrm{rl}}$, which is an application of Lemma 4.12 in our setting. Finally, the extension discussed in [1, Example 6.3] suggests to use linear polynomial interpretations over $\mathbb{N}$ of the shape as in (2) to prove termination of $\star(\mathcal{R})$. Note that these interpretations are still weight functions. This explains

[4] The condition that $\{s \to t \mid u \leftarrow s \to t$ is a critical pair$\}/\mathcal{R}$ is terminating also eliminates the variable condition.

why the approach from [1] fails to establish confluence of the TRSs in Examples 4.16 and 4.18 since a weight function cannot show termination of the rules $\mathsf{f}_1(\mathsf{g}_1(x)) \to \mathsf{g}_1(\mathsf{f}_1(x))$ and $\mathsf{f}_1(\mathsf{h}_1(x)) \to \mathsf{h}_1(\mathsf{g}_1(\mathsf{f}_1(x)))$, respectively.

Note that both recent approaches [1, 32] based on decreasing diagrams fail to prove the TRS $\mathcal{R}$ from Example 4.1 confluent. The former can, e.g., not cope with the non-terminating rule $\times_1(x) \to +_0(\times_1(x))$ in $\mathcal{R}^\star_>$ (cf. Example 4.26) while overlaps with the non-terminating rule $x + y \to y + x \in \mathcal{R}$ prevent the latter approach from succeeding. In contrast, Examples 4.15 and 4.26 give two confluence proofs based on our setting.

4.5 Implementation

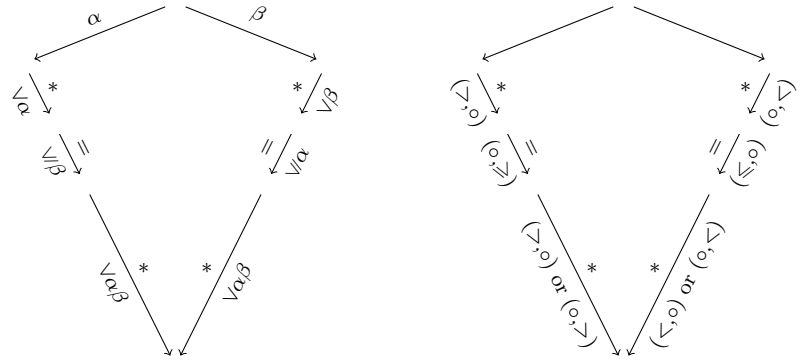
In this section we sketch how the results from this chapter are implemented in CSI.

Before decreasingness of critical peaks can be investigated, the critical pairs must be shown to be convergent. For a critical pair $t \leftarrow\!\!\rtimes\!\!\rightarrow u$ in our implementation we consider all joining sequences such that $t \to^{\leqslant n} \cdot {}^{\leqslant n}\!\leftarrow u$ and there is no smaller n that admits a common reduct. While in theory longer joining sequences might be easier to label decreasingly, preliminary experiments revealed that the effort due to the consideration of additional diagrams decreased performance.

To exploit the possibility for incremental confluence proofs by lexicographically combining labels (cf. Lemmata 4.7 and 4.12) our implementation considers lists of labels. The search for relative termination proofs (and thus the labelings) is implemented by encoding the constraints in non-linear (integer) arithmetic. Below we describe how we combine existing labels (some partial progress) with the search for a new labeling that shows the critical peaks decreasing. Note that labelings use different domains (natural numbers, terms), and, even worse, different orders (matrix interpretations, LPO, etc.). The crucial observation for incremental labeling is that neither the actual labels nor the precise order on the labels have to be recorded but only how the labels in the join relate to the labels from the peak. We use the following encoding. Let the local peak have labels $t \,{}_\alpha\!\leftarrow s \to_\beta u$. Then a step $v \to_\gamma w$ is labeled by the pair $(\circ_\alpha, \circ_\beta)$ where $\circ_\alpha$ and $\circ_\beta$ indicates if $\alpha \circ_\alpha \gamma$ and $\beta \circ_\beta \gamma$, respectively. Here $\{\circ_\alpha, \circ_\beta\} \subseteq \{>, \geqslant, ?\}$ and $?$ means that the labels are incomparable, e.g., $\mathsf{f}(x) \,?\, \mathsf{g}(y)$ in LPO or $2x + 1 \,?\, x + 2$ for (matrix) interpretations.Decreasingness as depicted in Figure 4.8(a) can then be captured by the conditions shown in Figure 4.8(b), where $\circ$ can be replaced by any symbol.

It is straightforward to implement Corollary 4.14. After establishing termination of $\mathcal{R}_\mathsf{d}/\mathcal{R}_\mathsf{nd}$ (e.g., by an external termination prover) any weak LL-labeling can be tried to show the critical peaks decreasing. In [1, 32] it is shown how the rule labeling can be implemented by encoding the constraints in linear arithmetic. Note that when using weak LL-labelings the implementation does not have to test condition 2 in Definition 4.8 since this property is intrinsic to weak LL-labelings.

We sketch how to implement the labeling $\ell^{\mathcal{S}}_{\mathrm{rt}}$ from Lemma 4.6 as a relative



(a) Decreasingness. (b) Encoding of decreasingness.

Figure 4.8: Encoding the order on the labels.

termination problem. First we fix a suitable set $\mathcal{S}$, i.e., the critical diagram steps (see Section 4.4). Facing the relative termination problem $\mathcal{S}/\mathcal{R}$ we try to simplify it according to Theorem 2.41 into some $\mathcal{S}'/\mathcal{R}'$. Note that it is not necessary to finish the proof. By Theorem 2.41 the relative TRS $(\mathcal{S} \setminus \mathcal{S}')/\mathcal{R}$ is terminating and hence by Lemma 4.6 $\ell_{\mathrm{rt}}^{\mathcal{S}\setminus\mathcal{S}'}$ is an L-labeling. Let $\geqslant = \to_{\mathcal{R}}^*$ and $> = \to_{(\mathcal{S}\setminus\mathcal{S}')/\mathcal{R}}^+$. Since $\geqslant$ and $>$ can never increase by rewriting, it suffices to exploit the first decrease with respect to $>$. Consider a rewrite sequence $v_1 \to_{\mathcal{R}} v_2 \to_{\mathcal{R}} \cdots \to_{\mathcal{R}} v_l$. Take the smallest k such that $v_1 \to v_{k+1} \in \mathcal{S}$ but $v_1 \to v_{k+1} \notin \mathcal{S}'$. Then $v_i \to_{(\geqslant,\geqslant)} v_{i+1}$ for $1 \leqslant i \leqslant k$ and $v_i \to_{(>,>)} v_{i+1}$ for $k < i < l$. If no such k exists set $v_i \to_{(\geqslant,\geqslant)} v_{i+1}$ for $1 \leqslant i < l$. We demonstrate the above idea on an example.

Example 4.64. Consider the following TRS $\mathcal{R}$ from [6]:

$$\mathsf{I}(x) \to \mathsf{I}(\mathsf{J}(x)) \quad \mathsf{J}(x) \to \mathsf{J}(\mathsf{K}(\mathsf{J}(x))) \quad \mathsf{H}(\mathsf{I}(x)) \to \mathsf{K}(\mathsf{J}(x)) \quad \mathsf{J}(x) \to \mathsf{K}(\mathsf{J}(x))$$

We show how the critical peak $\mathsf{H}(\mathsf{I}(\mathsf{J}(x))) \leftarrow \mathsf{H}(\mathsf{I}(x)) \to \mathsf{K}(\mathsf{J}(x))$ can be closed decreasingly $\mathsf{H}(\mathsf{I}(\mathsf{J}(x))) \to_{(\geqslant,\geqslant)} \mathsf{K}(\mathsf{J}(\mathsf{J}(x))) \to_{(>,>)} \mathsf{K}(\mathsf{J}(\mathsf{K}(\mathsf{J}(x)))) \;{}_{(\leqslant,\leqslant)}\!\leftarrow \mathsf{K}(\mathsf{J}(x))$ by $\ell_{\mathrm{rt}}^{\mathcal{S}}$. Let $\mathcal{S}$ be the TRS consisting of the critical diagram steps from the above diagram, i.e.,

$$\begin{aligned} &\mathsf{H}(\mathsf{I}(x)) \to \mathsf{H}(\mathsf{I}(\mathsf{J}(x))) && \mathsf{H}(\mathsf{I}(x)) \to \mathsf{K}(\mathsf{J}(\mathsf{J}(x))) \\ &\mathsf{H}(\mathsf{I}(x)) \to \mathsf{K}(\mathsf{J}(x)) && \mathsf{H}(\mathsf{I}(x)) \to \mathsf{K}(\mathsf{J}(\mathsf{K}(\mathsf{J}(x)))) \end{aligned}$$

The interpretation $\mathsf{H}_{\mathbb{N}}(x) = \mathsf{J}_{\mathbb{N}}(x) = \mathsf{K}_{\mathbb{N}}(x) = x$ and $\mathsf{I}_{\mathbb{N}}(x) = x + 1$ allows to "simplify" termination of the problem $\mathcal{S}/\mathcal{R}$ according to Theorem 2.41. Since the rules that reduce the number of $\mathsf{I}'s$ are dropped from $\mathcal{S}$ (and $\mathcal{R}$), those rules admit a decrease in the labeling.

The abstraction works similarly for the labelings $\ell_\star$ and $\ell_\triangle$ from Lemmata 4.21 and 4.27, respectively.

Finally, we explain why $\overset{*}{\star}(\mathcal{R})$ need not be computed explicitly to implement Corollary 4.23 with the labeling from Lemma 4.25. The idea is to start with $\star(\mathcal{R})$

and incrementally prove termination of $\mathcal{R}^\star_>/\mathcal{R}^\star_=$ until some $\mathcal{S}_1/\mathcal{S}_2$ is reached. If all left-hand sides in $\mathcal{S}_1$ are distinct then they must have been derived from different combinations (l, x) with $l \to r \in \mathcal{R}$ and $x \in \mathcal{V}ar(l)$.[5] Hence they are exactly those rules which should be placed in $\mathcal{R}^\star_=$. We show the idea by means of an example.

Example 4.65. We revisit Example 4.1 and try to prove termination of $\star(\mathcal{R})$. By an application of Theorem 2.41 with the interpretation given in Example 4.26 the problem is termination equivalent to $\mathcal{R}_\dagger/\mathcal{R}^\star_=$ and by another application of Theorem 2.41 the same proof can be used to show termination of $(\mathcal{R}^\star_> \setminus \mathcal{R}^\star_\dagger)/(\mathcal{R}^\star_= \cup \mathcal{R}^\star_\dagger)$ which is a suitable candidate for $\overset{\star}{\star}(\mathcal{R})$ since the rules in $\mathcal{R}^\star_\dagger$ have different left-hand sides.

We have also implemented Theorems 4.39 and 4.41. The requirements of Theorem 4.39 can be checked effectively by the following characterization of $t \in \mathcal{T}_{\trianglelefteq\,\alpha}(\mathcal{F}, \mathcal{V})$:

Remark 4.66. The condition $t \in \mathcal{T}_{\trianglelefteq\,\alpha}(\mathcal{F}, \mathcal{V})$ holds if and only if t is S-sorted and $S(t)\ (\leqslant \cup \lhd_1)^*\ \alpha$, where the relation $\lhd_1$ on sorts relates argument types to result types: $S(f, i) \lhd_1 S(f)$ for all function symbols $f \in \mathcal{F}$ of arity n and $1 \leqslant i \leqslant n$.

We only implemented the simplest case of Theorem 4.41, where ℓ is a rule labeling. First, using Remark 4.66, we determine for which rules $l \to r \in \mathcal{R}$, $l' \to r' \in \mathcal{R}$, it is possible to nest $l' \to r'$ below a duplicating variable of $l \to r$. We add constraints $i(l \to r) > i(l' \to r')$ to our constraint satisfaction problem for the rule labeling. The hard work is done by an SMT solver.

To postpone the expensive computation (and labeling) of parallel critical pairs as long as possible we implemented Theorem 4.54 according the following lazy approach. We first find ordinary weak LL-labelings for the critical diagrams, as described earlier in this section. Only if confluence cannot be established by considering this weak LL-labeling for (non-parallel) critical peaks, we generate parallel critical peaks together with joining sequences. Finally, we check whether the weak LL-labeling joins all resulting diagrams (critical and parallel critical) decreasing as per Theorem 4.54. This check is also responsible for combining single steps into a parallel one for the joining sequence. We confess that this implementation for Theorem 4.54 is somewhat opportunistic but allows to reuse partial progress (the weak LL-labeling) while postponing parallel critical pairs as long as possible.

4.6 Conclusion

In this chapter we studied how the decreasing diagrams technique can be automated. We presented conditions (subsuming recent related results) that ensure confluence of a left-linear TRS whenever its critical peaks are decreasing. The labelings we proposed can be combined lexicographically which allows

[5] When computing $\star(\mathcal{R})$ the implementation renames variables such that (ℓ, x) uniquely identifies a rule $\ell \to r$.

incremental proofs of confluence and has a modular flavor in the following sense: Whenever a new labeling function is invented, the whole framework gains power. We discussed several situations (Examples 4.1, 4.16, 4.18, 4.58) where traditional confluence techniques fail but our approach easily establishes confluence. We have also considered parallel rewriting resulting in a significantly more powerful approach.

Chapter 5

Confluence with Layer Systems

We introduce layer systems for proving generalizations of the modularity of confluence for first-order rewrite systems. Layer systems specify how terms can be divided into layers. We establish structural conditions on those systems that imply confluence. Our abstract framework covers known results like modularity, many-sorted persistence, layer-preservation and currying. We present a counterexample to an extension of persistence to order-sorted rewriting and derive new sufficient conditions for the extension to hold. All our proofs are constructive. The contents of this chapter is based on [19].

5.1 Introduction

We revisit the celebrated modularity result of confluence, due to Toyama [75]. It states that the union of two confluent rewrite systems is confluent, provided the participating rewrite systems do not share function symbols. This result has been reproved several times, using category theory [49], ordered completion [37], and decreasing diagrams [63]. While confluence is also modular for rewriting modulo [37, 36], the situation is different for higher-order rewriting [7]. In practice, modularity is of limited use. More useful techniques, in the sense that rewrite systems can be decomposed into smaller systems that share function symbols and rules, are based on type introduction [4], layer-preservation [54], and commutativity [67].

Type introduction [80] restricts the set of terms that have to be considered to the well-typed terms according to some many-sorted type discipline which is compatible with the rewrite system under consideration. A property of (many-sorted) rewrite systems which is preserved and reflected under type removal is called persistent and Aoto and Toyama [4] showed that confluence is persistent. In [3] they extended the latter result by considering an order-sorted type discipline. However, we show that the conditions imposed in [3] are not sufficient for confluence.

The proofs in [54] and [3, 4] are adaptations of the proof of Toyama's modularity result by [43]. A more complicated proof using concepts from [43] has been given by Kahrs, who showed in [39] that confluence is preserved under currying [41]. In this chapter we introduce *layer systems* as a common framework to capture the results of [4, 39, 54, 75] and to identify appropriate conditions to restore the persistence of confluence for order-sorted rewriting [3]. Layer systems identify the parts that are available when decomposing terms. The key proof idea remains the same. We treat each such layer independently from

the others where possible, and deal with interactions between layers separately. The main advantage of and motivation for our proof is that the result becomes reusable; rather than checking every detail of a complex proof, we have to check a couple of comparatively simple, structural conditions on layer systems instead. Such a common framework also facilitates a formalization of these results in a theorem prover like Isabelle or Coq.

Besides the theoretical results of this chapter we stress practical implications: Due to an implementation of Theorem 5.65 in our confluence tool CSI [78] it supports a decomposition result based on ordered sorts, exceeding the criteria available in other tools. A second result of practical importance is preservation and reflection of confluence under currying [39] which is used as a preprocessing step when deciding confluence of ground TRSs [18].

The remainder of this chapter is organized as follows. In the next section we recall preliminaries. Section 5.2 introduces layer systems and establishes results how rewriting interacts with layers. The main (abstract) results for confluence via layer systems are presented in Section 5.3 and instantiated in Section 5.4 to obtain various known results. The new result on order-sorted persistence is covered in Section 5.5. Differences to related work are discussed in Section 5.6, which might be consulted in advance by readers familiar with the literature. We conclude in Section 5.7.

5.2 Layer Systems

In this section we introduce layer systems, which are sets of contexts satisfying special properties. The top-down decomposition of a term into maximal layers admits the notion of the rank of a term. Since for suitable layer systems rewriting does not increase the rank this is a valid measure for proofs by induction.

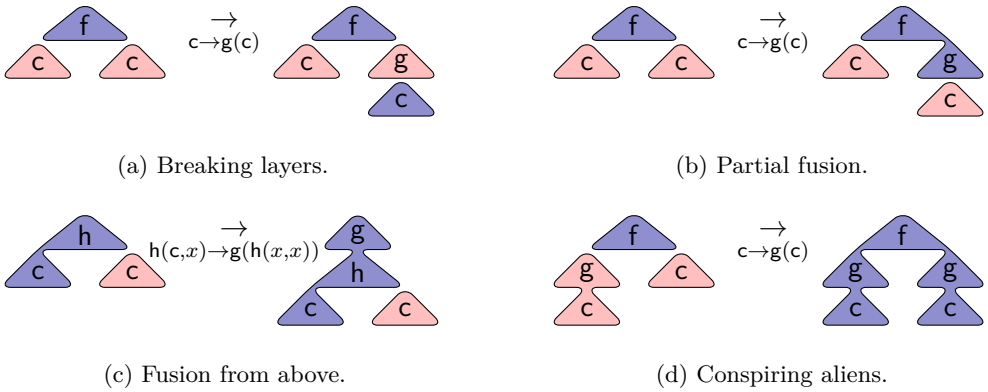
Definition 5.1. Let $\mathbb{L} \subseteq \mathcal{C}(\mathcal{F}, \mathcal{V})$ be a set of contexts. Then $L \in \mathbb{L}$ is called a *top* of a context $C \in \mathcal{C}(\mathcal{F}, \mathcal{V})$ (according to $\mathbb{L}$) if $L \sqsubseteq C$. A top is a *max-top* of C if it is maximal with respect to $\sqsubseteq$ among the tops of C.

Note that terms are contexts without holes, so they have tops and max-tops as well. In the sequel we use subsets $\mathbb{L} \subseteq \mathcal{C}(\mathcal{F}, \mathcal{V})$ to layer terms. The process is top-down, taking the max-top of a term as layer and proceeding recursively.

Example 5.2. Let $\mathcal{F}$ consist of a binary function symbol f, a unary function symbol g, and constants a, b, and c. We consider the following candidates for $\mathbb{L}$:

$$\begin{aligned}
\mathbb{L}_0 &= \varnothing \\
\mathbb{L}_1 &= \{\mathsf{f}(v,w), \mathsf{g}(v), \mathsf{a}, \mathsf{b}, \mathsf{c}, v \mid v, w \in \mathcal{V}_\Box\} \\
\mathbb{L}_2 &= \{\mathsf{f}(\mathsf{g}^n(v), \mathsf{g}^m(w)), \mathsf{g}^n(v), \mathsf{g}^n(\mathsf{c}), \mathsf{a}, \mathsf{b} \mid v, w \in \mathcal{V}_\Box,\ n, m \in \mathbb{N}\} \\
\mathbb{L}_3 &= \{\mathsf{f}(\mathsf{g}^n(v), \mathsf{g}^m(w)), \mathsf{g}^n(v), \mathsf{a}, \mathsf{b} \mid v, w \in \mathcal{V}_\Box \cup \{\mathsf{c}\},\ n, m \in \mathbb{N}\}
\end{aligned}$$

Regard the terms $s = \mathsf{f}(\mathsf{c}, \mathsf{c})$ and $t = \mathsf{f}(\mathsf{c}, \mathsf{g}(\mathsf{c}))$. According to $\mathbb{L}_0$, neither s nor t have any tops. According to $\mathbb{L}_1$, the tops of both s and t are $\Box$ and $\mathsf{f}(\Box, \Box)$, and the latter is the max-top of s and t. According to $\mathbb{L}_2$, $\Box$ and $\mathsf{f}(\Box, \Box)$ are tops of s and t, and $\mathsf{f}(\Box, \mathsf{g}(\Box))$ is a top of t but not of s. The max-tops of s



(a) Breaking layers.

(b) Partial fusion.

(c) Fusion from above.

(d) Conspiring aliens.

Figure 5.1: Undesired behavior on layers.

and t are $\mathsf{f}(\Box, \Box)$ and $\mathsf{f}(\Box, \mathsf{g}(\Box))$, respectively. Finally, the max-tops of s and t according to $\mathbb{L}_3$ are s and t themselves.

Our goal is to deduce confluence of $\mathcal{R}$ when rewriting is restricted to $\mathbb{L} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$. To this end, we need to impose restrictions on $\mathbb{L}$. This leads to the central definition of the chapter.

Definition 5.3. Let $\mathcal{F}$ be a signature. A set $\mathbb{L} \subseteq \mathcal{C}(\mathcal{F}, \mathcal{V})$ of contexts is called a *layer system* if it satisfies properties (L_1), (L_2), and (L_3). The elements of $\mathbb{L}$ are called *layers*. A TRS $\mathcal{R}$ over $\mathcal{F}$ is *weakly layered (according to a layer system $\mathbb{L}$)* if condition (W) is satisfied for each $\ell \to r \in \mathcal{R}$. It is *layered (according to a layer system $\mathbb{L}$)* if conditions (W), (C_1), and (C_2) are satisfied. The conditions are as follows:

(L_1) Each term in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ has a non-empty top.

(L_2) If $x \in \mathcal{V}$ and $C \in \mathcal{C}(\mathcal{F}, \mathcal{V})$ then $C[x]_p \in \mathbb{L}$ if and only if $C[\Box]_p \in \mathbb{L}$.

(L_3) If $L, N \in \mathbb{L}$, $p \in \mathcal{P}os_{\mathcal{F}}(L)$, and $L|_p \sqcup N$ is defined then $L[L|_p \sqcup N]_p \in \mathbb{L}$.

(W) If M is a max-top of s, $p \in \mathcal{P}os_{\mathcal{F}}(M)$, and $s \to_{p,\ell\to r} t$ then $M \to_{p,\ell\to r} L$ for some $L \in \mathbb{L}$.

(C_1) In (W) either L is a max-top of t or $L = \Box$.

(C_2) If $L, N \in \mathbb{L}$ and $L \sqsubseteq N$ then $L[N|_p]_p \in \mathbb{L}$ for any $p \in \mathcal{P}os_{\Box}(L)$.

Example 5.4 (Example 5.2 revisited)**.** Consider the TRS $\mathcal{R}_1$ consisting of the rewrite rules

$$\mathsf{f}(x, x) \to \mathsf{a} \qquad \mathsf{f}(x, \mathsf{g}(x)) \to \mathsf{b} \qquad \mathsf{c} \to \mathsf{g}(\mathsf{c})$$

from [34]. It is non-confluent because $\mathsf{a} \mathrel{\mathcal{R}_1\!\leftarrow} \mathsf{f}(\mathsf{c}, \mathsf{c}) \to_{\mathcal{R}_1} \mathsf{f}(\mathsf{c}, \mathsf{g}(\mathsf{c})) \to_{\mathcal{R}_1} \mathsf{b}$, and a, b are in normal form. However, $\mathcal{R}_1$ is confluent on $\mathbb{L}_0 \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$, $\mathbb{L}_1 \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ and $\mathbb{L}_2 \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ (on the other hand, note that $\mathbb{L}_1 \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ is not closed under rewriting by $\mathcal{R}_1$ because $\mathsf{c} \to_{\mathcal{R}_1} \mathsf{g}(\mathsf{c})$), but $\mathcal{R}_1$ is not confluent on $\mathbb{L}_3 \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$,

because $\mathsf{a}, \mathsf{f}(\mathsf{c},\mathsf{c}), \mathsf{f}(\mathsf{c},\mathsf{g}(\mathsf{c})), \mathsf{b} \in \mathbb{L}_3 \cap \mathcal{T}(\mathcal{F},\mathcal{V})$. Clearly $\mathbb{L}_0$ violates (L_1), and therefore any attempt of proving confluence of $\mathcal{R}_1$ by decomposing terms into a max-top and remaining subterms is doomed to fail. Our basic idea for establishing confluence of a (weakly) layered TRS is to perform rewrite steps on arbitrary terms on the corresponding elements of a layer system in the terms' decomposition, with subterms replaced by variables (this replacement is enabled by (L_2)).

Figure 5.1(a) depicts the rewrite step $\mathsf{f}(\mathsf{c},\mathsf{c}) \to_{\mathcal{R}_1} \mathsf{f}(\mathsf{c},\mathsf{g}(\mathsf{c}))$ with both terms decomposed according to $\mathbb{L}_1$. Note that the c subterm rewrites to $\mathsf{g}(\mathsf{c})$, but the resulting subterm is split into two layers. Note furthermore that $\mathsf{f}(\mathsf{c},\mathsf{g}(\mathsf{c})) \to_{\mathcal{R}_1} \mathsf{b}$, but the corresponding left-hand side $\mathsf{f}(x,\mathsf{g}(x))$ does not match any part of the decomposition of $\mathsf{f}(\mathsf{c},\mathsf{g}(\mathsf{c}))$. Condition (W) (which is violated by $\mathbb{L}_1$) helps ensuring that rewrite steps on terms can be adequately simulated on layers.

Next consider Figure 5.1(b), depicting the rewrite step $\mathsf{f}(\mathsf{c},\mathsf{c}) \to_{\mathcal{R}_1} \mathsf{f}(\mathsf{c},\mathsf{g}(\mathsf{c}))$ with terms decomposed according to $\mathbb{L}_2$. Note that $\mathbb{L}_2$ satisfies (L_1), (L_2) and (W). Nevertheless, the result of the rewrite step $\mathsf{c} \to_{\mathcal{R}_1} \mathsf{g}(\mathsf{c})$ is broken apart: only a part of $\mathsf{g}(\mathsf{c})$ is merged with the max-top of $\mathsf{f}(\mathsf{c},\mathsf{g}(\mathsf{c}))$. Condition ($\mathrm{L}_3$) prevents such partial fusion. We can see that it is violated by $\mathbb{L}_2$: we have $\mathsf{f}(\Box,\mathsf{g}(\Box)) \in \mathbb{L}_2$ and $\mathsf{g}(\mathsf{c}) \in \mathbb{L}_2$, but $\mathsf{f}(\Box,\mathsf{g}(\Box) \sqcup \mathsf{g}(\mathsf{c})) = \mathsf{f}(\Box,\mathsf{g}(\mathsf{c})) \notin \mathbb{L}_2$. Finally, $\mathbb{L}_3$ weakly layers $\mathcal{R}$.

In order to motivate (C_1), we consider the TRS $\mathcal{R}_2$ consisting of the rewrite rules

$$\mathsf{f}(x,x) \to \mathsf{a} \qquad \mathsf{f}(x,\mathsf{g}(x)) \to \mathsf{b} \qquad \mathsf{h}(\mathsf{c},x) \to \mathsf{g}(\mathsf{h}(x,x))$$

which is closely related to $\mathcal{R}_1$; instead of the rewrite step $\mathsf{c} \to_{\mathcal{R}_1} \mathsf{g}(\mathsf{c})$ we have $t_c \to_{\mathcal{R}_2} \mathsf{g}(t_c)$ for $t_c = \mathsf{h}(\mathsf{c},\mathsf{c})$, and therefore $\mathsf{a} \,{}_{\mathcal{R}_2}\!\leftarrow \mathsf{f}(t_c,t_c) \to_{\mathcal{R}_2} \mathsf{f}(t_c,\mathsf{g}(t_c)) \to_{\mathcal{R}_2} \mathsf{b}$. We define a layer system $\mathbb{L}_4$ by

$$\begin{aligned}\mathbb{L}_c &= \{v, \mathsf{h}(v,w), \mathsf{h}(\mathsf{c},v) \mid v,w \in \mathcal{V}_\Box\}\\ \mathbb{L}_4 &= \{\mathsf{f}(\mathsf{g}^n(s),\mathsf{g}^m(t)), \mathsf{g}^n(t), \mathsf{a}, \mathsf{b}, \mathsf{c}, s \mid s,t \in \mathbb{L}_c,\ n,m \in \mathbb{N}\}\end{aligned}$$

It is straightforward to verify that $\mathbb{L}_4$ weakly layers $\mathcal{R}_2$ and that $\mathcal{R}_2$ is confluent on $\mathbb{L}_4 \cap \mathcal{T}(\mathcal{F},\mathcal{V})$. Figure 5.1(c) depicts the rewrite step $t_c \to_{\mathcal{R}_2} \mathsf{g}(t_c)$. It affects the max-top of t_c, but the max-top of the result, $\mathsf{g}(\mathsf{h}(\mathsf{c},\Box))$, is larger than the result of rewriting the max-top $\mathsf{h}(\mathsf{c},\Box)$ of t_c: $\mathsf{h}(\mathsf{c},\Box) \to \mathsf{g}(\mathsf{h}(\Box,\Box))$. In the case of $\mathcal{R}_2$, there are rewrite sequences in which such fusion from above happens infinitely often, and that presents another obstacle to confluence. Condition (C_1) is designed to rule out such fusion from above completely, and indeed the rewrite step $t_c \to_{\mathcal{R}_2} \mathsf{g}(t_c)$ shows that (C_1) is violated by $\mathbb{L}_4$ and $\mathcal{R}_2$.

Finally consider the layer system

$$\mathbb{L}_5 = \{\mathsf{f}(v,w), \mathsf{f}(\mathsf{g}^{n+1}(\mathsf{c}),\mathsf{g}^{m+1}(\mathsf{c})), \mathsf{a}, \mathsf{g}^n(\mathsf{c}), \mathsf{g}^n(v), v \mid v,w \in \mathcal{V}_\Box,\ n,m \in \mathbb{N}\}$$

which weakly layers the TRS $\mathcal{R}_3$ consisting of the rewrite rules

$$\mathsf{f}(x,x) \to \mathsf{a} \qquad \mathsf{c} \to \mathsf{g}(\mathsf{c})$$

and satisfies (C_1). Figure 5.1(d) shows a rewrite step $\mathsf{f}(\mathsf{g}(\mathsf{c}),\mathsf{c}) \to_{\mathcal{R}_3} \mathsf{f}(\mathsf{g}(\mathsf{c}),\mathsf{g}(\mathsf{c}))$. What happens here is that the result of rewriting the subterm $\mathsf{c} \to \mathsf{g}(\mathsf{c})$ fuses with the previous top, $\mathsf{f}(\Box,\Box)$, but only if the unrelated first subterm $\mathsf{g}(\mathsf{c})$ fuses at the same time. This phenomenon causes problems in our proof, and (C_2) prevents that. To wit, we have $\mathsf{f}(\Box,\Box) \in \mathbb{L}_5$ and $\mathsf{f}(\mathsf{g}(\mathsf{c}),\mathsf{g}(\mathsf{c})) \in \mathbb{L}_5$, so by ($C_2$) with $p = 1$, there should be $\mathsf{f}(\Box \sqcup \mathsf{g}(\mathsf{c}),\Box) \in \mathbb{L}_5$, but this is not the case.

The following convention helps to differentiate various contexts.

Convention 5.5. We use C and D to denote *contexts*, B to denote *base contexts* (to be introduced in Section 5.3), L and N to denote arbitrary *layers*, and M to denote a max-top of a term or context.

In the sequel we implicitly assume a given layer system $\mathbb{L}$. In light of the next lemma we speak of *the* max-top of a term or context.

Lemma 5.6. *Any non-empty context has a unique and non-empty max-top.*

Proof. Let C be a non-empty context. To show that C has a non-empty top let x be a variable not occurring in C and consider $C[x,\dots,x]$, which has a non-empty top L_x by (L_1). Then $L := L_x\sigma$ with $\mathrm{dom}(\sigma) = \{x\}$ and $\sigma(x) = \Box$ is a top of C since $L \in \mathbb{L}$ by (L_2) and $L \sqsubseteq C$ by construction. It is non-empty since $L = \Box$ implies $L_x = x$, hence $C[x,\dots,x] = x$ and consequently $C = \Box$ because x is fresh, contradicting the premises. Hence the set S of non-empty tops of C is non-empty. Since it also is finite it has a (non-empty) maximal element.

To show uniqueness let M and M' be max-tops of C. Then $M \sqsubseteq C$ and $M' \sqsubseteq C$ ensures that $M \sqcup M'$ is defined, and a layer by (L_3) (if $\Box \in \{M, M'\}$ then (L_3) is not needed). If $M \neq M'$ then $M \sqsubset M \sqcup M'$ or $M' \sqsubset M \sqcup M'$. Since $M \sqcup M' \sqsubseteq C$ this gives the desired contradiction. □

Next we introduce the key notion of the rank of a term.

Definition 5.7. Let $t = M[t_1,\dots,t_n]$ with M the max-top of t. Then $t_1,\dots,t_n$ are the *aliens* of t. We define $\mathrm{rank}(t) = 1 + \max\{\mathrm{rank}(t_i) \mid 1 \leqslant i \leqslant n\}$, where $\max(\varnothing) = 0$ by convention.

Since the max-top of a term is uniquely defined (Lemma 5.6), it follows that also its aliens are uniquely defined. The next example shows that rewriting might increase the rank of a term. In Lemma 5.13 we show that this cannot happen in weakly layered TRSs.

Example 5.8. Consider the layer system

$$\mathbb{L}_6 = \{v, \mathsf{f}(v), \mathsf{g}(v), \mathsf{h}(v), \mathsf{f}(\mathsf{g}(\mathsf{h}(v))), \mathsf{g}(\mathsf{g}(v)), \mathsf{a} \mid v \in \mathcal{V}_\Box\}$$

Note that (in contrast to modularity) subterms can have larger rank. E.g., if $s = \mathsf{f}(\mathsf{g}(\mathsf{h}(x)))$ and $t = \mathsf{g}(\mathsf{h}(x))$ then $\mathrm{rank}(s) = 1 < 2 = \mathrm{rank}(t)$. Furthermore $s \to_{\mathcal{R}} t$ in the TRS $\mathcal{R}$ containing the rule $\mathsf{f}(\mathsf{g}(x)) \to \mathsf{g}(x)$. Note that $\mathcal{R}$ is not weakly layered according to $\mathbb{L}_6$.

The next lemma states a useful decomposition result.

Lemma 5.9. *Let $t = L[t_1, \dots, t_n]$, L a top of t, and k be the maximum of* $\mathrm{rank}(t_i)$ *for $1 \leqslant i \leqslant n$. Then* $\mathrm{rank}(t) \leqslant k + 1$ *and aliens of t that are not rooted at hole positions of L have rank less than k.*

Proof. Let M be the max-top of t. We show the (stronger) property for any context C with $C \sqsubseteq M$ (instead of a top L of t). Note that $L \sqsubseteq M$. The proof is by induction on $|t| - |C|_{\mathcal{F}\cup\mathcal{V}}$, which is a natural number because $C \sqsubseteq t$. If $C = M$ then $\mathrm{rank}(t) = 1 + \max\{\mathrm{rank}(t_i) \mid 1 \leqslant i \leqslant n\} = 1 + k$ and all aliens of t are rooted at hole positions of C, so we are done. Otherwise, let M_i be the max-top of t_i. There is a unique maximal context C' such that $C' \sqsubseteq C[M_1, \dots, M_n]$ and $C' \sqsubseteq M$. Furthermore, we have $C \sqsubset C'$ because the M_i are non-empty by Lemma 5.6. Because $C' \sqsubseteq M \sqsubseteq t$, $t = C'[t'_1, \dots, t'_m]$ where t'_j is the subterm of t at the position of the j-th hole in C'. For each $p \in \mathcal{P}os_{\Box}(C)$ there are three possibilities. Let $C[t_1, \dots, t_n]|_p = t_i$.

1. If $p \in \mathcal{P}os_{\Box}(C')$ then $C'[t'_1, \dots, t'_m]|_p = t'_j$ and $t_i = t'_j$ for some j.
2. If $p \in \mathcal{P}os_{\mathcal{V}}(C')$ then there are no holes below p in C'.
3. If $p \in \mathcal{P}os_{\mathcal{F}}(C')$ then $p \in \mathcal{P}os_{\mathcal{F}}(M)$ and $M[M|_p \sqcup M_i]_p \in \mathbb{L}$ by (L_3). Because M is the max-top of t this implies $M_i \sqsubseteq M|_p$ and therefore $C'|_p = M_i$ by construction of C'. Hence all t'_j corresponding to holes of C' below p are aliens of t_i having rank less than $\mathrm{rank}(t_i)$.

We can now apply the induction hypothesis to $C'[t'_1, \dots, t'_m]$ since $C \sqsubset C'$ implies $|t| - |C|_{\mathcal{F}\cup\mathcal{V}} > |t| - |C'|_{\mathcal{F}\cup\mathcal{V}}$. To conclude, note that any alien rooted at a hole position of C' but not at a hole position of C equals a t'_j from case (3) and therefore has rank less than k. □

Lemma 5.10. *Let $\mathcal{R}$ be a TRS that is weakly layered according to $\mathbb{L}$. Then $\mathbb{L}$ is closed under rewriting by $\mathcal{R}$.*

Proof. Let $L \in \mathbb{L}$ and $L \to_{\mathcal{R}} N$. Obviously $L[x, \dots, x] \to_{\mathcal{R}} N[x, \dots, x]$ for a fresh variable x. Since $L[x, \dots, x] \in \mathbb{L}$ by (L_2) it is its own max-top. We conclude since $N[x, \dots, x] \in \mathbb{L}$ by (W) and hence $N \in \mathbb{L}$ by (L_2). □

We now present technical results about rewriting contexts. In the sequel we often want to replace variables affected by some substitution σ by holes. We therefore denote by $\sigma_{\Box}(x)$ the substitution obtained by letting $\sigma_{\Box}(x) = \Box$ for $x \in \mathrm{dom}(\sigma)$ and $\sigma_{\Box}(x) = x$ otherwise. For a context C we denote by $C_{\Box}$ the context obtained from C by replacing all variables by holes.

Lemma 5.11. *Let C be a context and ℓ a non-variable term. If $\ell \leqslant C|_p$ then there is a term c such that*

1. *$\ell \leqslant c|_p$ and $C = c\sigma_{\Box}$ for some substitution σ, and*
2. *if $C \sqsubseteq D$ for a context D and $\ell \leqslant D|_p$ then $c \leqslant D$.*

Proof. Assume that C has $n \geqslant 0$ holes. We may assume without loss of generality that C and ℓ have no variables in common. Let $c_0 := C[x_1, \ldots, x_n]$ with fresh variables $x_1, \ldots, x_n$. The context C witnesses the fact that c_0 and $c_1 := c_0[\ell]_p$ are unifiable. Let c be a most general instance of c_0 and c_1. Note that variables in c can be renamed freely. If $C \sqsubseteq D$ then D is an instance of c_0. Furthermore, if $\ell \leqslant D|_p$ then D must be an instance of c_1 as well and therefore $c \leqslant D$. In particular, $c \leqslant C$ and thus $C = c\sigma$ for some substitution σ. Let τ be a substitution such that $c = c_0\tau$. For $x \in \mathcal{V}ar(C)$, $\sigma(\tau(x)) = x$, which implies that $\tau(x)$ is a variable. We can rename each $\tau(x)$ to x in c. Therefore we may assume without loss of generality that $\sigma(x) = \tau(x) = x$ for $x \in \mathcal{V}ar(C)$. For the variables x_i, we have $\sigma(\tau(x_i)) = \Box$ for all $1 \leqslant i \leqslant n$, which is only possible if σ maps those variables to $\Box$. Consequently, $\sigma_\Box = \sigma$. □

If a rewrite rule is applied to a context then each hole may be erased, copied or duplicated. The same holds for the terms used to fill the holes in a context, as formalized by the next lemma.

Lemma 5.12. *If* $C \to_{p,\ell\to r} C'$ *and* $\ell \leqslant C[s_1, \ldots, s_n]|_p$ *then* $C[s_1, \ldots, s_n] \to_{p,\ell\to r} C'[t_1, \ldots, t_m]$ *and* $\{t_1, \ldots, t_m\} \subseteq \{s_1, \ldots, s_n\}$.

Proof. Since $\ell \leqslant C|_p$, Lemma 5.11(1) yields a term c and a substitution $\sigma_\Box$ such that $\ell \leqslant c|_p$ and $C = c\sigma_\Box$. Furthermore due to $C \sqsubseteq C[s_1, \ldots, s_n]$ and $\ell \leqslant C[s_1, \ldots, s_n]|_p$, there is a substitution σ with $c\sigma = C[s_1, \ldots, s_n]$ by Lemma 5.11(2). Hence $C \to_{p,\ell\to r} C'$ mirrors a rewrite step $c \to_{p,\ell\to r} c'$ with $C' = c'\sigma_\Box$ and $C'[t_1, \ldots, t_m] = c'\sigma$. Since $t_1, \ldots, t_m$ can only come from σ we conclude. □

This section ends with a key lemma that enables the use of induction on the rank of terms for proving confluence of $\mathcal{R}$.

Lemma 5.13. *Let* $\mathcal{R}$ *be a weakly layered TRS. If* $s \to_\mathcal{R} t$ *then* $\mathrm{rank}(s) \geqslant \mathrm{rank}(t)$.

Proof. By induction on the rank of s. Let $s \to_p t$ and $s = M[s_1, \ldots, s_n]$ be the decomposition of s into max-top and aliens. We distinguish two cases.

If $p \in \mathcal{P}os_\mathcal{F}(M)$ then condition (W) yields $M \to_p L$ and L a top of t. Let $t = L[t_1, \ldots, t_m]$. By Lemma 5.12 $\{t_1, \ldots, t_m\} \subseteq \{s_1, \ldots, s_n\}$ since $M \to_p L$. Hence $\mathrm{rank}(t) \leqslant 1 + \max\{\mathrm{rank}(t_i) \mid 1 \leqslant i \leqslant m\} \leqslant 1 + \max\{\mathrm{rank}(s_i) \mid 1 \leqslant i \leqslant n\} = \mathrm{rank}(s)$ using Lemma 5.9.

If $p \notin \mathcal{P}os_\mathcal{F}(M)$ then $s_j \to s'_j$ and $t = M[s_1, \ldots, s'_j, \ldots, s_n]$ for some $1 \leqslant j \leqslant n$. The induction hypothesis yields $\mathrm{rank}(s_j) \geqslant \mathrm{rank}(s'_j)$. Since M is a top of t, Lemma 5.9 yields $\mathrm{rank}(t) \leqslant 1 + \max\{\mathrm{rank}(s'_j), \mathrm{rank}(s_i) \mid 1 \leqslant i \leqslant n, i \neq j\} \leqslant 1 + \max\{\mathrm{rank}(s_i) \mid 1 \leqslant i \leqslant n\} = \mathrm{rank}(s)$. □

5.3 Confluence by Layer Systems

We start this long section by stating our main results. All results reduce the task of proving confluence of a TRS to the easier task of proving confluence of the terms in a suitable layer system, i.e., the terms in $\mathbb{L} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$, which are precisely the terms of rank one. The first result imposes left-linearity.

Theorem 5.14. *Let $\mathcal{R}$ be a weakly layered TRS that is confluent on terms of rank one. If $\mathcal{R}$ is left-linear then $\mathcal{R}$ is confluent.*

The second result exchanges left-linearity for a condition that is weaker than non-duplication.

Definition 5.15. Let $\mathcal{R}$ be a TRS and $\Diamond$ a fresh unary function symbol. Then $\mathcal{R}$ is *bounded duplicating* if the relative rewrite system $\{\Diamond(x) \to x\}/\mathcal{R}$ is terminating.

Theorem 5.16. *Let $\mathcal{R}$ be a weakly layered TRS that is confluent on terms of rank one. If $\mathcal{R}$ is bounded duplicating then $\mathcal{R}$ is confluent.*

Lemma 5.17. *Non-duplicating TRSs are bounded duplicating.*

Proof. Let $\mathcal{R}$ be a non-duplicating TRS. To show termination of $\{\Diamond(x) \to x\}/\mathcal{R}$ we measure terms by counting the number of occurrences of the $\Diamond$ symbol. Clearly each application of the $\Diamond(x) \to x$ rule decreases that number and rules of $\mathcal{R}$ do not increase it because they do not duplicate $\Diamond$ symbols and cannot introduce any new ones. □

Bounded duplication strictly extends non-duplication; the TRS consisting of the rewrite rule $\mathsf{f}(x, x) \to \mathsf{g}(x, x, x)$ is duplicating but still bounded duplicating. This can be shown by the polynomial interpretation [48] given by

$$\mathsf{f}_{\mathbb{N}}(x, y) = 2x + 2y \qquad \mathsf{g}_{\mathbb{N}}(x, y, z) = x + y + z \qquad \Diamond_{\mathbb{N}}(x) = x + 1$$

By combining Theorem 5.16 with Lemma 5.17, we obtain the following corollary.

Corollary 5.18. *Let $\mathcal{R}$ be a weakly layered TRS that is confluent on terms of rank one. If $\mathcal{R}$ is non-duplicating then $\mathcal{R}$ is confluent.* □

The third result does not impose any conditions on $\mathcal{R}$ but further limits the layer systems that can be employed to derive confluence.

Theorem 5.19. *Let $\mathcal{R}$ be a layered TRS that is confluent on terms of rank one. Then $\mathcal{R}$ is confluent.*

Hence for duplicating TRSs there are three possibilities to prove confluence, either by weakly layering a left-linear rewrite system (Theorem 5.14), by establishing bounded duplication for a weakly layered rewrite system (Theorem 5.16), or by layering the rewrite system (Theorem 5.19). Table 5.1 shows that the three results are pairwise incomparable where $\mathbb{L} = \{v, \mathsf{k}(v, w), \mathsf{b} \mid v, w \in \mathcal{V}_\Box\}$ and $\mathbb{L}_6$ is as in Example 5.8.

The following subsections are devoted to the proof development for Theorems 5.14, 5.16, and 5.19. In Section 5.3.1 we describe the proof setup and introduce auxiliary rewrite relations. In Sections 5.3.2 and 5.3.3 we show that the auxiliary relations are locally decreasing. Finally, we wrap up the proofs in Section 5.3.4.

rewrite rule	layer system	Thm. 5.14	Thm. 5.16	Thm. 5.19
$\mathsf{f}(\mathsf{g}(\mathsf{h}(x))) \to \mathsf{g}(x)$	$\mathbb{L}_6$	✓	✓	×
$\mathsf{k}(\mathsf{b}, x) \to \mathsf{k}(x, x)$	$\mathbb{L}$	✓	×	✓
$\mathsf{k}(x, x) \to \mathsf{k}(x, x)$	$\mathbb{L}$	×	✓	✓

Table 5.1: Incomparability of the main results.

5.3.1 Proof Setup

Assume we are given a weakly layered TRS $\mathcal{R}$ such that $\mathcal{R}$ is confluent on terms of rank one. We will show confluence of $\mathcal{R}$ on all terms by induction on the rank of terms. In the sequel we prepare for the induction step, hence:

We fix r and assume terms with rank at most r to be confluent.

Next we generalize the crucial concepts of [63] from the modularity setting to layer systems. We have renamed *non-native* to *foreign* because the former is confusing.

Definition 5.20. Terms with rank at most $r + 1$ are called *native*. An alien of a native term is *tall* if its rank equals r and *short* otherwise. *Foreign* terms have rank less than or equal to r.

Note that by definition, foreign terms are also native. However, we will only call terms foreign if they are descendants of aliens.

Definition 5.21. Let t be a native term. Its *base context* B is obtained by replacing all tall aliens in t with holes. The tall aliens form the *base sequence* $\mathbf{t}$, which satisfies $t = B[\mathbf{t}]$.

Definition 5.22. Sequences of foreign terms are called *foreign sequences*. The *imbalance* of a foreign sequence $\mathbf{t}$ is the number of distinct terms in $\mathbf{t}$. The imbalance of a native term t is the imbalance of its base sequence. If $\mathbf{s}$ and $\mathbf{t}$ are sequences of length n, then we write $\mathbf{s} \propto \mathbf{t}$ if $s_i = s_j$ implies $t_i = t_j$ for all $1 \leqslant i, j \leqslant n$.

Note that the relation $\propto$ is transitive. It is useful for analyzing the imbalance of foreign sequences. If $\mathbf{s} \propto \mathbf{t}$ then the imbalance of $\mathbf{t}$ is no larger than that of $\mathbf{s}$.

Definition 5.23. Let s and t be native terms. A *short step* $s \blacktriangleright\!\!\blacktriangleright_{s_0} t$ is a sequence of $\mathcal{R}$-steps $s \to_{\mathcal{R}}^* t$ that is mirrored by a rewrite sequence $B \to_{\mathcal{R}}^* C$ from the base context B of s. Short steps are labeled by terms s_0 that are predecessors of the source: $s_0 \to_{\mathcal{R}}^* s$. We omit the label when it is irrelevant.

There are two ways in which short steps arise: either by rewriting short aliens (hence the name), or by rewriting the max-top of a term. In the sequel we will sometimes use the fact that in Definition 5.23, $C \sqsubseteq t$ by Lemma 5.12, and when writing $s = B[\mathbf{s}]$ and $t = C[\mathbf{t}]$, each element of $\mathbf{t}$ is an element of $\mathbf{s}$.

term	foreign	native	max-top	base context	base sequence	imbalance
$\mathsf{f}(\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a}))$	×	✓	$\mathsf{f}(\Box,\Box)$	$\mathsf{f}(\Box,\Box)$	$(\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a}))$	1
$\mathsf{f}(\mathsf{H}(\mathsf{a}),\mathsf{G}(\mathsf{a}))$	×	✓	$\mathsf{f}(\Box,\Box)$	$\mathsf{f}(\Box,\Box)$	$(\mathsf{H}(\mathsf{a}),\mathsf{G}(\mathsf{a}))$	2
$\mathsf{f}(\mathsf{J},\mathsf{G}(\mathsf{a}))$	×	✓	$\mathsf{f}(\Box,\Box)$	$\mathsf{f}(\mathsf{J},\Box)$	$(\mathsf{G}(\mathsf{a}))$	1
$\mathsf{f}(\mathsf{K},\mathsf{K})$	✓	✓	$\mathsf{f}(\Box,\Box)$	$\mathsf{f}(\mathsf{K},\mathsf{K})$	$()$	0

Table 5.2: Properties for $r = 2$.

Definition 5.24. Let B and $\mathbf{s}$ be the base context and base sequence of a native term s. If $\mathbf{s} \to_{\mathcal{R}}^{*} \mathbf{t}$ then $s = B[\mathbf{s}] \rhd\!\rhd_{\iota} B[\mathbf{t}] = t$ is a *tall step*. Here the label ι is the imbalance of $\mathbf{t}$.

Note that $\mathbf{t}$ in Definition 5.24 is a foreign sequence because $\mathcal{R}$ is weakly layered. Further note that the imbalance of t may be smaller than ι (since B need not be the base context of t). The following example illustrates the above concepts.

Example 5.25. Consider the TRSs $\mathcal{R}_1 = \{\mathsf{f}(x,x) \to x\}$ over $\mathcal{F}_1 = \{\mathsf{f},\mathsf{a}\}$ and $\mathcal{R}_2 = \{\mathsf{G}(x) \to \mathsf{I}, \mathsf{I} \to \mathsf{K}, \mathsf{G}(x) \to \mathsf{H}(x), \mathsf{H}(x) \to \mathsf{J}, \mathsf{J} \to \mathsf{K}\}$ over $\mathcal{F}_2 = \{\mathsf{I},\mathsf{J},\mathsf{K},\mathsf{G},\mathsf{H}\}$ and let $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$. Then $\mathbb{L} = \mathcal{C}(\mathcal{F}_1,\mathcal{V}) \cup \mathcal{C}(\mathcal{F}_2,\mathcal{V})$ layers $\mathcal{R}_1$ and $\mathcal{R}_2$ (cf. the proof of Theorem 5.50). Assume that $r = 2$. Table 5.2 demonstrates some properties and notions. We have $\mathsf{f}[\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a})] \blacktriangleright\!\blacktriangleright \mathsf{G}(\mathsf{a})$ but $\mathsf{f}[\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a})] \blacktriangleright\!\blacktriangleright \mathsf{I}$ is not possible since the step $\mathsf{G}(\mathsf{a}) \to_{\mathcal{R}} \mathsf{I}$ is not in the base context of $\mathsf{f}(\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a}))$. We also have $\mathsf{f}[\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a})] \rhd\!\rhd_2 \mathsf{f}[\mathsf{J},\mathsf{G}(\mathsf{a})] = u$, despite the imbalance of u being 1 (note that $\mathsf{f}(\Box,\Box)$ is not the base context of u). Furthermore, $(\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a})) \not\propto (\mathsf{J},\mathsf{G}(\mathsf{a}))$ but as the latter can be further rewritten $(\mathsf{J},\mathsf{G}(\mathsf{a})) \to_{\mathcal{R}}^{*} (\mathsf{J},\mathsf{J})$ we obtain $(\mathsf{G}(\mathsf{a}),\mathsf{G}(\mathsf{a})) \propto (\mathsf{J},\mathsf{J})$.

Remark 5.26. The constraint on short steps is subtle. It implies that the rewrite steps do not overlap with any descendants of the tall aliens of s, but furthermore it also has the effect of delaying fusion of those tall aliens with the base context until the end of the rewrite sequence, continuing to treat terms that started out as aliens as aliens even if they could fuse with the top.

We prove confluence of $\mathcal{R}$ on native terms by showing that any local peak consisting of short steps and/or tall steps may be joined decreasingly. Steps are compared as follows. Tall steps are ordered by their imbalance, tall steps are ordered above short steps, and short steps are compared by a well-founded order introduced later (in the proof of Lemma 5.46).

In the remainder of this section we use s, t, and u to denote native terms.

5.3.2 Local Decreasingness of Peaks involving Tall Steps

Based on Lemma 5.12 we obtain the following result:

Lemma 5.27. *Let* $\mathbf{s}$ *and* $\mathbf{t}$ *be sequences of contexts with* $\mathbf{s} \propto \mathbf{t}$ *and* $C \to_{p,\ell\to r} C'$. *If* $\ell \leqslant C[\mathbf{s}]|_p$ *then* $C[\mathbf{t}] \to_{p,\ell\to r} C'[\mathbf{t}']$ *with each element of* $\mathbf{t}'$ *belonging to* $\mathbf{t}$.

Proof. We extend the proof of Lemma 5.12 as follows. Let τ be the substitution

$$\tau(x) = \begin{cases} t_i & \text{if } x \in \mathrm{dom}(\sigma_\Box) \text{ and } \sigma(x) = s_i \\ x & \text{otherwise} \end{cases}$$

Note that $C[\mathbf{t}] = c\tau$ because $\mathbf{s} \propto \mathbf{t}$. We have $c\tau \to_{p,\ell\to r} c'\tau$. Comparing $c'\tau$ and $C' = c'\sigma_\Box$ establishes the claim that $c'\tau = C'[\mathbf{t}']$ with each element of $\mathbf{t}'$ equaling some element of $\mathbf{t}$. □

Lemma 5.28. *Let* $\mathbf{s}$, $\mathbf{t}$, $\mathbf{u}$ *be foreign sequences. If* $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{t}$ *and* $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{u}$ *then there is a foreign sequence* $\mathbf{v}$ *such that* $\mathbf{t} \to_{\mathcal{R}}^* \mathbf{v}$, $\mathbf{u} \to_{\mathcal{R}}^* \mathbf{v}$ *with* $\mathbf{t} \propto \mathbf{v}$ *and* $\mathbf{u} \propto \mathbf{v}$.

Proof. Let m be the length of $\mathbf{s}$. We use induction on the number of disequalities $t_i \neq u_i$ for $1 \leqslant i \leqslant m$. If this number is zero then $\mathbf{t} = \mathbf{u}$ and we can take $\mathbf{v} = \mathbf{t}$. Otherwise, $t_i \neq u_i$ for some $1 \leqslant i \leqslant m$. Both t_i and u_i are reducts of s_i and thus have a common reduct v since $\mathcal{R}$ is confluent on foreign terms. By replacing every occurrence of t_i and u_i in $\mathbf{t}$, $\mathbf{u}$ by v, we obtain new sequences $\mathbf{t}'$, $\mathbf{u}'$ that satisfy $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{t} \to_{\mathcal{R}}^* \mathbf{t}'$, $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{u} \to_{\mathcal{R}}^* \mathbf{u}'$, $\mathbf{t} \propto \mathbf{t}'$ and $\mathbf{u} \propto \mathbf{u}'$. Since the number of disequalities $t'_i \neq u'_i$ is decreased, we conclude by the induction hypothesis and the transitivity of $\propto$. □

A step in the base context is short.

Lemma 5.29. *Let* p *be a non-hole position of the base context of* s. *If* $s \to_p t$ *then* $s \blacktriangleright\!\blacktriangleright t$.

Proof. Let B be the base context of s and let $s \to_p t$. We show $B \to_p C$ for some context C. Because left-hand sides of rules are not variables, $p \in \mathcal{P}os_{\mathcal{F}}(B)$. Let M be the max-top of s, which is also the max-top of B. We distinguish two cases. If $p \in \mathcal{P}os_{\mathcal{F}}(M)$ then consider the decomposition $s = M[\mathbf{s}]$. According to (W) there is a layer L with $M \to_p L$. We have $B = M[\mathbf{s}']$ where $s'_i = s_i$ if s_i is a short alien and $s'_i = \Box$ if s_i is tall. Clearly $\mathbf{s} \propto \mathbf{s}'$ and hence we conclude by Lemma 5.27. If $p \notin \mathcal{P}os_{\mathcal{F}}(M)$ then $s|_p$ is a subterm of a short alien of s and thus $B|_p = s|_p$. Hence $B \to_p C$ for the context $C := B[t|_p]_p$. □

When doing a short step $s = B[\mathbf{s}] \blacktriangleright\!\blacktriangleright C[\mathbf{s}'] = t$, in general the context C is not the base context of t (because of fusion from above or conspiring aliens). Similarly, for a tall step $s = B[\mathbf{s}] \rhd\!\rhd B[\mathbf{t}] = t$ in general the context B is not the base context of t (because of fusion caused by steps in the aliens of t), but both contexts (B and C) satisfy the more general property defined below.

Definition 5.30. We call a context *shallow* if its rank is at most r and all its aliens are terms from $\mathcal{T}(\mathcal{F}, \mathcal{V})$.

Note that the base contexts of native terms are shallow. The same holds for the max-tops of native terms. Furthermore, shallow contexts are closed under rewriting, as shown by the next lemma.

Lemma 5.31. *If* C *is a shallow context and* $C \to_{\mathcal{R}} D$ *then* D *is a shallow context.*

Proof. Assume that $C \to_{p,\ell\to r} D$. Then $C[x,\dots,x] \to_{p,\ell\to r} D[x,\dots,x]$ for a fresh variable x. Let M_x be the max-top of $C[x,\dots,x]$ and note that the max-top M of C is obtained by replacing each occurrence of x by a hole in M_x. If $p \in \mathcal{P}os_{\mathcal{F}}(M) = \mathcal{P}os_{\mathcal{F}}(M_x)$ then by (W) there is a rewrite step $M_x \to_{p,\ell\to r} L_x$ where L_x is a layer, and even a top of $D[x,\dots,x]$ by Lemma 5.12. There is a mirroring rewrite step $M \to_{p,\ell\to r} L$ where L is a top of D. By Lemma 5.12, each hole of L corresponds to a hole or a term without holes in D. If $p \notin \mathcal{P}os_{\mathcal{F}}(M)$ then we take $L = M$, which is a top of D. Again, each hole of L corresponds to a hole or a term in D. In both cases we conclude by noting that any holes of D are holes of L and therefore also of the max-top of D and that the rank of D, which equals the rank of D_x, is at most r by Lemma 5.13. □

Let $s = B[\mathbf{s}]$ be the decomposition of s into base context and base sequence. From the previous result we get that $B[\mathbf{s}] \blacktriangleright\!\blacktriangleright C[\mathbf{s}'] = t$ (with $B \to_{\mathcal{R}}^* C$) implies that C is shallow. The next result establishes that the shallow context C is never larger than the base context of t.

Lemma 5.32. *Let C be a shallow context and t a native term. If $C \sqsubseteq t$ then $C \sqsubseteq B$ for the base context B of t.*

Proof. Let $C = M[\mathbf{s}]$ be the decomposition of C into max-top and aliens. Since C is shallow, elements of $\mathbf{s}$ are either holes or terms of rank less than r. From $M \sqsubseteq C \sqsubseteq t$ we infer the existence of a sequence $\mathbf{t}'$ such that $t = M[\mathbf{t}']$ and $s_i = t'_i$ whenever $s_i \neq \Box$. By Lemma 5.9 every tall alien in t is a subterm of a term of rank at least r in $\mathbf{t}'$. Hence $C \sqsubseteq B$ as desired. □

Steps within shallow contexts are short steps.

Lemma 5.33. *Let p be a non-hole position in a shallow context C with $s = C[\mathbf{s}]$. If $s \to_p t$ then $s \blacktriangleright\!\blacktriangleright t$.*

Proof. By Lemmata 5.32 and 5.29. □

Steps below a shallow context can be decomposed into tall and short steps.

Lemma 5.34 (tall–short factorization)**.** *Let $s = C[\mathbf{s}]$ with a shallow context C and a foreign sequence $\mathbf{s}$. If $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{t}$ and ι is the imbalance of $\mathbf{t}$ then $C[\mathbf{s}] \rhd\!\rhd_{\leqslant\iota} \cdot \blacktriangleright\!\blacktriangleright^* C[\mathbf{t}]$.*

Proof. Let B and $\mathbf{s}'$ be the base context and base sequence of s. Note that by Lemma 5.9 (with L equal to the max-top of C) the tall aliens $\mathbf{s}'$ of s are a subsequence of $\mathbf{s}$, because all aliens of C have rank less than r. For the corresponding subsequence $\mathbf{t}'$ of $\mathbf{t}$, we obtain $s = B[\mathbf{s}'] \rhd\!\rhd_{\leqslant\iota} B[\mathbf{t}']$, while the remaining elements of $\mathbf{s}$ and $\mathbf{t}$ give rise to a rewrite sequence $B = C[\mathbf{s}''] \to_{\mathcal{R}}^* C[\mathbf{t}'']$, where $\mathbf{s}''$ ($\mathbf{t}''$) is obtained by replacing the terms corresponding to the elements of $\mathbf{s}'$ ($\mathbf{t}'$) by holes. Consequently, $B[\mathbf{t}'] = C[\mathbf{s}''][\mathbf{t}'] \blacktriangleright\!\blacktriangleright^* C[\mathbf{t}''][\mathbf{t}'] = C[\mathbf{t}]$ by Lemma 5.33. □

Example 5.35. Continuing Example 5.25. Let $s = \mathsf{f}(\mathsf{J}, \mathsf{G}(\mathsf{a}))$. Then $s = C[\mathbf{s}]$ for the shallow context $C = \mathsf{f}(\Box, \Box)$ with $\mathbf{s} = (\mathsf{J}, \mathsf{G}(\mathsf{a}))$. Let $\mathbf{t} = (\mathsf{K}, \mathsf{I})$. Since $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{t}$ the conditions of Lemma 5.34 hold and we have $C[\mathbf{s}] \rhd\rhd_{\leqslant 2} \cdot \blacktriangleright\blacktriangleright^* C[\mathbf{t}]$. The tall step arises as $s = \mathsf{f}(\mathsf{J}, \Box)[\mathsf{G}(\mathsf{a})] \rhd\rhd_1 \mathsf{f}(\mathsf{J}, \Box)[\mathsf{I}] = \mathsf{f}(\mathsf{J}, \mathsf{I})$ while $\mathsf{f}(\mathsf{J}, \mathsf{I}) \blacktriangleright\blacktriangleright \mathsf{f}(\mathsf{K}, \mathsf{I})$ is a short step since $\mathsf{f}(\mathsf{J}, \mathsf{I})$ is its own base context.

Lemma 5.36. *Peaks of tall steps are locally decreasing:*

$$ {}_{\iota}\lhd\rhd \cdot \rhd\rhd_{\kappa} \;\subseteq\; \rhd\rhd_{\leqslant\kappa} \cdot \blacktriangleright\blacktriangleright^* \cdot {}^*\blacktriangleleft\blacktriangleleft \cdot {}_{\leqslant\iota}\lhd\rhd $$

Proof. Let $t\ {}_{\iota}\lhd\rhd\ s \rhd\rhd_{\kappa} u$ and let the base context and base sequence of s be B and $\mathbf{s}$. There are foreign sequences $\mathbf{t}$ and $\mathbf{u}$ such that $\mathbf{t}\ {}_{\mathcal{R}}^*\!\leftarrow \mathbf{s} \to_{\mathcal{R}}^* \mathbf{u}$ and $t = B[\mathbf{t}]$, $u = B[\mathbf{u}]$. By Lemma 5.28, we can find a foreign sequence $\mathbf{v}$ such that $\mathbf{t} \to_{\mathcal{R}}^* \mathbf{v}\ {}_{\mathcal{R}}^*\!\leftarrow \mathbf{u}$, $\mathbf{t} \propto \mathbf{v}$, and $\mathbf{u} \propto \mathbf{v}$. Hence the imbalance of $\mathbf{v}$ is less than or equal to both ι and κ and we conclude by Lemma 5.34. □

Example 5.37. To demonstrate Lemma 5.36, we extend Example 5.25. Let $s = \mathsf{f}(\mathsf{G}(\mathsf{a}), \mathsf{G}(\mathsf{a}))$. Then $t = \mathsf{f}(\mathsf{H}(\mathsf{a}), \mathsf{I})\ {}_2\lhd\rhd\ s \rhd\rhd_2 \mathsf{f}(\mathsf{I}, \mathsf{H}(\mathsf{a})) = u$. Note that $\mathsf{I} \to_{\mathcal{R}} \mathsf{K}$ and $\mathsf{H}(\mathsf{a}) \to_{\mathcal{R}} \mathsf{J} \to_{\mathcal{R}} \mathsf{K}$. The base contexts of t and u are $\mathsf{f}(\Box, \mathsf{I})$ and $\mathsf{f}(\mathsf{I}, \Box)$, respectively. Consequently, $t \rhd\rhd_1 \mathsf{f}(\mathsf{K}, \mathsf{I}) \blacktriangleright\blacktriangleright \mathsf{f}(\mathsf{K}, \mathsf{K}) \blacktriangleleft\blacktriangleleft \mathsf{f}(\mathsf{I}, \mathsf{K})\ {}_1\lhd\rhd\ u$.

Lemma 5.38. *Peaks involving a tall and a short step are locally decreasing:*

$$ {}_{\iota}\lhd\rhd \cdot \blacktriangleright\blacktriangleright \;\subseteq\; \rhd\rhd_{<\iota}^{=} \cdot \blacktriangleright\blacktriangleright^* \cdot {}^*\blacktriangleleft\blacktriangleleft \cdot {}_{\leqslant\iota}\lhd\rhd $$

Proof. Let $t\ {}_{\iota}\lhd\rhd\ s \blacktriangleright\blacktriangleright u$ and let the base context and base sequence of s be B and $\mathbf{s}$. We have $t = B[\mathbf{t}]$ with $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{t}$ for some foreign sequence $\mathbf{t}$ and $u = C[\mathbf{u}]$. We construct $\mathbf{v}$ and $\mathbf{w}$ such that

$$ B[\mathbf{t}] \rhd\rhd_{<\iota}^{=} \cdot \blacktriangleright\blacktriangleright^* B[\mathbf{v}] \blacktriangleright\blacktriangleright^* C[\mathbf{w}]\ {}^*\blacktriangleleft\blacktriangleleft \cdot {}_{\leqslant\iota}\lhd\rhd\ C[\mathbf{u}] $$

We distinguish two cases.

1. If $\mathbf{s} \propto \mathbf{t}$ then we let $\mathbf{v} = \mathbf{t}$. Hence $B[\mathbf{t}] = B[\mathbf{v}]$ and thus $B[\mathbf{t}] \rhd\rhd_{<\iota}^{=} \cdot \blacktriangleright\blacktriangleright^* B[\mathbf{v}]$.

2. Otherwise, using Lemma 5.28 with $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{t}$ and $\mathbf{s} \to_{\mathcal{R}}^* \mathbf{s}$ we can find a foreign sequence $\mathbf{v}$ such that $\mathbf{t} \to_{\mathcal{R}}^* \mathbf{v}$, $\mathbf{t} \propto \mathbf{v}$, and $\mathbf{s} \propto \mathbf{v}$. Since the imbalance of $\mathbf{v}$ is less than ι ($\mathbf{s} \not\propto \mathbf{t}$ means that there are i, j with $s_i = s_j$ and $t_i \neq t_j$. By $\mathbf{s} \propto \mathbf{v}$, we have $v_i = v_j$, and $\mathbf{t} \propto \mathbf{v}$ ensures that all other equalities between elements of $\mathbf{t}$ carry over to $\mathbf{v}$, so the imbalance becomes smaller) we obtain $B[\mathbf{t}] \rhd\rhd_{<\iota}^{=} \cdot \blacktriangleright\blacktriangleright^* B[\mathbf{v}]$ from Lemma 5.34.

By the definition of $\blacktriangleright\blacktriangleright$ we get $B \to_{\mathcal{R}}^* C$ mirroring $s = B[\mathbf{s}] \to_{\mathcal{R}}^* C[\mathbf{u}] = u$. Hence $\mathbf{u}$ is a sequence of foreign terms such that all elements of $\mathbf{u}$ are elements of $\mathbf{s}$, which follows by repeated application of Lemma 5.12. We define $w_i = v_j$ if $u_i = s_j$. Then $\mathbf{u} \to_{\mathcal{R}}^* \mathbf{w}$ and the imbalance of $\mathbf{w}$ is at most ι. Hence $C[\mathbf{u}] \rhd\rhd_{\leqslant\iota} \cdot \blacktriangleright\blacktriangleright^* C[\mathbf{w}]$ by Lemma 5.34. We also have $B[\mathbf{v}] \to_{\mathcal{R}}^* C[\mathbf{w}]$ with no rewrite step affecting a tall alien and thus $B[\mathbf{v}] \blacktriangleright\blacktriangleright^* C[\mathbf{w}]$ by Lemma 5.33. □

Example 5.39. We revisit Example 5.25. Let $s = \mathsf{f}(\mathsf{f}(\mathsf{G}(\mathsf{a}), \mathsf{G}(\mathsf{a})), \mathsf{I})$. The base context of s is $\mathsf{f}(\mathsf{f}(\Box, \Box), \mathsf{I})$. Then $t = \mathsf{f}(\mathsf{f}(\mathsf{I}, \mathsf{H}(\mathsf{a})), \mathsf{I}) \;{}_2\!\lhd\!\!\lhd\; s \blacktriangleright\!\blacktriangleright \mathsf{f}(\mathsf{G}(\mathsf{a}), \mathsf{K}) = u$. The base context of t is $\mathsf{f}(\mathsf{f}(\mathsf{I}, \Box), \mathsf{I})$ and we have $t \rhd\!\rhd_1 \mathsf{f}(\mathsf{f}(\mathsf{I}, \mathsf{K}), \mathsf{I}) \blacktriangleright\!\blacktriangleright \mathsf{f}(\mathsf{f}(\mathsf{K}, \mathsf{K}), \mathsf{K}) \blacktriangleright\!\blacktriangleright \mathsf{f}(\mathsf{K}, \mathsf{K}) = v$, whereas the base context of u is $\mathsf{f}(\Box, \mathsf{K})$ and $u \rhd\!\rhd_1 v$.

Lemma 5.40 (Main Lemma)**.** *If $\blacktriangleright\!\blacktriangleright$ is locally decreasing then $\mathcal{R}$ is confluent on native terms.*

Proof. Every rewrite step $s \to_\mathcal{R} t$ can be written as $s \blacktriangleright\!\blacktriangleright t$ by Lemma 5.29 or $s \rhd\!\rhd t$ if the rewrite rule is applied to a tall alien of s. Consequently, $\to_\mathcal{R} \subseteq \rhd\!\rhd \cup \blacktriangleright\!\blacktriangleright \subseteq \to_\mathcal{R}^*$ and the claim follows from the confluence of $\rhd\!\rhd \cup \blacktriangleright\!\blacktriangleright$. The latter is a consequence of Theorem 3.21 in connection with the assumption and Lemmata 5.36 and 5.38. □

The various versions of the main theorem will follow from Lemma 5.40.

5.3.3 Local Decreasingness of Short Steps

In this section we study conditions to make short steps locally decreasing. The following result allows to represent a native term s by a foreign term s' and a substitution π such that $s = s'\pi$. This will be the key for joining the peak originating from s by the confluence assumption of s'.

Lemma 5.41 (peak analysis)**.** *For a peak $t \blacktriangleleft\!\blacktriangleleft s \blacktriangleright\!\blacktriangleright u$ there are foreign terms s', t', u', v' and substitutions π, $\pi_\Box$ such that*

1. *π is a bijection with $\mathrm{dom}(\pi) \cap \mathcal{V}ar(s) = \varnothing$,*
2. *$s'\pi = s$, $t'\pi = t$, $u'\pi = u$, $s'\pi_\Box$ is the base context of s, and $t'\pi_\Box$ and $u'\pi_\Box$ are shallow contexts of t and u, and*
3. *$v' \;{}^*_\mathcal{R}\!\!\leftarrow t' \;{}^*_\mathcal{R}\!\!\leftarrow s' \to_\mathcal{R}^* u' \to_\mathcal{R}^* v'$ and $t \to_\mathcal{R}^* v \;{}^*_\mathcal{R}\!\!\leftarrow u$ with $v = v'\pi$.*

Proof. Let $s = B[\mathbf{s}]$ be the decomposition of s into base context and base sequence, and recall that base contexts are shallow. According to the definition of $\blacktriangleright\!\blacktriangleright$ there are rewrite sequences $B \to_\mathcal{R}^* C_t$, $B \to_\mathcal{R}^* C_u$ mirroring $s \to_\mathcal{R}^* t$, $s \to_\mathcal{R}^* u$, respectively. Using Lemma 5.31 repeatedly, we find that C_t and C_u are shallow contexts. Let π be a bijection between the tall aliens of s and fresh variables, and define $s' = B[\pi^{-1}(\mathbf{s})]$. We have $\mathbf{s} \propto \pi^{-1}(\mathbf{s})$ and therefore repeated application of Lemma 5.27 yields rewrite sequences $s' \to_\mathcal{R}^* t'$ and $s' \to_\mathcal{R}^* u'$ mirroring $s'\pi = s \to_\mathcal{R}^* t = t'\pi$ and $s'\pi = s \to_\mathcal{R}^* u = u'\pi$. Since s' is a foreign term and therefore confluent, t' and u' have a common reduct: $t' \to_\mathcal{R}^* v' \;{}^*_\mathcal{R}\!\!\leftarrow u'$. By applying π to this valley we obtain $t \to_\mathcal{R}^* v \;{}^*_\mathcal{R}\!\!\leftarrow u$. Note that $s'\pi_\Box = B$, $t'\pi_\Box = C_t$ and $u'\pi_\Box = C_u$ are shallow contexts as claimed. □

Example 5.42. Consider the layer system $\mathbb{L}$ given by

$$\mathbb{L}_0 = \{v, \mathsf{a}, \mathsf{b}, \mathsf{f}(v), \mathsf{g}(v), \mathsf{g}(\mathsf{b}) \mid v \in \mathcal{V}_\Box\}$$
$$\mathbb{L} = \mathbb{L}_0 \cup \{\mathsf{h}(C, C', C'') \mid C, C', C'' \in \mathbb{L}_0\}$$

which weakly layers the TRS $\mathcal{R} = \{\mathsf{h}(x,y,z) \to \mathsf{h}(y,x,z), \mathsf{f}(x) \to \mathsf{g}(x), \mathsf{a} \to \mathsf{b}\}$. Assume that $r = 1$ and let $s = \mathsf{h}(\mathsf{a}, \mathsf{f}(\mathsf{a}), \mathsf{f}(\mathsf{b}))$. The base context of s is $\mathsf{h}(\mathsf{a}, \mathsf{f}(\Box), \mathsf{f}(\Box))$. There is a peak of short steps

$$t = \mathsf{h}(\mathsf{b}, \mathsf{g}(\mathsf{a}), \mathsf{f}(\mathsf{b})) \blacktriangleleft\!\blacktriangleleft\ s\ \blacktriangleright\!\blacktriangleright\ \mathsf{h}(\mathsf{f}(\mathsf{a}), \mathsf{a}, \mathsf{g}(\mathsf{b})) = u$$

From Lemma 5.41, we may obtain $\pi = \{\mathsf{a}/x, \mathsf{b}/y\}$, $s' = \mathsf{h}(\mathsf{a}, \mathsf{f}(x), \mathsf{g}(y))$, $t' = \mathsf{h}(\mathsf{b}, \mathsf{g}(x), \mathsf{f}(y))$, $u' = \mathsf{h}(\mathsf{f}(x), \mathsf{a}, \mathsf{g}(y))$, and $v' = \mathsf{h}(\mathsf{g}(x), \mathsf{b}, \mathsf{g}(y))$. Note that $t'\pi_\Box = \mathsf{h}(\mathsf{b}, \mathsf{g}(\Box), \mathsf{f}(\Box))$ is the base context of t but $u'\pi_\Box = \mathsf{h}(\mathsf{f}(\Box), \mathsf{a}, \mathsf{g}(\Box))$ does not equal $\mathsf{h}(\mathsf{f}(\Box), \mathsf{a}, \mathsf{g}(\mathsf{b}))$, the base context of u.

Lemma 5.43. *If $\mathcal{R}$ is left-linear then $\blacktriangleright\!\blacktriangleright$ is locally decreasing.*

Proof. Consider a local peak $t\ {}_{s_0}\!\blacktriangleleft\!\blacktriangleleft\ s\ \blacktriangleright\!\blacktriangleright_{s_1}\ u$. First we apply Lemma 5.41. Let t'' be a linearization of t', which we obtain by replacing each variable in t' by a fresh variable. Because $\mathcal{R}$ is left-linear, $t' \to_\mathcal{R}^* v'$ can be mirrored as $t'' \to_\mathcal{R}^* v''$. Let B_t be the base context of t and $C_t = t'\pi_\Box$. We have $C_t \sqsubseteq B_t$ by Lemma 5.32, which implies $t'' \leqslant B_t$ and thus $B_t = t''\sigma$ for some substitution σ. We have $B_t \to_\mathcal{R}^* v''\sigma$. Together with $t \to_\mathcal{R}^* v$, which mirrors $B_t \to_\mathcal{R}^* v''\sigma$, we obtain $t \blacktriangleright\!\blacktriangleright v$. This step can be labeled with s_1 because $s_1 \to_\mathcal{R}^* s \to_\mathcal{R}^* t$. By symmetry we obtain $u \blacktriangleright\!\blacktriangleright_{s_0} v$ and hence $\blacktriangleright\!\blacktriangleright$ is locally decreasing. □

Next we deal with bounded duplicating TRSs. In order to exploit relative termination, we insert $\Diamond$ symbols in front of tall aliens as follows.

Definition 5.44. Let s be a native term with base context B and base sequence $\mathbf{s}$. Then $s^\Diamond = B[\Diamond(\mathbf{s})]$ where $\Diamond(\mathbf{s})$ denotes the result of replacing each element u of $\mathbf{s}$ by $\Diamond(u)$.

Lemma 5.45. *If $s \to_\mathcal{R} t$ then $s^\Diamond \to_\mathcal{R} \cdot \to_{\Diamond(x)\to x}^* t^\Diamond$.*

Proof. Let $s \to_{p,\ell\to r} t$ and let B be the base context of s. If $p \in \mathcal{P}os_\mathcal{F}(B)$ then by Lemma 5.27 we obtain a term t' and a context C such that $s^\Diamond \to_{p,\ell\to r} t'$ and $B \to_{p,\ell\to r} C$. Decomposing t as $t = C[\mathbf{t}]$ we find that $t' = C[\Diamond(\mathbf{t})]$. If $p \notin \mathcal{P}os_\mathcal{F}(B)$, then the rewrite step is within a tall alien of s. Hence letting $C = B$ and decomposing t as $C[\mathbf{t}]$, we find that $s^\Diamond = C[\Diamond(\mathbf{s})] \to_\mathcal{R} C[\Diamond(\mathbf{t})]$. In either case, Lemma 5.9 (with L equal to the max-top of C) shows that the tall aliens of t are a subsequence of $\mathbf{t}$, and therefore $C[\Diamond(\mathbf{t})] \to_{\Diamond(x)\to x}^* t^\Diamond$, using that $\Diamond(t_i) \to_{\Diamond(x)\to x} t_i$ for those t_i that are not tall aliens. □

Lemma 5.46. *If $\mathcal{R}$ is bounded duplicating then $\blacktriangleright\!\blacktriangleright$ is locally decreasing.*

Proof. Since $\mathcal{R}$ is bounded duplicating, we may assume a fresh function symbol $\Diamond$ such that $\{\Diamond(x) \to x\}/\mathcal{R}$ is terminating. In order to compare the labels we define a well-founded order on terms by $s_0 > s_1$ if $s_0^\Diamond \to_{\{\Diamond(x)\to x\}/\mathcal{R}}^+ s_1^\Diamond$. Consider a peak $t\ {}_{s_0}\!\blacktriangleleft\!\blacktriangleleft\ s\ \blacktriangleright\!\blacktriangleright_{s_1}\ u$ which we first subject to Lemma 5.41. We analyze the sequence $t \to_\mathcal{R}^* v$ resulting from the peak analysis by distinguishing two cases.

1. If $t'\pi_\Box$ is the base context of t then the rewrite sequence $t'\pi_\Box \to_\mathcal{R}^* v'\pi_\Box$ mirrors $t \to_\mathcal{R}^* v$. Hence we obtain $t \blacktriangleright\!\blacktriangleright_{s_1} v$, noting that the label s_1 satisfies $s_1 \to_\mathcal{R}^* s \to_\mathcal{R}^* t$.

2. If $t'\pi_\Box$ is not the base context then like in the proof of Lemma 5.45, we can decompose t as $t = t'\pi_\Box[\mathbf{t}']$ in order to obtain $s^\Diamond \to_\mathcal{R}^* t'\pi_\Box[\Diamond(\mathbf{t}')]$. Since $t'\pi_\Box$ is not the base context, the tall aliens of t are a proper subsequence of $\mathbf{t}'$ and therefore, $t'\pi_\Box[\Diamond(\mathbf{t}')] \to^+_{\Diamond(x)\to x} t^\Diamond$. We also have $s_1 \to_\mathcal{R}^* s$, which implies $s_1^\Diamond \to^*_{\mathcal{R}\cup\{\Diamond(x)\to x\}} s$ by Lemma 5.45. As a consequence, $s_1^\Diamond \to^+_{\mathcal{R}/\{\Diamond(x)\to x\}} t^\Diamond$ and $s_1 > t$ follow. By applying Lemma 5.29 several times we obtain $t \blacktriangleright\!\blacktriangleright^*_t v$ and thus $t \blacktriangleright\!\blacktriangleright^*_{\vee s_1} v$.

The analogous analysis of $u \to_\mathcal{R}^* v$ yields $u \blacktriangleright\!\blacktriangleright_{s_0} v$ or $u \blacktriangleright\!\blacktriangleright^*_{\vee s_0} v$ and hence $\blacktriangleright\!\blacktriangleright$ is locally decreasing. □

Finally, we prepare for the main result about layered TRSs, where condition (C_1) of Definition 5.3 is crucial.

Lemma 5.47. *Let $\mathcal{R}$ be a layered TRS and $t \to_{p,\ell\to r} t'$ for native terms t and t' If $p \in \mathcal{P}os_\mathcal{F}(B)$ for the base context B of t then either $B \to_{p,\ell\to r} B'$ for the base context B' of t' or t' is its own base context.*

Proof. Let M and M' be the max-tops of t and t'. We distinguish two cases.

1. If $p \in \mathcal{P}os_\mathcal{F}(M)$ then by (C_1) either $M \to_{p,\ell\to r} \Box$ or $M \to_{p,\ell\to r} M'$. In the former case t' equals an alien of t. Since the rank of t' is at most r, t' is its own base context. So assume $M \to_{p,\ell\to r} M'$. By Lemma 5.11 there exist a term m and a substitution σ such that $m \to_{p,\ell\to r} m'$ for some m' (since $\ell \leqslant m|_p$), $t = m\sigma$, and $M = m\sigma_\Box$. Define a substitution τ as follows:

$$\tau(x) = \begin{cases} \Box & \text{if } x \in \mathrm{dom}(\sigma_\Box) \text{ and } \sigma(x) \text{ is a tall alien of } t \\ \sigma(x) & \text{otherwise} \end{cases}$$

   We have $B = m\tau$ by construction of τ. Let $B' = m'\tau$. Clearly $B \to_{p,\ell\to r} B'$. By comparing $m'\tau$ to $M' = m'\sigma_\Box$, we see that B' is the base context of t'.

2. If $p \notin \mathcal{P}os_\mathcal{F}(M)$ then a short alien of t is rewritten. By letting B and $\mathbf{t}$ be the base context and base sequence of t, by Lemma 5.12 we obtain a rewrite step $t = B[\mathbf{t}] \to_{p,\ell\to r} B'[\mathbf{t}'] = t'$ with $\mathbf{t}' = \mathbf{t}$ because p is parallel to the hole positions of B. We claim that B' is the base context of t'. Suppose to the contrary that some t_i is not a tall alien of t'. Let q be its position in t, which is also its position in t'. Since $q \in \mathcal{P}os_\Box(M)$ and $q \notin \mathcal{P}os_\Box(M')$, $M \sqsubset M[M'|_q]_q$. Hence $M[M'|_q]_q \in \mathbb{L}$ by (C_2) and thus $M[M'|_q]_q \sqsubseteq t$, contradicting the fact that M is a max-top of t. □

□

The following example shows that (C_2) is essential for Lemma 5.47.

Example 5.48. Recall Figure 5.1 and the underlying layer system $\mathbb{L}$, which satisfies (W) and (C_1). However (C_2) is violated, e.g., we have $L = \mathsf{k}(\Box,\Box) \in \mathbb{L}$ and $N = \mathsf{k}(\mathsf{h}(\Box),\mathsf{h}(\Box)) \in \mathbb{L}$ but $L[N|_2]_2 = \mathsf{k}(\Box,\mathsf{h}(\Box)) \notin \mathbb{L}$. Consider the term $t = \mathsf{k}(\mathsf{f}(\mathsf{a}),\mathsf{h}(\mathsf{a}))$ of rank 3. Its base context is $B = \mathsf{k}(\mathsf{f}(\mathsf{a}),\Box)$. We have $t \to \mathsf{k}(\mathsf{h}(\mathsf{a}),\mathsf{h}(\mathsf{a})) =: t'$. The base context of t' is $\mathsf{k}(\mathsf{h}(\Box),\mathsf{h}(\Box)) =: B'$ but $B \not\to_\mathcal{R} B'$.

Lemma 5.49. *If $\mathcal{R}$ is layered then $\blacktriangleright\!\blacktriangleright$ is locally decreasing.*

Proof. Consider a local peak $t\ {}_{s_0}\!\blacktriangleleft\!\blacktriangleleft\ s\ \blacktriangleright\!\blacktriangleright_{s_1}\ u$. First we analyze the peak by Lemma 5.41. The rewrite sequence $t'\pi_\Box \to_\mathcal{R}^* v'\pi_\Box$ mirrors $t = t'\pi \to_\mathcal{R}^* v'\pi = v$. We find by repeated application of Lemma 5.47 that the base context B_t of t equals $t'\pi_\Box$ or t. In both cases, we have $t \blacktriangleright\!\blacktriangleright_{s_1} v$, noting that $t \to_\mathcal{R}^* v$ mirrors itself, and that $s_1 \to_\mathcal{R}^* t$. We obtain $u \blacktriangleright\!\blacktriangleright_{s_0} v$ in the same way and hence $\blacktriangleright\!\blacktriangleright$ is locally decreasing. □

5.3.4 Proof of Main Theorems

Because the proofs are similar, we prove all main results in one go.

Proof of Theorems 5.14, 5.16, and 5.19. By assumption the TRS $\mathcal{R}$ is weakly layered and confluent on terms of rank one. We have to show that

- if $\mathcal{R}$ is left-linear then $\mathcal{R}$ is confluent (Theorem 5.14),
- if $\mathcal{R}$ is bounded duplicating then $\mathcal{R}$ is confluent (Theorem 5.16), and
- if $\mathcal{R}$ is layered then $\mathcal{R}$ is confluent (Theorem 5.19).

We show confluence of all terms by induction on the rank r of a term. In the base case we consider terms of rank one, which are confluent by assumption. Assume as induction hypothesis that confluence of terms of rank r or less has been established. We consider terms of rank $r + 1$, to which the analysis of Sections 5.3.1–5.3.3 applies. By Lemma 5.40 in conjunction with Lemma 5.43 (for weakly layered left-linear $\mathcal{R}$), Lemma 5.46 (for weakly layered bounded duplicating $\mathcal{R}$), or Lemma 5.49 (for layered $\mathcal{R}$), we obtain confluence of $\mathcal{R}$ on terms of rank up to $r + 1$, completing the induction step. □

5.4 Applications

In this section the abstract confluence results via layer systems are instantiated by concrete applications. Section 5.4.1 treats the plain modularity case [75] and Section 5.4.2 covers layer-preservation [54]. The result for quasi-ground systems [42] is less known but also fits our framework, as outlined in Section 5.4.3. Currying [39] is the topic of Section 5.4.4, before many-sorted persistence [4] is discussed in Section 5.4.5.

For the results in this section the reverse directions also hold. We do not give the (easy) proofs since they do not require layer systems.

In Sections 5.4.1, 5.4.2, and 5.4.3 we deal with two TRSs $\mathcal{R}_1$ and $\mathcal{R}_2$ that are defined over the respective signatures $\mathcal{F}_1$ and $\mathcal{F}_2$. We let $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_2$ and $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$.

5.4.1 Modularity

We recall the classical modularity result for confluence [75].

Theorem 5.50. *Suppose $\mathcal{F}_1 \cap \mathcal{F}_2 = \varnothing$. If $\mathcal{R}_1$ and $\mathcal{R}_2$ are confluent then $\mathcal{R}$ is confluent.*

Proof. Define

$$\mathbb{L} := \mathcal{C}(\mathcal{F}_1, \mathcal{V}) \cup \mathcal{C}(\mathcal{F}_2, \mathcal{V})$$

We show that $\mathcal{R}$ is layered. Since $\mathcal{V} \subseteq \mathbb{L}$ and $f(\Box, \ldots, \Box) \in \mathbb{L}$ for all function symbols $f \in \mathcal{F}_1 \cup \mathcal{F}_2$, every term in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ has a non-empty top. Hence condition (L_1) holds. Also condition (L_2) holds because $\mathbb{L}$ is closed under the operation of interchanging variables and holes. For condition (L_3) we observe that if $L \in \mathcal{C}(\mathcal{F}_i, \mathcal{V})$, $p \in \mathcal{P}os_{\mathcal{F}}(L)$, and $N \in \mathbb{L}$ such that $L|_p \sqcup N$ is defined then $\mathrm{root}(L|_p) \in \mathcal{F}_i$ and thus $N \in \mathcal{C}(\mathcal{F}_i, \mathcal{V})$. Consequently, $L[L|_p \sqcup N]_p \in \mathcal{C}(\mathcal{F}_i, \mathcal{V}) \subseteq \mathbb{L}$. Since each rule is over a single signature, and layers are closed under rewriting, condition (W) follows easily. For condition (C_1) we consider a term s with max-top M, $p \in \mathcal{P}os_{\mathcal{F}}(M)$, and rewrite step $s \to_{p,\ell\to r} t$ which is mirrored by $M \to_{p,\ell\to r} L$. Suppose $M \in \mathcal{C}(\mathcal{F}_i, \mathcal{V})$. We have $L \in \mathcal{C}(\mathcal{F}_i, \mathcal{V})$. The case $L = \Box$ is obtained when t is an alien of s, which is only possible if the rule $\ell \to r$ is collapsing. Otherwise L is the max-top of t since the root symbols of aliens of s belong to $\mathcal{F}_{3-i}$ and hence cannot fuse with L to form a larger top. Finally, condition (C_2) holds because if $N \in \mathcal{C}(\mathcal{F}_i, \mathcal{V})$ then $L \sqsubseteq N$ implies $L \in \mathcal{C}(\mathcal{F}_i, \mathcal{V})$ and thus also $L[N|_p]_p$ belongs to $\mathcal{C}(\mathcal{F}_i, \mathcal{V})$.

According to Theorem 5.19, $\mathcal{R}$ is confluent if we show that $\mathcal{R}$ is confluent on terms of rank one. The latter follows from the fact that rewriting does not increase the rank of a term (Lemma 5.13) together with the observation that non-variable terms of rank one belong to either $\mathcal{T}(\mathcal{F}_1, \mathcal{V})$ or $\mathcal{T}(\mathcal{F}_2, \mathcal{V})$ and only rewrite rules of $\mathcal{R}_i$ apply to terms in $\mathcal{T}(\mathcal{F}_i, \mathcal{V})$, in connection with the confluence assumptions of $\mathcal{R}_1$ and $\mathcal{R}_2$. □

5.4.2 Layer-Preservation

Layer-preserving TRSs are a special class of TRSs with shared function symbols for which confluence is modular as shown in [54]. In this section, we reprove this result using layer systems. Let $\mathcal{T}_X(\mathcal{F}, \mathcal{V})$ denote the set of terms with root symbol from X. Let $\mathcal{B} := \mathcal{F}_1 \cap \mathcal{F}_2$, $\mathcal{D}_1 := \mathcal{F}_1 \setminus \mathcal{F}_2$ and $\mathcal{D}_2 := \mathcal{F}_2 \setminus \mathcal{F}_1$. The result on layer preservation can be stated as follows.

Theorem 5.51. *Let $\mathcal{R}_1 \subseteq \mathcal{T}(\mathcal{B}, \mathcal{V})^2 \cup \mathcal{T}_{\mathcal{D}_1}(\mathcal{F}_1, \mathcal{V})^2$, $\mathcal{R}_2 \subseteq \mathcal{T}(\mathcal{B}, \mathcal{V})^2 \cup \mathcal{T}_{\mathcal{D}_2}(\mathcal{F}_2, \mathcal{V})^2$, and $\mathcal{R}_1 \cap \mathcal{T}(\mathcal{B}, \mathcal{V})^2 = \mathcal{R}_2 \cap \mathcal{T}(\mathcal{B}, \mathcal{V})^2$. If $\mathcal{R}_1$ and $\mathcal{R}_2$ are confluent then $\mathcal{R}$ is confluent.*

Proof. We define

$$\mathbb{L} := \mathcal{C}(\mathcal{B}, \mathcal{V}) \cup \mathcal{T}_{\mathcal{D}_1}(\mathcal{F}_1 \cup \{\Box\}, \mathcal{V}) \cup \mathcal{T}_{\mathcal{D}_2}(\mathcal{F}_2 \cup \{\Box\}, \mathcal{V})$$

It is easy to verify that $\mathbb{L}$ layers $\mathcal{R} := \mathcal{R}_1 \cup \mathcal{R}_2$, much like in the modularity case. In particular, $\mathbb{L}$ is closed under rewriting. Consider a term s of rank one

and a peak $t \mathrel{{}^*_{\mathcal{R}}\!\leftarrow} s \to^*_{\mathcal{R}} u$. Let $i \in \{1,2\}$ be such that $s \in \mathcal{T}(\mathcal{F}_i, \mathcal{V})$. The only rules of $\mathcal{R}_{3-i}$ that can be used in the peak come from $\mathcal{T}(\mathcal{B}, \mathcal{V})^2$ and hence also appear in $\mathcal{R}_i$. Since $\mathcal{R}_i$ is confluent on $\mathcal{T}(\mathcal{F}_i, \mathcal{V})$ we obtain joinability of t and u in $\mathcal{R}_i$ and thus also in $\mathcal{R}$. Hence $\mathcal{R}$ is confluent on terms of rank one and we conclude by Theorem 5.19. □

Toyama's modularity result has been adapted in [55] to constructor-sharing combinations in which the participating TRSs may share constructor symbols under the additional condition that neither collapsing nor constructor-lifting rules are present. This result is subsumed by Theorem 5.51, cf. [57, p. 249]. Still, layer-preservation and modularity are incomparable (since layer-preservation places collapsing rules in both systems).

5.4.3 Quasi-Ground Systems

We show modularity of quasi-ground TRSs [42, Theorem 1] using layer systems.

Definition 5.52. We call a context C *quasi-ground* if for all $p \in \mathcal{P}os(C)$ with $\mathrm{root}(C|_p) \in \mathcal{F}_1 \cap \mathcal{F}_2$, $C|_p$ is ground over $\mathcal{F}$, i.e., $C|_p \in \mathcal{T}(\mathcal{F})$.

Theorem 5.53. *Suppose* $\mathrm{root}(\ell) \notin \mathcal{F}_1 \cap \mathcal{F}_2$ *and* ℓ *and* r *are quasi-ground, for all* $\ell \to r \in \mathcal{R}$. *If* $\mathcal{R}_1$ *and* $\mathcal{R}_2$ *are confluent then* $\mathcal{R}$ *is confluent.*

Proof. We define a layer system $\mathbb{L} := \mathbb{L}_1 \cup \mathbb{L}_2 \cup \mathbb{L}_c$ with

$$\mathbb{L}_i = \{C \in \mathcal{C}(\mathcal{F}_i, \mathcal{V}) \mid C \text{ is quasi-ground}\} \quad \text{for } i = 1, 2$$
$$\mathbb{L}_c = \{f(v_1, \ldots, v_n) \mid f \in \mathcal{F}_1 \cap \mathcal{F}_2 \text{ and } v_i \in \mathcal{V}_\Box \text{ for } 1 \leqslant i \leqslant n\}$$

We readily check that (L_1) and (L_2) are satisfied. For (L_3), $\mathbb{L}_1$, $\mathbb{L}_2$ and $\mathbb{L}_c$ are individually closed under merging at function positions. Fix $i \in \{1,2\}$. If we merge $L \in \mathbb{L}_i$ with $N \in \mathbb{L}_{3-i} \cup \mathbb{L}_c$ at $p \in \mathcal{P}os_{\mathcal{F}}(L)$ then either $N = \Box$ and $L[L|_p \sqcup N] = L \in \mathbb{L}_i$, or $\mathrm{root}(L|_p) \in \mathcal{F}_1 \cap \mathcal{F}_2$, which implies $L|_p \in \mathcal{T}(\mathcal{F})$ and hence $L[L|_p \sqcup N]_p = L[L|_p]_p = L \in \mathbb{L}_i$. Note that $L \in \mathbb{L}_c$ can be merged with $N \in \mathbb{L}_i$ only at position $p = \epsilon$. If $N = \Box$ then $L \sqcup \Box = L \in \mathbb{L}_c$ and otherwise $L \sqcup N = N \in \mathbb{L}_i$. For (W) we let M be the max-top of s, $p \in \mathcal{P}os_{\mathcal{F}}(M)$, and consider a rewrite step $s \to_{p,\ell\to r} t$. We assume without loss of generality that $\ell \to r \in \mathcal{R}_1$. Hence $M \in \mathbb{L}_1$ because $\mathrm{root}(\ell) \in \mathcal{F}_1 \setminus \mathcal{F}_2$. Note that $\mathbb{L}_1$ is closed under taking subterms and that for any substitution $\tau\colon \mathcal{V} \to \mathbb{L}_1$ we have $\ell\tau \in \mathbb{L}_1$. Let σ be a substitution such that $s|_p = \ell\sigma$ and let τ be the substitution that maps each variable $x \in \mathcal{V}ar(\ell)$ to the $\mathbb{L}_1$-max-top of $\sigma(x)$. We have $M = M[\ell\tau]_p$ and thus $M \to_{p,\ell\to r} L$ with $L = M[r\tau]_p \in \mathbb{L}_1$. For (C_1) it is easy to see that L is the $\mathbb{L}_1$-max-top of t. Suppose $L \neq \Box$. We claim that L is the max-top (with respect to $\mathbb{L}$) of t. This follows from the observation that if there is a top of t that comes from $\mathbb{L}_2$ or $\mathbb{L}_c$ then $\mathrm{root}(L) \in \mathcal{F}_1 \cap \mathcal{F}_2$ and thus $L \in \mathcal{T}(\mathcal{F})$, which cannot be made larger. Condition (C_2) follows as in the proof of Theorem 5.50.

Now let $\mathcal{R}_1$ and $\mathcal{R}_2$ be confluent. We show that $\mathcal{R}$ is confluent on terms of rank one. Consider a term of rank one. Note that rules from $\mathcal{R}_1$ only apply to elements of $\mathbb{L}_1$. Furthermore, $\mathbb{L}_1$ is closed under rewriting by $\mathcal{R}_1$. Likewise, rules from $\mathcal{R}_2$ only apply to elements of $\mathbb{L}_2$, which is closed under rewriting by

$\mathcal{R}_2$. We conclude that $\mathcal{R}$ is confluent on terms of rank one and by Theorem 5.19 this implies that $\mathcal{R}$ is confluent. □

5.4.4 Currying

Currying is a transformation of TRSs such that the resulting TRS has only one non-constant function symbol Ap that represents partial applications. It is useful in the construction of polynomial-time procedures for deciding properties of TRSs, e.g., [14]. [39] proved that confluence is preserved by currying.

Definition 5.54. Given a TRS $\mathcal{R}$ over a signature $\mathcal{F}$, let $\mathcal{F}_\mathcal{C} = \{\mathsf{Ap}\} \cup \{f_0 \mid f \in \mathcal{F}\}$ where Ap is a fresh binary function symbol and all function symbols in $\mathcal{F}$ become constants. The *curried version* $\mathsf{Cu}(\mathcal{R})$ of $\mathcal{R}$ is the TRS over the signature $\mathcal{F}_\mathcal{C}$ with rules $\{\mathsf{Cu}(\ell) \to \mathsf{Cu}(r) \mid \ell \to r \in \mathcal{R}\}$. Here $\mathsf{Cu}(t) = t$ if t is a variable or a constant and $\mathsf{Cu}(f(t_1, \dots, t_n)) = \mathsf{Ap}(\cdots \mathsf{Ap}(f_0, \mathsf{Cu}(t_1)) \cdots, \mathsf{Cu}(t_n))$ (with n occurrences of Ap). Let $\mathcal{F}_\mathcal{U} = \{\mathsf{Ap}\} \cup \{f_i \mid f \in \mathcal{F} \text{ and } 0 \leqslant i \leqslant \mathrm{arity}(f)\}$, where each f_i has arity i and $f_{\mathrm{arity}(f)}$ is identified with f. The *partial parametrization* $\mathsf{PP}(\mathcal{R})$ of $\mathcal{R}$ is the TRS $\mathcal{R} \cup \mathcal{U}$ over the signature $\mathcal{F}_\mathcal{U}$, where $\mathcal{U}$ consists of all *uncurrying* rules:

$$\mathsf{Ap}(f_i(x_1, \dots, x_i), x_{i+1}) \to f_{i+1}(x_1, \dots, x_{i+1})$$

for all $f \in \mathcal{F}$ and $0 \leqslant i < \mathrm{arity}(f)$.

The next example familiarizes the reader with the above concepts.

Example 5.55. For the TRS $\mathcal{R} = \{\mathsf{f}(x, x) \to \mathsf{f}(\mathsf{a}, \mathsf{b})\}$ we have

$$\begin{aligned} \mathsf{Cu}(\mathcal{R}) &= \{\mathsf{Ap}(\mathsf{Ap}(\mathsf{f}_0, x), x) \to \mathsf{Ap}(\mathsf{Ap}(\mathsf{f}_0, \mathsf{a}), \mathsf{b})\} \\ \mathcal{U} &= \{\mathsf{Ap}(\mathsf{f}_0, x) \to \mathsf{f}_1(x), \mathsf{Ap}(\mathsf{f}_1(x), y) \to \mathsf{f}(x, y)\} \\ \mathsf{PP}(\mathcal{R}) &= \mathcal{R} \cup \mathcal{U} \end{aligned}$$

Note that for a term $s = \mathsf{Ap}(\mathsf{Ap}(\mathsf{Ap}(\mathsf{f}_0, x), x), x)$ we have

$$s \to_{\mathsf{Cu}(\mathcal{R})} \mathsf{Ap}(\mathsf{Ap}(\mathsf{Ap}(\mathsf{f}_0, \mathsf{a}), \mathsf{b}), x)$$

and

$$s \to_\mathcal{U} \mathsf{Ap}(\mathsf{Ap}(\mathsf{f}_1(x), x), x) \to_\mathcal{U} \mathsf{Ap}(\mathsf{f}(x, x), x) \to_\mathcal{R} \mathsf{Ap}(\mathsf{f}(\mathsf{a}, \mathsf{b}), x)$$

so the partial parametrization is closely related to currying.

Note that $\mathcal{U}$ is both terminating and orthogonal, hence confluent. By $s{\downarrow_\mathcal{U}}$ we denote the unique $\mathcal{U}$-normal form of a term s.

Lemma 5.56 ([39, Proposition 3.1])**.** *Let $\mathcal{R}$ be a TRS. If $\mathsf{PP}(\mathcal{R})$ is confluent then $\mathsf{Cu}(\mathcal{R})$ is confluent.* □

Theorem 5.57 ([39, Theorem 5.2])**.** *Let $\mathcal{R}$ be a TRS. If $\mathcal{R}$ is confluent then $\mathsf{Cu}(\mathcal{R})$ is confluent.*

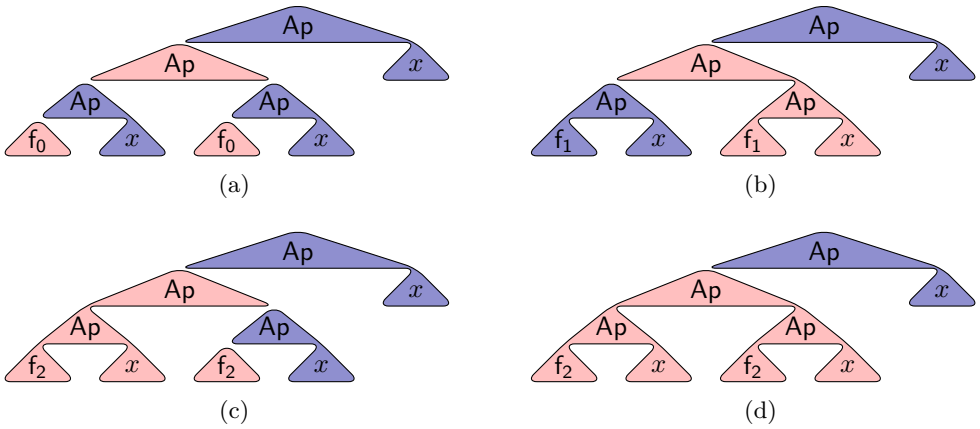


Figure 5.2: Layering terms in $\mathsf{PP}(\mathcal{R})$ for the TRS $\mathcal{R}$ in Example 5.55.

Proof. According to Lemma 5.56 it suffices to show that $\mathsf{PP}(\mathcal{R})$ is confluent. To this end, we let $\mathbb{L} := \mathbb{L}_1 \cup \mathbb{L}_2$, where $\mathbb{L}_1$ is the smallest extension of $\mathcal{V}_\Box$ such that

$$\mathsf{Ap}(\cdots \mathsf{Ap}(f_m(s_1, \ldots, s_m), s_{m+1}) \cdots, s_n) \in \mathbb{L}_1$$

for all $f_m \in \mathcal{F}_\mathcal{U} \setminus \{\mathsf{Ap}\}$, $s_1, \ldots, s_n \in \mathbb{L}_1$, with n less than or equal to the arity of f in the original TRS $\mathcal{R}$, and

$$\mathbb{L}_2 = \{\mathsf{Ap}(v, t) \mid v \in \mathcal{V}_\Box \text{ and } t \in \mathbb{L}_1\}$$

It is not difficult to see that $\mathbb{L}_1$ consists of those contexts in $\mathcal{C}(\mathcal{F}_\mathcal{U}, \mathcal{V})$ whose $\mathcal{U}$-normal form contains no occurrences of Ap. See Figure 5.2 for some layered terms.

We claim that $\mathsf{PP}(\mathcal{R})$ is layered. Conditions (L_1) and (L_2) are trivial and conditions (L_3) and (C_2) are easily shown by induction on the definition of $\mathbb{L}_1$. The interesting case for (L_3) is when $L \in \mathbb{L}_1$. Since merging cannot create new Ap symbols above any f_m, the result is in $\mathbb{L}_1$, whenever defined. For (W) and (C_1), we let M be the max-top of s, $p \in \mathcal{P}os_\mathcal{F}(M)$, and consider a rewrite step $s \to_{p,\ell\to r} t$ with $\ell \to r \in \mathsf{PP}(\mathcal{R})$ Because $\mathbb{L}$ is closed under taking subterms, $M|_p$ is a top of $s|_p$. It is the max-top because otherwise we could merge the max-top of $s|_p$ with M at position p and obtain a larger top of s. Note that $\ell_\Box \in \mathbb{L}_1$ (recall that $\ell_\Box$ is obtained by replacing all variables in ℓ by $\Box$) We have $\ell_\Box \sqsubseteq s|_p$ and therefore $\ell_\Box \sqsubseteq M|_p$. As a matter of fact, $M|_p$ is obtained from $\ell_\Box$ by replacing each hole at position q by the max-top (in $\mathbb{L}_1$) of $s|_{pq}$. Because equal subterms have equal max-tops, $s \preccurlyeq M|_p$ and hence there is a rewrite step $M \to_{p,\ell\to r} L$. We have $L \in \mathbb{L}_1$ because $\mathbb{L}_1$ is closed under rewriting by $\mathsf{PP}(\mathcal{R})$. Furthermore, the max-tops of the aliens of s do not belong to $\mathbb{L}_1$ and therefore the aliens of s are still aliens of L, unless $L = \Box$. It follows that both (W) and (C_1) hold.

To show confluence of $\mathsf{PP}(\mathcal{R})$ on terms of rank one, first note that elements of $\mathbb{L}_2$ allow no root steps and therefore it suffices to show confluence on terms in $\mathbb{L}_1$. It is easy to see that $s \to_{\mathcal{R}\cup\mathcal{U}} t$ implies $s{\downarrow_\mathcal{U}} \to_\mathcal{R}^= t{\downarrow_\mathcal{U}}$. Hence, for a peak

$t \; {}_{\mathcal{R}\cup\mathcal{U}}^{*}\!\leftarrow s \rightarrow_{\mathcal{R}\cup\mathcal{U}}^{*} u$ there is a corresponding peak $t{\downarrow_{\mathcal{U}}} \; {}_{\mathcal{R}}^{*}\!\leftarrow s{\downarrow_{\mathcal{U}}} \rightarrow_{\mathcal{R}}^{*} u{\downarrow_{\mathcal{U}}}$, which is joinable by the confluence of $\mathcal{R}$. Hence t and u are joinable in $\mathsf{PP}(\mathcal{R})$. We conclude by Theorem 5.19. □

5.4.5 Many-sorted Persistence

In this subsection, we prove persistence of confluence [3]. We begin by recalling many-sorted terms and rewriting.

Definition 5.58. Let S be a set of *sorts*. A *sort attachment* $\mathcal{S}$ *associates with each function symbol* $f \in \mathcal{F}$ *of arity* n *a type* $f : \alpha_1 \times \cdots \times \alpha_n \to \alpha$ *with* $\alpha_i, \alpha \in S$ *for* $1 \leqslant i \leqslant n$*, and with each variable* $x \in \mathcal{V}$ *a sort from* S. Let $\mathcal{V}_\alpha$ denote the set of variables of sort α. We assume that each $\mathcal{V}_\alpha$ is countably infinite.

Note that $\mathcal{V}_\alpha \cap \mathcal{V}_\beta = \varnothing$ for all $\alpha, \beta \in S$ whenever $\alpha \neq \beta$.

Definition 5.59. Let $\mathcal{S}$ be a sort attachment. We define terms of sort α inductively by $\mathcal{T}_\alpha(\mathcal{F}, \mathcal{V}) = \mathcal{V}_\alpha \cup \{f(t_1, \ldots, t_n) \mid f : \alpha_1 \times \cdots \times \alpha_n \to \alpha \text{ and } t_i \in \mathcal{T}_{\alpha_i}(\mathcal{F}, \mathcal{V}) \text{ for } 1 \leqslant i \leqslant n\}$. The set of many-sorted terms is defined $\mathcal{T}_\mathcal{S}(\mathcal{F}, \mathcal{V}) = \bigcup_{\alpha \in S} \mathcal{T}_\alpha(\mathcal{F}, \mathcal{V})$.

Definition 5.60. A TRS $\mathcal{R}$ is *compatible* with a sort attachment $\mathcal{S}$ if for each rule $\ell \to r \in \mathcal{R}$, there is a sort $\alpha \in S$ with $\ell, r \in \mathcal{T}_\alpha(\mathcal{F}, \mathcal{V})$.

Remark 5.61. If a TRS $\mathcal{R}$ is compatible with a sort attachment $\mathcal{S}$ then $\mathcal{T}_\alpha(\mathcal{F}, \mathcal{V})$ is closed under rewriting by $\mathcal{R}$, for each $\alpha \in S$.

Confluence is a persistent property of TRSs.

Theorem 5.62. *Let a TRS* $\mathcal{R}$ *be compatible with a sort attachment* $\mathcal{S}$*. If* $\mathcal{R}$ *is confluent on* $\mathcal{T}_\mathcal{S}(\mathcal{F}, \mathcal{V})$ *then* $\mathcal{R}$ *is confluent.*

Proof. Assume that $\mathcal{R}$ is confluent on $\mathcal{T}_\mathcal{S}(\mathcal{F}, \mathcal{V})$. We let $\mathbb{L}$ be the smallest set such that $\mathcal{T}_\mathcal{S}(\mathcal{F}, \mathcal{V}) \subseteq \mathbb{L}$ and $\mathbb{L}$ is closed under replacing variables by holes and vice versa (cf. (L_2)). It is easy to see that $\mathcal{R}$ is layered according to $\mathbb{L}$. (W) and (C_1) follow from the compatibility assumption and Remark 5.61. Also (C_2) is confirmed easily. We show that $\mathcal{R}$ is confluent on terms of rank one. To this end, consider a term $s \in \mathbb{L} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$. The confluence assumption on $\mathcal{T}_\mathcal{S}(\mathcal{F}, \mathcal{V})$ does not immediately apply to s since the variables need not match the type of their context. If s is a variable then s is confluent. Otherwise, there is a term s' in $\mathcal{T}_\mathcal{S}(\mathcal{F}, \mathcal{V})$ that has s as an instance. Because subterms of sort α are interchangeable in many-sorted terms, we may choose s' in such a way that $s'|_p = s'|_q$ if $s'|_p, s'|_q \in \mathcal{V}_\alpha$ for some α and $s|_p = s|_q$. Note that for each p the sort of $s'|_p$ is uniquely determined by s. Because the sets $\mathcal{T}_\alpha(\mathcal{F}, \mathcal{V})$ are pairwise disjoint, any rewrite sequence on $s \in \mathbb{L} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ is mirrored by a rewrite sequence from $s' \in \mathcal{T}_\mathcal{S}(\mathcal{F}, \mathcal{V})$. By assumption, s' is confluent and hence s is confluent as well. We conclude that $\mathcal{R}$ is confluent on terms of rank one and hence confluent by Theorem 5.19. □

5.5 Order-sorted Persistence

In this section we establish order-sorted persistence. Section 5.5.1 introduces order-sorted rewriting, states the main result, and explains how to exploit it for establishing confluence. In Section 5.5.2 we prove the result for left-linear systems before Section 5.5.3 shows that layer systems cannot immediately cover arbitrary TRSs. We refine them such that they become suitable and give an alternative proof for many-sorted persistence (Section 5.5.4) before we finally prove order-sorted persistence in Section 5.5.5. We compare our work with the earlier result from [3] in Section 5.6.1.

5.5.1 Confluence via Order-sorted Persistence

To obtain order-sorted terms, we equip a set of sorts S with a precedence $>$ and modify Definition 5.59 as follows.

Definition 5.63. Let $\mathcal{S}$ be a sort attachment. We define terms of sort α inductively by $\mathcal{T}_\alpha(\mathcal{F},\mathcal{V}) = \mathcal{V}_\alpha \cup \{f(t_1,\dots,t_n) \mid f : \alpha_1 \times \cdots \times \alpha_n \to \alpha,\ t_i \in \mathcal{T}_{\beta_i}(\mathcal{F},\mathcal{V}),\ \alpha_i \geqslant \beta_i,\ \text{and } 1 \leqslant i \leqslant n\}$. The set of all order-sorted terms is $\mathcal{T}_\mathcal{S}(\mathcal{F},\mathcal{V}) = \bigcup_{\alpha\in S} \mathcal{T}_\alpha(\mathcal{F},\mathcal{V})$. A term t is *strictly order-sorted* if $\mathrm{root}(t|_p) : \alpha_1 \times \cdots \times \alpha_n \to \alpha$ and $t|_{pi} \in \mathcal{V}_\beta$ imply $\alpha_i = \beta$, for all $p \in \mathcal{P}os_\mathcal{F}(t)$.

Note that we obtain many-sorted terms by letting $> = \varnothing$. Next we define when a TRS is *compatible* with a sort attachment $\mathcal{S}$ in the order-sorted setting.

Definition 5.64. A TRS $\mathcal{R}$ is *compatible* with a sort attachment $\mathcal{S}$ if each rule $\ell \to r \in \mathcal{R}$ satisfies condition (1), and *strongly compatible* with $\mathcal{S}$ if condition (2) is satisfied as well.

1. If $\ell \in \mathcal{T}_\alpha(\mathcal{F},\mathcal{V})$ and $r \in \mathcal{T}_\beta(\mathcal{F},\mathcal{V})$ then $\alpha \geqslant \beta$ and ℓ is strictly order-sorted.
2. If $r \in \mathcal{V}_\beta$ then β is maximal in S. If $r \notin \mathcal{V}$ then r is strictly order-sorted.

Note that condition (1) ensures that well-typed terms are closed under rewriting. The main result on order-sorted persistence is stated below.

Theorem 5.65. *Let $\mathcal{R}$ be compatible with a sort attachment $\mathcal{S}$. Furthermore assume that $\mathcal{R}$ is left-linear, bounded duplicating, or strongly compatible with $\mathcal{S}$. If $\mathcal{R}$ is confluent on $\mathcal{T}_\mathcal{S}(\mathcal{F},\mathcal{V})$ then it is confluent.*

Theorem 5.65 gives rise to a decomposition result (presented in [3, 4]) based on order-sorted persistence. The decomposition is based on the observation that the sort of a term restricts the rules that can be applied when rewriting it; therefore we can decompose a TRS $\mathcal{R}$ that is compatible with a sort attachment $\mathcal{S}$ into several TRSs $\mathcal{R}_\alpha$ ($\alpha \in S$) each containing the rules applicable to terms of sort α or less. Formally, we define $\trianglerighteq$ on sorts as the smallest transitive relation such that $> \subseteq \trianglerighteq$ and $\alpha \trianglerighteq \alpha_i$ whenever $f : \alpha_1 \times \cdots \times \alpha_n \to \alpha$, and then define $\mathcal{R}_\alpha = \{\ell \to r \mid \ell \to r \in \mathcal{R},\ \ell \in \mathcal{T}_\beta(\mathcal{F},\mathcal{V}),\ \text{and } \alpha \trianglerighteq \beta\}$.

The next example shows that order-sorted persistence is more powerful than many-sorted persistence for decomposing TRSs.

Example 5.66 (adapted from [3])**.** Consider the TRS $\mathcal{R}$ consisting of the rewrite rules

$$(1)\ \mathsf{f}(x,\mathsf{a}) \to \mathsf{g}(x) \quad (2)\ \mathsf{f}(x,\mathsf{f}(x,\mathsf{b})) \to \mathsf{b} \quad (3)\ \mathsf{g}(\mathsf{c}) \to \mathsf{c} \quad (4)\ \mathsf{h}(x) \to \mathsf{h}(\mathsf{g}(x))$$

and the set of sorts $S = \{0,1,2\}$ with $1 \geqslant 0$. Let the sort attachment be given by $\mathsf{a},\mathsf{b} : 1$, $\mathsf{c} : 0$, $\mathsf{f} : 0 \times 1 \to 1$, $\mathsf{g} : 0 \to 0$, $\mathsf{h} : 0 \to 2$, and $x : 0$. It is straightforward to check that $\mathcal{R}$ is consistent with $\mathcal{S}$. In the order-sorted TRS, only rules (1), (2), and (3) can be applied to terms of sort 1 and their reducts, rules (3) and (4) can be applied to terms of sort 2, and only rule (3) can be applied to terms of sort 0. Hence, since $\mathcal{R}_1 = \{(1),(2),(3)\}$ (which is terminating and has no critical pairs), $\mathcal{R}_2 = \{(3),(4)\}$ (which is orthogonal), and $\mathcal{R}_0 = \{(3)\}$ (orthogonal) are confluent, $\mathcal{R}$ is confluent. No such decomposition can be obtained with many-sorted persistence. Consider a *most general* sort attachment making all rules many-sorted: $\mathsf{a},\mathsf{b},\mathsf{c},x : 0$, $\mathsf{f} : 0 \times 0 \to 0$, $\mathsf{g} : 0 \to 0$, and $\mathsf{h} : 0 \to 2$. Since terms of sort 2 can have subterms of sort 0, no decomposition is possible.

The weaker conditions in Definition 5.64 for left-linear TRSs are beneficial.

Example 5.67. Consider the TRS $\mathcal{R}$ consisting of the rewrite rules

$$\mathsf{f}(\mathsf{a}) \to \mathsf{f}(\mathsf{f}(\mathsf{h}(\mathsf{c}))) \qquad \mathsf{g}(\mathsf{b}) \to \mathsf{g}(\mathsf{g}(\mathsf{h}(\mathsf{c}))) \qquad \mathsf{h}(x) \to x$$

and the set of sorts $S = \{0,1,2\}$ with $1,2 \geqslant 0$. Let the sort attachment be given by $\mathsf{a} : 1$, $\mathsf{b} : 2$, $\mathsf{c},x : 0$, $\mathsf{f} : 1 \to 1$, $\mathsf{g} : 2 \to 2$, and $\mathsf{h} : 0 \to 0$, Note that $\mathcal{R}$ is compatible with $\mathcal{S}$. We can decompose $\mathcal{R}$ into the component induced by sort 1: $\mathcal{R}_1 = \{\mathsf{f}(\mathsf{a}) \to \mathsf{f}(\mathsf{f}(\mathsf{h}(\mathsf{c}))), \mathsf{h}(x) \to x\}$, sort 2: $\mathcal{R}_2 = \{\mathsf{g}(\mathsf{b}) \to \mathsf{g}(\mathsf{g}(\mathsf{h}(\mathsf{c}))), \mathsf{h}(x) \to x\}$, and sort 0: $\mathcal{R}_0 = \{\mathsf{h}(x) \to x\}$. If we add the restrictions for non-left-linear systems, the collapsing rule $\mathsf{h}(x) \to x$ enforces $\mathsf{h} : \alpha \to \alpha$ for a maximal sort α. Hence also the arguments of f and g have sort α, and α is greater than or equal to the sort of $\mathsf{a},\mathsf{b},\mathsf{c},\mathsf{f}(x),\mathsf{g}(x)$. So the component induced by α contains all rules.

5.5.2 Order-sorted Persistence for Left-linear Systems

In this section we show that layer systems can establish order-sorted persistence for left-linear TRSs.

Theorem 5.68. *Let $\mathcal{R}$ be compatible with a sort attachment $\mathcal{S}$. If $\mathcal{R}$ is left-linear and confluent on $\mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V})$ then it is confluent.*

Proof. Let $\mathbb{L}$ be the smallest set such that $\mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V}) \subseteq \mathbb{L}$ and $\mathbb{L}$ is closed under (L_2). First we show that $\mathbb{L}$ weakly layers $\mathcal{R}$. In the sequel we call contexts *weakly order-sorted* if they are order-sorted except that arbitrary variables may occur at any position. (These are exactly the elements of $\mathbb{L}$ and weakly order-sorted terms are those in $\mathbb{L} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$.)

Condition (L_1) holds trivially and condition (L_2) holds by assumption. For (L_3) we assume that $L|_p \sqcup N = N'$ with $p \in \mathcal{P}os_{\mathcal{F}}(L)$ is defined. Since $L,N \in \mathbb{L}$ obviously N' is weakly order-sorted and so is $L[N']_p$ since $\mathrm{root}(L|_p) = \mathrm{root}(N')$ and hence $L[N']_p \in \mathbb{L}$. The final condition is (W). So let $s \to_{p,\ell\to r} t$ with $p \in$

$\mathcal{P}os_{\mathcal{F}}(M)$ for the max-top M of s. We have $\mathrm{root}(M|_p) = \mathrm{root}(\ell)$ and hence $M[\ell]_p$ is a layer. Since M is the max-top of s and ℓ is left-linear there is a substitution σ such that $M[\ell\sigma]_p = M$. Hence $M \to_{p,\ell\to r} M[r\sigma]_p$. By compatibility with the sort attachment $\mathcal{S}$ we have $r\sigma \in \mathbb{L}$. Furthermore if α and β are the sorts of ℓ and r then $\alpha \geqslant \beta$ ensures that $M[r\sigma]_p$ is weakly order-sorted and hence a member of $\mathbb{L}$.

Next we show confluence of terms of rank one. To this end let $s \in \mathbb{L} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$. Then there are a term $s' \in \mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V})$ and a variable substitution χ such that $s = s'\chi$. Let $t \mathrel{{}^*_{\mathcal{R}}\!\leftarrow} s \to^*_{\mathcal{R}} u$. By left-linearity of $\mathcal{R}$ there are terms t' and u' with $t = t'\chi$ and $u = u'\chi$ such that $t' \mathrel{{}^*_{\mathcal{R}}\!\leftarrow} s' \to^*_{\mathcal{R}} u'$. The confluence assumption on $\mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V})$ yields $t' \to^*_{\mathcal{R}} v' \mathrel{{}^*_{\mathcal{R}}\!\leftarrow} u'$. Hence $t = t'\chi \to^*_{\mathcal{R}} v'\chi \mathrel{{}^*_{\mathcal{R}}\!\leftarrow} u'\chi = t$. We conclude by Theorem 5.14. □

5.5.3 Variable-restricted Layer Systems

The following example shows that Theorem 5.19 alone cannot establish Theorem 5.65 for TRSs which are neither left-linear nor bounded duplicating.

Example 5.69. Consider the set of sorts $S = \{0, 1, 2, 3, 4\}$, where $2 \geqslant 0$ and $2 \geqslant 1$. The sort attachment $\mathcal{S}$ is given by

$u : 0$	$v : 1$	$\mathsf{f} : 3 \times 3 \to 4$	$\mathsf{h} : 2 \times 2 \times 0 \times 1 \to 3$
$x : 2$	$y : 3$	$\mathsf{g} : 3 \to 3$	$\mathsf{a}, \mathsf{b} : 4$

and the TRS $\mathcal{R}$ consists of the rules

$$\mathsf{f}(y, y) \to \mathsf{a} \qquad \mathsf{f}(y, \mathsf{g}(y)) \to \mathsf{b} \qquad \mathsf{h}(x, x, u, v) \to \mathsf{g}(\mathsf{h}(u, v, u, v))$$

Then $\mathcal{R}$ is confluent on $\mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V})$ because it is locally confluent and terminating on order-sorted terms, noting that u and v never represent equal terms due to sort constraints. However, if we take $\mathbb{L}$ to be the closure of $\mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V})$ under (L_2) then with $t = \mathsf{h}(z, z, z, z)$ the term $\mathsf{f}(t, t)$ is not confluent because $\mathsf{a} \leftarrow \mathsf{f}(t, t) \to \mathsf{f}(t, \mathsf{g}(t)) \to \mathsf{b}$. Note that $\mathsf{f}(t, t)$ is not order-sorted but contained in $\mathbb{L}$. Furthermore, observe that $\mathbb{L}$ is the smallest layer system layering $\mathcal{R}$ that contains $\mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V})$.

The above example does not contradict Theorem 5.65 since $\mathcal{R}$ is not strongly compatible with $\mathcal{S}$; the right-hand sides of $\mathcal{R}$ are not strictly order-sorted although $\mathcal{R}$ is neither left-linear nor bounded duplicating. In particular we have an infinite reduction

$$\begin{aligned}\mathsf{h}(z, z, \Diamond(z), \Diamond(z)) &\to_{\mathcal{R}} \mathsf{g}(\mathsf{h}(\Diamond(z), \Diamond(z), \Diamond(z), \Diamond(z))) \\ &\to^+_{\Diamond(x)\to x} \mathsf{g}(\mathsf{h}(z, z, \Diamond(z), \Diamond(z))) \to_{\mathcal{R}} \cdots\end{aligned}$$

The problem is that layer systems allow to replace variables by variables of a different sort and hence contain terms which are not order-sorted, enabling new rewrite steps (which does never happen in the many-sorted case nor for left-linear systems in the order-sorted setting). Since $\mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V}) \subsetneq \mathbb{L} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$, we have to study when confluence on $\mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V})$ implies confluence on $\mathbb{L} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$

in order to apply Theorem 5.19. Instead of proving the missing implication directly, we again pursue a general approach. To this end we relax condition (L_2) such that variables need not be replaced by variables of different sort, to enable the representation of $\mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V})$ as $\mathbb{L} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$, where $\mathbb{L}$ satisfies the following refined notion of layer systems.

Definition 5.70. Recall the conditions from Definition 5.3. We introduce the following condition:

(L_2') If $C[x]_p \in \mathbb{L}$ then $C[\Box]_p \in \mathbb{L}$. If $C[\Box]_p \in \mathbb{L}$ then $\{x \in \mathcal{V} \mid C[x]_p \in \mathbb{L}\}$ is an infinite set.

We call a set $\mathbb{L} \subseteq \mathcal{C}(\mathcal{F},\mathcal{V})$ a *variable-restricted layer system* if it satisfies the conditions (L_1), (L_2'), and (L_3). Analogously, a variable-restricted layer system *weakly layers* $\mathcal{R}$ if (W) is satisfied and *layers* $\mathcal{R}$ if (W), (C_1), and (C_2) are satisfied.

To distinguish between variable-restricted and (unrestricted) layer systems we denote the former by $\mathbb{L}'$ in the future. Note that (L_2) implies (L_2'), hence any layer system is also a variable-restricted layer system. Furthermore, for each variable-restricted layer system $\mathbb{L}'$ there is a corresponding (unrestricted) layer system $\mathbb{L}'_{\mathbb{L}} = \mathbb{L}' \cup \{C[x]_p \mid C[\Box]_p \in \mathbb{L}' \text{ and } x \in \mathcal{V}\}$. Obviously $\mathbb{L}' \subseteq \mathbb{L}'_{\mathbb{L}}$.

With the new condition (L_2') it is now possible to adequately represent $\mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V})$ by a variable-restricted layer system.

Example 5.71 (Example 5.69 revisited)**.** To obtain a variable-restricted layer system, let $\mathbb{L}'$ be the smallest set such that $\mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V}) \subseteq \mathbb{L}'$ and $\mathbb{L}'$ is closed under replacing variables by holes. Then it satisfies (L_2'). Note that $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V}) = \mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V})$ and hence $t = \mathsf{h}(z,z,z,z) \notin \mathbb{L}'$ and thus $\mathsf{f}(t,t) \notin \mathbb{L}'$.

For a weakly layered TRS the reduct of a rank one term again is a rank one term.

Lemma 5.72. *Let $\mathbb{L}'$ be a variable-restricted layer system that weakly layers a TRS $\mathcal{R}$. Then $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ is closed under rewriting by $\mathcal{R}$.*

Proof. Let $t \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ and $t \to_{\mathcal{R}} u$. Note that t is its own max-top. By (W), its reduct u is a layer and hence $u \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$. □

In the remainder of this section we show the analogues of Theorems 5.14, 5.16, and 5.19 for variable-restricted layer systems (cf. Corollary 5.86).

The case of left-linear systems is straightforward.

Lemma 5.73. *Let $\mathbb{L}'$ be a variable-restricted layer system that weakly layers a left-linear TRS $\mathcal{R}$. If $\mathcal{R}$ is confluent on $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ then $\mathcal{R}$ is confluent on $\mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$.*

Proof. Let $s \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$. By ($L_2$) and ($L_2'$) there are a term $s' \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ and a variable substitution χ such that $s'\chi = s$. Now consider rewrite sequences $t \mathrel{{}^*_{\mathcal{R}}\!\leftarrow} s \to^*_{\mathcal{R}} u$. Thanks to left-linearity, there are terms t' and u' with $t'\chi = t$, $u'\chi = u$, and $t' \mathrel{{}^*_{\mathcal{R}}\!\leftarrow} s' \to^*_{\mathcal{R}} u'$. By repeated application of Lemma 5.72, t', u' as well as all intermediate terms are elements of $\mathbb{L}' \cap$

$\mathcal{T}(\mathcal{F},\mathcal{V})$. From the assumption we obtain a valley $t' \to_{\mathcal{R}}^* v' \,{}_{\mathcal{R}}^*\!\leftarrow u'$, inducing a valley $t = t'\chi \to_{\mathcal{R}}^* v'\chi \,{}_{\mathcal{R}}^*\!\leftarrow u'\chi = u$. Note that $v'\chi \in \mathbb{L}'_{\mathbb{L}}$ (by Lemma 5.10) and obviously $v'\chi \in \mathcal{T}(\mathcal{F},\mathcal{V})$. □

To prepare for a result concerning bounded duplicating TRSs, we generalize bounded duplication to weakly bounded duplication, which turns out to be more suitable for the proof of Lemma 5.76 below.

Definition 5.74. We call $\mathcal{R}$ *weakly bounded duplicating* if $\{\top \to \bot\}/\mathcal{R}$ is terminating for fresh constants $\top$ and $\bot$.

Lemma 5.75. *Any bounded duplicating TRS is weakly bounded duplicating.*

Proof. Assume that $\mathcal{R}$ is not weakly bounded duplicating. So there exists an infinite rewrite sequence $t_0 \to t_1 \to \cdots$ in $\mathcal{R} \cup \{\top \to \bot\}$ that contains infinitely many applications of the rule $\top \to \bot$. Let t'_i be obtained from t_i by replacing all occurrences of $\top$ by $\Diamond(\bot)$. Since $\top$ does not appear in the rules of $\mathcal{R}$, we obtain an infinite rewrite sequence $t'_0 \to t'_1 \to \cdots$ in $\mathcal{R} \cup \{\Diamond(x) \to x\}$ with infinitely many applications of the instance $\Diamond(\bot) \to \bot$ of $\Diamond(x) \to x$. Hence $\mathcal{R}$ is not bounded duplicating. □

To see that weakly bounded duplication generalizes bounded duplication, consider the TRS $\mathcal{R}$ consisting of the single rule $\mathsf{f}(\mathsf{a},x) \to \mathsf{f}(x,x)$, which is not bounded duplicating since $\mathsf{f}(\mathsf{a},\Diamond(\mathsf{a})) \to_{\mathcal{R}} \mathsf{f}(\Diamond(\mathsf{a}),\Diamond(\mathsf{a})) \to_{\Diamond(x)\to x} \mathsf{f}(\mathsf{a},\Diamond(\mathsf{a})) \to_{\mathcal{R}} \cdots$, but weakly bounded duplicating.

Below we will establish the following two lemmata.

Lemma 5.76. *Let $\mathbb{L}'$ be a variable-restricted layer system that weakly layers a weakly bounded duplicating TRS $\mathcal{R}$. If $\mathcal{R}$ is confluent on $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ then $\mathcal{R}$ is confluent on $\mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$.*

Lemma 5.77. *Let $\mathbb{L}'$ be a variable-restricted layer system that layers a TRS $\mathcal{R}$. If $\mathcal{R}$ is confluent on $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ then $\mathcal{R}$ is confluent on $\mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$.*

For both proofs we are given a variable-restricted layer system $\mathbb{L}'$ that weakly layers a TRS $\mathcal{R}$. We fix an initial term $s \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ and show that it is confluent. Since $\mathbb{L}' \subseteq \mathbb{L}'_{\mathbb{L}}$ the confluence assumption on $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ may not apply to s. To overcome this problem we use (L_2) and (L'_2) to construct a term $s' \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ and a variable substitution χ such that $s = s'\chi$ and fix a well-order $\gg$ on $\mathcal{V}ar(s')$. We extend $\gg$ to terms by closing it under contexts and transitivity.

Let $s \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$, $s' \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$, and χ with $s = s'\chi$ be fixed.

Definition 5.78. A term $t' \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ is a *representative* of $t \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ if $t = t'\chi$ and $\mathcal{V}ar(t') \subseteq \mathcal{V}ar(s')$. A representative t' of t is called *minimal* if it is minimal with respect to $\gg$.

Note that s' is a representative of s. Before proving key properties for representatives we show how they help to avoid the situation of Example 5.69.

Example 5.79 (Example 5.69 revisited)**.** Consider the variables with sorts

$$x_1, x_2, x_5, x_6 : 2 \qquad x_3, x_7 : 0 \qquad x_4, x_8 : 1$$

and order $x_8 \gg x_7 \gg \cdots \gg x_1$. The term $s = \mathsf{f}(t, t) \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ has the representative $s' = \mathsf{f}(\mathsf{h}(x_1, x_2, x_3, x_4), \mathsf{h}(x_5, x_6, x_7, x_8)) \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ and the (unique) minimal representative $\hat{s} = \mathsf{f}(\hat{t}, \hat{t}) \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ where $\hat{t} = \mathsf{h}(x_1, x_1, x_3, x_4)$. The peak $\mathsf{a} \leftarrow \mathsf{f}(t, t) \to \mathsf{f}(t, \mathsf{g}(t)) \to \mathsf{b}$ in $\mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ is simulated by

$$\mathsf{a} \leftarrow \mathsf{f}(\hat{t}, \hat{t}) \to \mathsf{f}(\hat{t}, \mathsf{g}(\mathsf{h}(x_3, x_4, x_3, x_4))) \gg \mathsf{f}(\hat{t}, \mathsf{g}(\hat{t})) \to \mathsf{b}$$

in $\mathbb{L}' \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$. Note that the $\gg$-step replaces $\mathsf{f}(\hat{t}, \mathsf{g}(\mathsf{h}(x_3, x_4, x_3, x_4)))$ by the least representative $\mathsf{f}(\hat{t}, \mathsf{g}(\hat{t}))$ of $\mathsf{f}(t, \mathsf{g}(t))$.

The key operation on representatives and related terms is copying variables between them, as justified by the following lemma.

Lemma 5.80. *Let $L, N \in \mathbb{L}'$ be layers with $L_\Box = N_\Box$. If $p \in \mathcal{P}os_{\mathcal{V}_\Box}(L)$ then $L[N|_p]_p \in \mathbb{L}'$.*

Proof. If $p = \epsilon$ then the claim is trivial. Otherwise, let $L' = L[\Box]_p$ and $N' = N[\Box]_{q \in \mathcal{P}os_{\mathcal{V}_\Box}(L) \setminus \{p\}}$. We have $L', N' \in \mathbb{L}'$ by applications of property (L'_2) and $L[N|_p]_p = L' \sqcup N'$ by assumption. Property (L_3) yields the desired $L[N|_p]_p \in \mathbb{L}'$. □

The next lemma establishes that the minimal representative (if it exists) is unique, justifying the name *least* representative. The proof makes the construction in Example 5.79 explicit and is illustrated by Example 5.82.

Lemma 5.81. *If $t \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F}, \mathcal{V})$ has a representative then it has a least representative.*

Proof. We have to show the existence and uniqueness of a minimal representative of t. From a representative t' we obtain $t'_\Box \in \mathbb{L}'$ using (L'_2) repeatedly. Consider $V_p = \{x \in \mathcal{V}ar(s') \mid \chi(x) = t|_p \text{ and } t'_\Box[x]_p \in \mathbb{L}'\}$ for each $p \in \mathcal{P}os_{\mathcal{V}}(t')$. Note that $t'|_p \in V_p$ because we can insert the variable $t'|_p$ into $t'_\Box$ at position p by Lemma 5.80 to obtain a layer in $\mathbb{L}'$. Hence V_p is non-empty. Since it is also finite it has a minimum element $\min(V_p)$ with respect to $\gg$. Let $\dot{t} = t'_\Box[\min(V_p)]_{p \in \mathcal{P}os_{\mathcal{V}}(t')}$. We have $\dot{t} \in \mathbb{L}'$ by (L'_2) and the definition of V_p. Clearly $\dot{t} \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ and $\mathcal{V}ar(\dot{t}) \subseteq \mathcal{V}ar(s')$ because all holes are replaced by some variable from $\mathcal{V}ar(s')$. Moreover, $\dot{t}\chi = t$ by construction, in particular the definition of V_p. It follows that $\dot{t}$ is a representative of t. Note that $\dot{t}$ does not depend on the choice of t' because $t'_\Box = t_\Box$. Therefore, $t' \gg^= \dot{t}$ for any representative t' of t, which makes $\dot{t}$ the least representative of t. □

Example 5.82 (Example 5.79 revisited)**.** Let $s = \mathsf{f}(\mathsf{h}(z, z, z, z), \mathsf{h}(z, z, z, z))$ and $s' = \mathsf{f}(\mathsf{h}(x_1, x_2, x_3, x_4), \mathsf{h}(x_5, x_6, x_7, x_8))$ with $\chi(x_i) = z$ for all $1 \leqslant i \leqslant 8$. Then $s'_\Box = \mathsf{f}(\mathsf{h}(\Box, \Box, \Box, \Box), \mathsf{h}(\Box, \Box, \Box, \Box))$. Since $V_{11} = V_{12} = V_{21} = V_{22} = \{x_1, \ldots, x_8\}$, $V_{13} = V_{23} = \{x_3, x_7\}$, and $V_{14} = V_{24} = \{x_4, x_8\}$ we obtain $\dot{s} = \mathsf{f}(\mathsf{h}(x_1, x_1, x_3, x_4), \mathsf{h}(x_1, x_1, x_3, x_4))$.

We denote the least representative term of a representable term $t \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ by $\hat{t} \in \mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$. The following lemma states that a rewrite step performed on a term in $\mathbb{L}'_{\mathbb{L}}$ can be mirrored on its least representative in $\mathbb{L}'$. Recall that in Example 5.79 the representative s' is a normal form but the step from s can be mirrored on $\hat{s}$.

Lemma 5.83. *Let $t, u \in \mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ with $t \to_{\mathcal{R}} u$ such that $\hat{t}$ exists.*

1. *If $\mathbb{L}'$ weakly layers $\mathcal{R}$ then $\hat{t} \to_{\mathcal{R}} u'$ for some representative u' of u.*
2. *If $\mathbb{L}'$ layers $\mathcal{R}$ then $u' = \hat{u}$ or $u' \in \mathcal{V}$ in* (1).

Proof.

1. Assume that $\hat{t}$ is the least representative of t and let $t \to_{p,\ell\to r} u$. We obtain a context $C \in \mathbb{L}'$ by replacing all variables in t by $\Box$. By Lemma 5.11, there is a term c with $C = c\sigma_{\Box}$ and $\ell \leqslant c|_p$. To ensure $c \leqslant \hat{t}$ we need to show $\hat{t}|_q = \hat{t}|_r$ for all $x \in \mathcal{V}ar(c)$ and $q, r \in \mathcal{P}os_x(c)$. To that end, fix x and let $P = \mathcal{P}os_x(c)$. For each $q \in P$, $\hat{t}|_q$ is a variable. Let $y = \min\{\hat{t}|_q \mid q \in P\}$. We will show that $\hat{t}|_q = y$ for all $q \in P$. Consider the max-top $M \in \mathbb{L}'$ of $C[y,\dots,y]$. Note that $c \leqslant C[y,\dots,y]$, so that $\ell \leqslant C[y,\dots,y]|_p$. From condition (W) we obtain $\ell \leqslant M|_p$ and thus $c \leqslant M$ by Lemma 5.11(2) since $C \sqsubseteq M$. By construction $\hat{t}|_q = y$ for some $q \in P$. Since $C[y]_q$ is a layer by Lemma 5.80, $M \sqcup C[y]_q$ is a layer according to (L_3). Because M is the max-top of $C[y,\dots,y]$, $M \sqcup C[y]_q = M$ and thus $M|_q = y$. It follows that $M|_q = y$ for all $q \in P$, since otherwise M would fail to be an instance of c. Repeated applications of Lemma 5.80 yields $t' = \hat{t}[y]_{q\in P} \in \mathbb{L}'$. We have $t' = \hat{t}$ by the choice of y and the minimality of $\hat{t}$. We conclude that $c \leqslant \hat{t}$ and hence $\ell \leqslant \hat{t}|_p$, which induces a rewrite step $\hat{t} \to_{p,\ell\to r} u'$ as claimed. The term u' is a representative of u because $u'\chi = u$, $u' \in \mathbb{L}'$ by Lemma 5.72, and rewriting does not introduce variables.

2. Assume that u' is not a least representative of u. We have $u' \gg \hat{u}$, so there is a position $q \in \mathcal{P}os_{\mathcal{V}}(u)$ with $z = u'|_q \gg \hat{u}|_q = y$. Let $C = c\sigma_{\Box}$ as in the proof of part (1). There is a rewrite step $c \to_{p,\ell\to r} d$ for some term d and $C \to_{p,\ell\to r} D = d\sigma_{\Box}$. Let $M \in \mathbb{L}'$ and $L \in \mathbb{L}'$ be the max-tops of $C_y = C[y,\dots,y]$ and $D_y = D[y,\dots,y]$. Note that $C_y \to_{p,\ell\to r} D_y$, which implies $M \to_{p,\ell\to r} L$ by (C_1) except when $M \to_{p,\ell\to r} \Box$. In the latter case r and thus also u' is a variable, and we are done. So assume $M \to_{p,\ell\to r} L$. Consider the variable $x = d|_q$. We must have $L|_q = y$ because otherwise we could copy $\hat{u}|_q = y$ to L by Lemma 5.80. The term $\hat{t}$ and the context M are instances of c and so there are substitutions $\sigma_{\hat{t}}$ and σ_M such that $c\sigma_{\hat{t}} = \hat{t}$ and $c\sigma_M = M$. We have $\sigma_{\hat{t}}(x) = u'|_q = z$ and $\sigma_M(x) = y$ because $d\sigma_M = L$. Since $x \in \mathcal{V}ar(d)$ and $c \to_{\mathcal{R}} d$, the set $\mathcal{P}os_x(c)$ is non-empty. Let $q' \in \mathcal{P}os_x(c)$. The layer $C[y]_{q'} \in \mathbb{L}'$ can be obtained by copying $M|_{q'} = y$ to C using Lemma 5.80. Since $\hat{t}|_{q'} = \sigma_{\hat{t}}(x) = z$, we obtain $\hat{t} \gg \hat{t}[y]_{q'} \in \mathbb{L}'$. The term $\hat{t}[y]_{q'}$ is a representative of t because $\chi(y) = \chi(z)$ (recall that $u = u'\chi = \hat{u}\chi$). Hence we obtained a contradiction with the minimality of $\hat{t}$. $\Box$

□

The following lemma shows that instead of adding a single rule $\top \to \bot$, we can extend a weakly bounded duplicating TRS with any terminating ARS, where the objects are regarded as fresh constants, and still obtain relative termination. The induced well-founded order will be used in the proof of Lemma 5.76.

Lemma 5.84. *Let $\mathcal{R}$ be a weakly bounded duplicating TRS and $\mathcal{A}$ be a terminating ARS. If $\mathcal{R}$ and $\mathcal{A}$ share no constants then $\mathcal{A}$ is terminating relative to $\mathcal{R}$.*

Proof. We use reduction pairs for this proof, which are pairs consisting of a quasi-order $\geqslant$ and a well-founded strict order $>$ that are compatible: $\geqslant \cdot > \cdot \geqslant \subseteq >$. Reduction pairs give rise to a multiset extension in a straightforward way (e.g., the definitions of $>_{\mathrm{gms}}$ and $\geqslant_{\mathrm{gms}}$ in [71]). We denote the objects in $\mathcal{A}$ by $\mathcal{O}$. Let $\mathcal{F}$ be the signature of $\mathcal{R}$. From the termination of $\mathcal{A}$ we obtain a well-founded order $>$ on $\mathcal{O}$ such that $\mathcal{A} \subseteq >$. For each $\alpha \in \mathcal{O}$ define a map π_α from $\mathcal{T}(\mathcal{F} \cup \mathcal{O}, \mathcal{V})$ to $\mathcal{T}(\mathcal{F} \cup \{\top, \bot\}, \mathcal{V})$ as follows:

$$\pi_\alpha(t) = \begin{cases} \top & \text{if } t = \alpha \\ \bot & \text{if } t \in \mathcal{O} \setminus \{\alpha\} \\ f(\pi_\alpha(t_1), \ldots, \pi_\alpha(t_n)) & \text{if } t = f(t_1, \ldots, t_n) \text{ with } f \in \mathcal{F} \\ t & \text{if } t \in \mathcal{V} \end{cases}$$

We measure terms by the set $\#t = \{(\alpha, \pi_\alpha(t)) \mid \alpha \in \mathcal{F}un(t) \cap \mathcal{O}\}$. The measures of two terms are compared by the multiset extension of the lexicographic product of the precedence $>$ on $\mathcal{O}$ and the reduction pair consisting of the well-founded (by the weakly bounded termination assumption) order $\to^+_{\{\top\to\bot\}/\mathcal{R}}$ and the compatible quasi-order $\to^*_{\mathcal{R}}$. Each application of a rule $\alpha \to \beta$ from $\mathcal{A}$ decreases the component associated with α in $\#t$ and introduces or modifies a component associated with β in $\#t$, giving rise to a decrease in the strict part of the multiset extension. Moreover, if $t \to_{\mathcal{R}} u$ then $\pi_\alpha(t) \to_{\mathcal{R}} \pi_\alpha(u)$, for all $\alpha \in \mathcal{O}$. Hence the terms are related by the non-strict part of the multiset extension. It follows that $\mathcal{A}$ is terminating relative to $\mathcal{R}$. □

Proof of Lemma 5.76. To show confluence of s we introduce a relation $\blacktriangleright\triangleright$ that allows to map an $\mathcal{R}$-peak from s to a $\blacktriangleright\triangleright$-peak. Afterwards we show confluence of $\blacktriangleright\triangleright$ and conclude by $\blacktriangleright\triangleright \subseteq \to^*_{\mathcal{R}}$.

We write $t \blacktriangleright\triangleright_{t'_0} u$ if $t'_0 \to^*_{\mathcal{R}} \hat{t}$ and $s \to^*_{\mathcal{R}} t \to^*_{\mathcal{R}} u$ such that $t \to^*_{\mathcal{R}} u$ is mirrored by $\hat{t} \to^*_{\mathcal{R}} u'$ with $u = u'\chi$. Labels are compared using the order $> := \to^+_{\gg/\mathcal{R}}$, which is well-founded according to Lemma 5.84 applied to the ARS $(\mathcal{V}ar(s'), \gg)$, where we regard the elements of $\mathcal{V}ar(s')$ as constants for this purpose.

First we show that a peak consisting of $\mathcal{R}$-steps can be represented as a peak of $\blacktriangleright\triangleright$-steps. To this end we claim that $t \blacktriangleright\triangleright_{\hat{t}} u$ whenever $s \to^*_{\mathcal{R}} t \to_{\mathcal{R}} u$. To show the claim, note that s has a least representative by Lemma 5.81, and that by Lemmata 5.83(1) and Lemma 5.81 each immediate successor of a term with a least representative also has a least representative. Therefore, t has a least representative, and we conclude by another application of Lemma 5.83(1).

Next we establish that $\blacktriangleright\!\triangleright$ is locally decreasing and hence confluent by Theorem 3.21. Consider a local peak $u \;{}_{t'_0}\!\triangleleft\!\blacktriangleleft\; t \;\blacktriangleright\!\triangleright_{t'_1}\; v$. By definition of $\blacktriangleright\!\triangleright$ there are representatives u' and v' of u and v such that $u' \;{}^*_{\mathcal{R}}\!\leftarrow \hat{t} \rightarrow^*_{\mathcal{R}} v'$. We obtain $u' \rightarrow^*_{\mathcal{R}} w' \;{}^*_{\mathcal{R}}\!\leftarrow v'$ from the confluence assumption on $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$. Consider the sequence $u' \rightarrow^*_{\mathcal{R}} w'$. If $u' = \hat{u}$ then $u \;\blacktriangleright\!\triangleright_{t'_1}\; w'\chi$, noting that $t'_1 \rightarrow^*_{\mathcal{R}} \hat{t} \rightarrow^*_{\mathcal{R}} u'$. Otherwise, there is a rewrite sequence $u = u'\chi = u_1 \rightarrow_{\mathcal{R}} \cdots \rightarrow_{\mathcal{R}} u_n = w'\chi = w$, such that $u' \gg \hat{u} \rightarrow^*_{\gg\cup\mathcal{R}} \hat{u}_i$ and thus $u' > \hat{u}_i$ for all $1 \leqslant i \leqslant n$. Hence we obtain $u \;\blacktriangleright\!\triangleright^*_{\vee t'_1}\; w$ by repeated use of the above claim. Analogously, we obtain $v \;\blacktriangleright\!\triangleright_{t'_0}\; w$ or $v \;\blacktriangleright\!\triangleright^*_{\vee t'_0}\; w$. The proof is concluded by the obvious observation that $\blacktriangleright\!\triangleright \subseteq \rightarrow^*_{\mathcal{R}}$. □

Proof of Lemma 5.77. Consider a peak $t \;{}^*_{\mathcal{R}}\!\leftarrow s \rightarrow^*_{\mathcal{R}} u$. Obviously s has a representative and hence also a least representative $\hat{s}$ by Lemma 5.81. Using Lemma 5.83 repeatedly we obtain a peak $t' \;{}^*_{\mathcal{R}}\!\leftarrow \hat{s} \rightarrow^*_{\mathcal{R}} u'$, noting that all reducts of $\hat{s}$ are least representatives of the corresponding reducts of s or variables, but since variables are normal forms the latter can only happen in the last step. From the confluence assumption on $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ we obtain $t' \rightarrow^*_{\mathcal{R}} v' \;{}^*_{\mathcal{R}}\!\leftarrow u'$. Applying the variable substitution χ yields $t = t'\chi \rightarrow^*_{\mathcal{R}} v'\chi \;{}^*_{\mathcal{R}}\!\leftarrow u'\chi = u$ on $\mathbb{L}'_{\mathbb{L}} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$. □

Lemma 5.85. *If a TRS is (weakly) layered according to a variable-restricted layer system then it is (weakly) layered according to the corresponding (unrestricted) layer system.*

Proof. The result for weakly layered TRSs is obvious. The result for layered TRSs follows from Lemma 5.83. □ □

Corollary 5.86. *The statements of Theorems 5.14, 5.16, and 5.19 remain true when based on a variable-restricted layer system.*

Proof. In case of a left-linear TRSs we conclude by Theorem 5.14 and Lemmata 5.73 and 5.85. For bounded duplicating TRSs we use Theorem 5.16 and Lemmata 5.75, 5.76, and 5.85. For TRSs that are layered according to a variable-restricted layer system we use Theorem 5.19 and Lemmata 5.77 and 5.85. □

5.5.4 Many-sorted Persistence by Variable-restricted Layer Systems

We demonstrate the usefulness of variable-restricted layer systems by the following alternative proof of Theorem 5.62, which avoids the complication of establishing confluence on $\mathbb{L} \cap \mathcal{T}(\mathcal{F},\mathcal{V})$.

Proof of Theorem 5.62. Assume that $\mathcal{R}$ is confluent on $\mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V})$. We let $\mathbb{L}'$ be the smallest set such that $\mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V}) \subseteq \mathbb{L}'$ and $\mathbb{L}'$ is closed under replacing variables by holes. So $\mathbb{L}'$ trivially satisfies (L'_2). Hence $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V}) = \mathcal{T}_{\mathcal{S}}(\mathcal{F},\mathcal{V})$ and thus $\mathcal{R}$ is confluent on $\mathbb{L}' \cap \mathcal{T}(\mathcal{F},\mathcal{V})$ by the assumption. It is easy to see that $\mathbb{L}'$ is a variable-restricted layer system layering $\mathcal{R}$; conditions (W) and (C_1) follow from the compatibility assumption. Therefore $\mathcal{R}$ is confluent by Corollary 5.86. □

5.5.5 Order-sorted Persistence by Variable-restricted Layer Systems

In this section we prove the main result on order-sorted persistence.

Proof of Theorem 5.65. Assume that $\mathcal{R}$ is compatible with $\mathcal{S}$. To define layers as order-sorted terms, we add a fresh, minimum sort $\bot$ with $\Box : \bot$ and require that no variable has sort $\bot$. The set $\mathbb{L}' := \mathcal{T}_{\mathcal{S}\cup\{\bot\}}(\mathcal{F} \cup \{\Box\}, \mathcal{V})$ is a variable-restricted layer system that satisfies (C_2).

We show that $\mathbb{L}'$ satisfies condition (W). So let M be the max-top of s, $p \in \mathcal{P}os_{\mathcal{F}}(M)$, and $s \to_{p,\ell\to r} t$. Because ℓ is order-sorted, $\mathcal{P}os(\ell) \subseteq \mathcal{P}os(M|_p)$. We claim that $\ell \leqslant M|_p$. If $\ell|_q = \ell|_{q'} \in \mathcal{V}_\alpha$ then $M|_{pq} = M|_{pq'} \in \mathcal{T}_{\alpha'}(\mathcal{F} \cup \{\Box\}, \mathcal{V})$ for some α' with $\alpha \geqslant \alpha'$, due to the fact that ℓ is strictly order-sorted. Let σ be a substitution such that $\ell\sigma = M|_p$. Using the compatibility condition (of Definition 5.63), we readily obtain $L = M[r\sigma]_p \in \mathbb{L}'$.

Next we show that if $\mathcal{R}$ is strongly compatible with $\mathcal{S}$, then condition (C_1) holds. So assume that $\mathcal{R}$ is neither left-linear nor bounded duplicating and $L \neq \Box$. We show that L is the max-top of t, Let L' be the max-top of t. First of all, if r is not a variable and $\ell|_q = r|_{q'} \in \mathcal{V}_\alpha$ then $L'|_{pq'} = M|_{pq} = L|_{pq'}$ because ℓ and r are strictly order-sorted. This implies $L = L'$. Next suppose that $r = x \in \mathcal{V}_\beta$. Let p' be the position directly above p and let $\mathrm{root}(L|_{p'}) : \beta_1 \times \cdots \times \beta_n \to \beta'$. We have $p = p'i$ for some $1 \leqslant i \leqslant n$. We claim that $\beta_i = \beta$. Let α be the sort of ℓ. We have $\alpha \geqslant \beta$ and $\beta_i \geqslant \alpha$. According to the second compatibility condition, β is maximal in S and thus $\beta = \alpha = \beta_i$. It follows that $L'|_p = M|_{pq} = L|_p$ for any $q \in \mathcal{P}os_x(\ell)$.

Note that $\mathbb{L}' \cap \mathcal{T}(\mathcal{F}, \mathcal{V}) = \mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V}) = \mathbb{L}' \cap \mathcal{T}_{\mathcal{S}}(\mathcal{F}, \mathcal{V})$. The proof is concluded with an appeal to Corollary 5.86. □

5.6 Related Work

As we already mentioned in the introduction, modularity of term rewrite systems has been reproved several times. A number of related results have been proved by adapting the proof of [43] and there have been several previous attempts to make the result more reusable. Ohlebusch [55] casts the modularity result in terms of a collapsing reduction $\to_c$, and shows that for composable TRSs, confluence is modular if $\to_c$ is normalizing. Toyama's theorem arises as a special case. Kahrs [39] proposes an abstract framework, based on so-called *pre-confluences* and *context selectors* constructed from pre-confluences. The latter can be seen as a precursor of layer systems. In particular, the selection of max-tops gives rise to a (proper) context selector. However, the notion of pre-confluences is geared towards the uncurrying application, and too restrictive to encompass modularity of confluence [40]. A third approach to abstraction is taken by Lüth [49]. In this work, modularity of confluence is proved using category theory, using the fact that terms can conveniently be modeled by a monad. Unfortunately, the development is flawed, and only applies to TRSs over unary function symbols and constants.[1]

[1] The paper claims that for any TRS Θ, the monad T_Θ is *strongly finitary*, which implies that it preserves coequalizers. This is not true in general. As an example, let $\star$ be the trivial

In the remainder of this section we discuss specific issues, starting with a comparison of our result on order-sorted persistence to [3] in Section 5.6.1. In Section 5.6.2 we reflect on the differences between [43] and [63] which correspond to changes from the earlier conference paper [22] to the present chapter. Finally we elaborate the constructivity claim in Section 5.6.3.

5.6.1 Order-sorted Persistence

In this section we compare our result from Section 5.5 to the main result of [3], which can be stated as follows.

Definition 5.87. A sort attachment $\mathcal{S}$ is *compatible** with a TRS $\mathcal{R}$ if condition $(\star)$ is satisfied for each rewrite rule $\ell \to r \in \mathcal{R}$:

($\star$) If $\ell \in \mathcal{T}_\alpha(\mathcal{F}, \mathcal{V})$ and $r \in \mathcal{T}_\beta(\mathcal{F}, \mathcal{V})$ then $\alpha \geqslant \beta$ and ℓ, r are strictly order-sorted.

The main claim in [3] is that Theorem 5.65 holds for compatible* systems. We show that this is incorrect.

Example 5.88. We use $\{0, 1, 2, 3\}$ as sorts where $1 \geqslant 0$ and sort attachment $\mathcal{S}$

$$\begin{array}{lllll} x : 0 & \mathsf{f} : 0 \to 2 & \mathsf{h} : 1 \times 0 \to 2 & \mathsf{e} : 0 \to 1 & \mathsf{c} : 1 \\ y : 2 & \mathsf{g} : 2 \to 2 & \mathsf{i} : 2 \times 2 \to 3 & \mathsf{a}, \mathsf{b} : 3 & \end{array}$$

Consider the TRS $\mathcal{R}$ consisting of the rules

$$\mathsf{f}(x) \to \mathsf{h}(\mathsf{e}(x), x) \quad \mathsf{h}(\mathsf{c}, x) \to \mathsf{g}(\mathsf{f}(x)) \quad \mathsf{e}(x) \to x \quad \mathsf{i}(y, y) \to \mathsf{a} \quad \mathsf{i}(y, \mathsf{g}(y)) \to \mathsf{b}$$

This TRS is compatible* with $\mathcal{S}$. It is locally confluent and terminating and thus confluent on order-sorted terms (note that x may not be instantiated by c due to the sort constraints). It is not confluent on arbitrary terms because

$$\mathsf{a} \leftarrow \mathsf{i}(\mathsf{f}(\mathsf{c}), \mathsf{f}(\mathsf{c})) \to^* \mathsf{i}(\mathsf{f}(\mathsf{c}), \mathsf{g}(\mathsf{f}(\mathsf{c}))) \to \mathsf{b}$$

Note that any compatible* TRS is strongly compatible (cf. Definition 5.64), unless it is neither left-linear nor bounded duplicating, and contains a collapsing rule. Indeed the TRS $\mathcal{R}$ of Example 5.88 has all these features. Ultimately, the culprit is the collapsing rule $\mathsf{e}(x) \to x$, causing fusion from above (cf. Figure 5.3). This case is not considered in the proof of [3, Proposition 3.9]. Definition 5.64 takes care of the problem with collapsing rules in Definition 5.87. Furthermore, it puts fewer constraints on the right-hand sides in case of left-linear or bounded duplicating systems, which is beneficial (cf. Example 5.67).

category and consider the coequalizer $Q : \star + \star \to \star$ of the injections $\iota_1, \iota_2 : \star \to \star + \star$. Furthermore let $\Theta = \{\mathsf{f}(x, x) \to x\}$. Then $T_\Theta(Q)$ equates $\mathsf{f}(\acute{}\iota_1\star, \acute{}\iota_1\star)$ and $\mathsf{f}(\acute{}\iota_1\star, \acute{}\iota_2\star)$, but the coequalizer of $T_\Theta(\iota_1)$ and $T_\Theta(\iota_2)$ does not, because $\mathsf{f}(\acute{}\iota_1\star, \acute{}\iota_1\star)$ is not in the image of either of these functors.

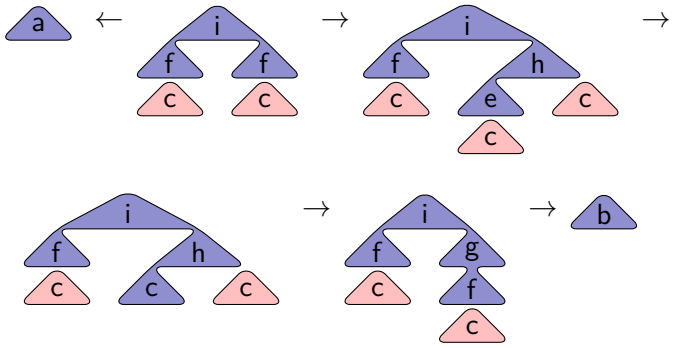


Figure 5.3: Non-confluence in Example 5.88.

5.6.2 Modularity

We compare the proof setups of [43] and [63].

The first difference concerns the decomposition of terms. Whereas Klop et al. split a term into its max-top and aliens, van Oostrom splits it into a base context and a sequence of tall aliens. This is the key for making the proof constructive: while fusion of an alien may cause many new aliens to appear, none of them will be tall, so they do not have to be tracked explicitly. In contrast, Klop et al. start by constructing witnesses, and thus prevent aliens from fusing while establishing confluence.

The other ingredients of the proofs are quite similar: The proof setup is an induction on the rank of the starting term. One distinguishes inner ($\to_i^*$, acting on aliens) and outer ($\to_o^*$, acting on the max-top) steps (Klop et al.), or tall ($\rhd\!\rhd_\iota$, acting on the tall aliens) and short ($\blacktriangleright\!\blacktriangleright$, acting on the base context) steps (van Oostrom). One then argues as follows:

1. Outer (short) steps are confluent because one can replace the principal (tall) subterms by suitable variables in the top (base) context, and then invoke the induction hypothesis.

2. Inner (tall) steps are confluent because they only act on principal subterms (tall aliens). In joining these subterms, one can ensure that any equalities between them are preserved (we call such sequences of inner steps *balanced*). In van Oostrom's proof, the resulting joining sequences may involve fusion and therefore short steps, but by ranking short steps below tall steps, the result becomes a decreasing diagram.

3. Balanced inner steps (tall steps) and outer steps (short steps) commute (can be joined decreasingly). The idea is to replace the principal subterms (tall aliens) of source and target of the inner steps by the same variables, so that the outer steps can be simulated on the result. In van Oostrom's proof, the target term has to be balanced (with respect to the source) first.

When specialized to modularity, the same differences and similarities can be encountered when comparing [22] to the present work. Short steps differ in two ways from [63]. The imbalance is defined differently and the underlying rewrite sequences are less restricted here. Nevertheless, they define the same relation on native terms. This covers Theorem 5.19. For Theorems 5.14 and 5.16, our proof deals with a new effect, namely fusion from above. This makes confluence of short short steps ($\blacktriangleright\!\!\blacktriangleright$) a non-trivial matter.

We remark that layer systems according to Definition 5.3 differ from those in [22]. The latter are closer to variable-restricted layer systems (Definition 5.70). Since the weakened condition (L_2') is only needed for the order-sorted setting, we decided to base the theory on the easier condition (L_2) instead and then derive the main results for variable-restricted layer systems separately (cf. Section 5.5.3). Furthermore, we remark that the notions of weak layering and layering (which are called weak consistency and consistency in [22]) have changed in an incomparable way, even for variable-restricted layer systems. This is due to the new condition (C_2), which is required for our constructive proof, as shown in Example 5.48.

5.6.3 Constructivity

We say that a TRS is *constructively confluent* if there is a procedure that, given a peak $t \mathrel{{}^*\!\leftarrow} s \to^* u$, constructs a valley $t \to^* v \mathrel{{}^*\!\leftarrow} u$. In [63] constructive confluence is proved to be a modular property for disjoint TRSs.

Most previous proofs of modularity and related results rely on the reduction of terms until they allow no further fusion, which requires checking whether the top layer of a term may collapse, a property which is undecidable. This includes [75, 43, 55, 39, 3, 4, 37, 36]. Interestingly, [49] is constructive, but not applicable in general as observed at the beginning of this section.

The key observation for obtaining a constructive result is that our main tool for establishing confluence, the decreasing diagrams technique, is constructive: If any given *local* peak can be joined decreasingly in a constructive way, then any conversion becomes joinable by exhaustively replacing local peaks by smaller conversions until none are left.

For our proofs to be constructive, the TRS needs to be constructively confluent on terms of rank one. Furthermore, the proofs rely on the decomposition of arbitrary terms into their max-top and aliens. Consequently, we must be able to decide whether a given context $C \sqsubseteq t$ is a max-top of t. In the applications from Section 5.4, this is indeed the case.

If these two assumptions are satisfied, then our proofs are constructive and we obtain the following corollary.

Corollary 5.89. *Let $\mathcal{R}$ be a TRS. Assume that $\mathcal{R}$ is left-linear and weakly layered, or bounded duplicating and weakly layered, or layered. If $\mathcal{R}$ is constructively confluent on terms of rank one and for any context C and term t it is decidable if C is a max-top of t, then $\mathcal{R}$ is constructively confluent.*

We remark that the above corollary extends to variable-restricted layer systems, and thus to the order-sorted application in Section 5.5.

5.7 Conclusion

In this chapter we have presented an abstract layer framework that covers several known results about the modularity and persistence of confluence. The framework enabled us to correct the result claimed in [3] on order-sorted persistence, and, by placing weaker conditions on left-linear or bounded duplicating systems, to increase its applicability. We have incorporated a decomposition technique based on order-sorted persistence (Theorem 5.65) into CSI [78], our confluence prover. In the implementation we approximate bounded duplication by non-duplication. We also showed how Kahrs' confluence result for curried systems is obtained as an instance of our layer framework.

Chapter 6

Deciding Confluence of Ground TRSs

It is well known that the confluence property of ground term rewrite systems (ground TRSs) is decidable in polynomial time. For an efficient implementation, the degree of this polynomial is of great interest. The best complexity bound in the literature is given by Comon, Godoy and Nieuwenhuis (2001), who describe an $O(n^5)$ algorithm, where n is the size of the ground TRS. In this chapter we improve this bound to $O(n^3)$. The contents of this chapter has appeared previously in [18].

6.1 Introduction

It is well known that confluence of ground TRSs can be decided in polynomial time. In this chapter, we are interested in the degree of the associated polynomial.

To derive a polynomial time decision procedure for confluence of ground TRSs, Comon et al. [14] base their approach on a transformation by Plaisted [66] that flattens the TRS. Then they test *deep joinability* of sides of rules. The authors sketch an implementation with complexity $O(n^5)$, where n is the size of the given TRS. Tiwari [73] and Godoy et al. [29] base their approach on a rewrite closure that constructs tree transducers—the given TRS $\mathcal{R}$ is converted into two TRSs $\mathcal{F}$ and $\mathcal{B}$ such that $\mathcal{F}$ and $\mathcal{B}^{-1}$ are left-flat, right-constant, $\mathcal{F}$ is terminating, and $\to_{\mathcal{R}}^* = \to_{\mathcal{F}}^* \cdot \to_{\mathcal{B}}^*$. They then consider *top-stabilizable* terms to derive conditions for confluence. Tiwari obtains a bound of $O(n^9)$ (but a more careful implementation would end up with $O(n^6)$), while Godoy et al. obtain a bound of $O(n^6)$. The algorithm of [14] is limited to ground TRSs, but [73] extends the algorithm to certain shallow, linear systems, and [28] treats shallow, linear systems in full generality.[1] In these extensions, however, the exponent depends on the maximum arity of the function symbols of the given TRS. In our work we combine ideas from [14, 73, 29] in order to improve the complexity bound to $O(n^3)$. The key ingredients are a Plaisted-style rewrite closure, which results in TRSs $\mathcal{F}$ and $\mathcal{B}$ of only quadratic size, and top-stabilizability, which is cheaper to test than deep joinability.

The remainder of this chapter is structured as follows: We describe the confluence check in Section 6.2. Some experimental results are presented in Section 6.3. Finally we conclude in Section 6.4.

[1] The same claim can be found in [29]. However, rule splitting, a key step in the proof of their Lemma 3.1, only works if left-hand side and right-hand side variables are disjoint for every rule.

6.2 Testing Confluence

We are given a finite ground TRS $\mathcal{R}_0$ over a finite signature $\mathcal{F}$. We may assume without loss of generality that $\mathcal{R}_0$ is curried, with a binary function symbol $\circ$ (representing function application) and no other non-constant function symbols. To curry an arbitrary ground TRS $\mathcal{R}$, one introduces a fresh binary function symbol $\circ$ and replaces all function applications $f(t_1, \ldots, t_n)$ by $(\ldots((f \circ t_1) \circ t_2)\ldots) \circ t_n$. The original function symbols become constants in the curried TRS. It is well-known that currying preserves (non-)confluence (e.g., [39]) and can be performed in linear time, increasing the total size of the rules of the TRS by a constant factor.

Furthermore we assume that $\mathcal{F}$ is minimal, i.e., only function symbols occurring in $\mathcal{R}_0$ are elements of $\mathcal{F}$. We denote the set of constants from $\mathcal{F}$ by $\mathcal{F}_C$. We can make this assumption because (non-)confluence is preserved under signature extension (this follows from the modularity of confluence, [75]). Let K be a countably infinite set of fresh constants (disjoint from $\mathcal{F}_C$), and let u, v, w denote elements of $\mathcal{F}_C \cup K$. We call $u \to v$ a C-rule (constant rule), and $u \circ v \to w$ a D-rule (decreasing rule).

The construction proceeds in four phases: First the TRS is flattened preserving confluence and non-confluence, then we determine its rewrite closure and congruence closure, and finally these closures are used for testing confluence of the flattened TRS.

Example 6.1. We will decide confluence of $\mathcal{R} = \{\mathsf{a} \to \mathsf{b}, \mathsf{a} \to \mathsf{f}(\mathsf{a}), \mathsf{b} \to \mathsf{f}(\mathsf{f}(\mathsf{b}))\}$ and $\mathcal{R}' = \mathcal{R} \cup \{\mathsf{f}(\mathsf{f}(\mathsf{f}(\mathsf{b}))) \to \mathsf{b}\}$. We start with the curried ground TRSs $\mathcal{R}_0 = \{\mathsf{a} \to \mathsf{b}, \mathsf{a} \to \mathsf{f} \circ \mathsf{a}, \mathsf{b} \to \mathsf{f} \circ (\mathsf{f} \circ \mathsf{b})\}$ and $\mathcal{R}'_0 = \mathcal{R}_0 \cup \{\mathsf{f} \circ (\mathsf{f} \circ (\mathsf{f} \circ \mathsf{b})) \to \mathsf{b}\}$.

In the remainder of this chapter, we use the following abbreviation: By $\mathcal{R}^-$, $\mathcal{R}^\pm$, we denote $\mathcal{R}^{-1}$ (where we view $\mathcal{R}$ as a relation on ground terms), $\mathcal{R} \cup \mathcal{R}^{-1}$, and by $|\mathcal{R}|$, $\|\mathcal{R}\|$ the number of rules in $\mathcal{R}$, and the total size of the rules, $\sum_{\ell \to r \in \mathcal{R}}(|\ell| + |r|)$, respectively.

6.2.1 Flattening

First, we flatten the TRS $\mathcal{R}_0$, as follows. We start with $(\mathcal{R}_0, \varnothing)$ and exhaustively apply the rules

$$(\mathcal{R}[u \circ v], \mathcal{E}) \vdash_{\text{ext}} (\mathcal{R}[w], \mathcal{E} \cup \{u \circ v \approx w\})$$
$$(\mathcal{R}[t], \mathcal{E} \cup \{t \approx u\}) \vdash_{\text{simp}} (\mathcal{R}[u], \mathcal{E} \cup \{t \approx u\}),$$

where $w \in K$ is fresh in $\vdash_{\text{ext}}$, and $\mathcal{R}[\cdot]$ is a context of $\mathcal{R}$, i.e., a context $\ell[\cdot]$ (or $r[\cdot]$) of a side of a rule $\ell \to r$ in $\mathcal{R}$, so that $\mathcal{R}[u]$ is obtained by replacing $\ell \to r$ by $\ell[u] \to r$ (or $\ell \to r[u]$) in $\mathcal{R}$. After each $\vdash_{\text{ext}}$ step, we apply $\vdash_{\text{simp}}$ as often as possible before using $\vdash_{\text{ext}}$ again.

In the resulting pair $(\mathcal{R}, \mathcal{E})$, $\mathcal{R}$ consists solely of C-rules (since otherwise, $\vdash_{\text{ext}}$ would be applicable), and $\mathcal{E}$ consists of D-rules. Furthermore, $\to_{\mathcal{R}_0}$ is confluent, if and only if $\to_{\mathcal{E}^\pm \cup \mathcal{R}}$ is, since every deduction step preserves and reflects the confluence property (cf. Lemma 6.2). Because $\vdash_{\text{simp}}$ is applied eagerly, no two left-hand sides in $\mathcal{E}$ are equal, and therefore $\mathcal{E}$ is confluent (it is orthogonal).

Note that as a result, every distinct subterm occurring in $\mathcal{R}$ is represented by exactly one constant from $\mathcal{F}_C \cup K$. This is similar in spirit to the Nelson-Oppen congruence closure algorithm [51].

Lemma 6.2. *If* $(\mathcal{R}, \mathcal{E}) \vdash_{\text{ext}} (\mathcal{R}', \mathcal{E}')$ *or* $(\mathcal{R}, \mathcal{E}) \vdash_{\text{simp}} (\mathcal{R}', \mathcal{E}')$ *then* $\to_{\mathcal{E}^{\pm} \cup \mathcal{R}}$ *is confluent if and only if* $\to_{\mathcal{E}'^{\pm} \cup \mathcal{R}'}$ *is.*

Proof. The rule $\vdash_{\text{ext}}$ can be split into two steps, first adding the rule $u \circ v \approx w$ to $\mathcal{E}$ followed by applying $\vdash_{\text{simp}}$. The first step preserves confluence since any application of the new $\mathcal{E}$-rule can be undone using the corresponding $\mathcal{E}^{-}$-rule and vice versa, and since w is fresh, no rule other than $w \approx u \circ v$ can affect a subterm containing w.

For $\vdash_{\text{simp}}$, we prove that $\to_{\mathcal{E}^{\pm} \cup \mathcal{R}[t]} \subseteq \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}[u]}$ and $\to_{\mathcal{E}^{\pm} \cup \mathcal{R}[u]} \subseteq \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}[t]}$. There are two cases. (i) If a left-hand side of a rule is changed, i.e., $\ell[t] \to r \in \mathcal{R}[t]$ is changed to $\ell[u] \to r \in \mathcal{R}[u]$, then observing that $s[\ell[t]] \to_{\mathcal{E}} s[\ell[u]] \to_{\mathcal{R}[u]} s[r]$ (simulating $\to_{\ell[t] \to r}$ using rules from $\mathcal{E}^{\pm} \cup \mathcal{R}[u]$) and $s[\ell[u]] \to_{\mathcal{E}^{-}} s[\ell[t]] \to_{\mathcal{R}[t]} s[r]$ (simulating $\to_{\ell[u] \to r}$ using rules from $\mathcal{E}^{\pm} \cup \mathcal{R}[t]$) establishes the claim, since all other rules are contained in both $\mathcal{R}[u]$ and $\mathcal{R}[t]$. (ii) If a right-hand side is changed, $\ell \to r[t] \in \mathcal{R}[t]$, $\ell \to r[u] \in \mathcal{R}[u]$, then the simulations $s[\ell] \to_{\mathcal{R}[t]} s[r[t]] \to_{\mathcal{E}} s[r[u]]$ and $s[\ell] \to_{\mathcal{R}[u]} s[r[u]] \to_{\mathcal{E}^{-}} s[r[t]]$ prove the claim. □

Let the result of flattening be $(\mathcal{R}_1, \mathcal{E})$, over an extended signature, where $\mathcal{F}_C$ includes the fresh constants added by $\vdash_{\text{ext}}$. Flattening is straightforward to implement by a bottom-up traversal of the sides of the TRS, replacing subterms of the shape $u \circ v$ by constants, and maintaining a lookup table of which such terms have been seen before. This takes time $O(\|\mathcal{R}_0\| \log(\|\mathcal{R}_0\|))$ (the $\log(\|\mathcal{R}_0\|)$ factor accounts for the lookup table operations), and we have $\|\mathcal{E}\| = O(\|\mathcal{R}_0\|)$, $\|\mathcal{R}_1\| = O(|\mathcal{R}_0|)$, i.e., the total size of the TRSs $\mathcal{R}_1$ and $\mathcal{E}$ is at most linear in that of $\mathcal{R}_0$.

Example 6.3 (continued from Example 6.1)**.** We introduce fresh constants fa, fb, ffb and fffb for $\mathsf{f} \circ \mathsf{a}$, $\mathsf{f} \circ \mathsf{b}$, $\mathsf{f} \circ \mathsf{fb}$ and $\mathsf{f} \circ \mathsf{ffb}$, respectively. The resulting TRSs are $(\mathcal{R}_1, \mathcal{E}) = (\{\mathsf{a} \to \mathsf{fa}, \mathsf{a} \to \mathsf{b}, \mathsf{b} \to \mathsf{ffb}\}, \{\mathsf{f} \circ \mathsf{a} \approx \mathsf{fa}, \mathsf{f} \circ \mathsf{b} \approx \mathsf{fb}, \mathsf{f} \circ \mathsf{fb} \approx \mathsf{ffb}\})$ and $(\mathcal{R}_1', \mathcal{E}') = (\mathcal{R}_1 \cup \{\mathsf{fffb} \to \mathsf{b}\}, \mathcal{E} \cup \{\mathsf{f} \circ \mathsf{ffb} \approx \mathsf{fffb}\})$.

6.2.2 Rewrite Closure

In this step, we are given a pair $(\mathcal{R}_1, \mathcal{E})$, where $\mathcal{R}_1$ is a system of C-rules and $\mathcal{E}$ is a system of D-rules. We want to obtain another pair $(\mathcal{R}_2, \mathcal{E})$, where $s \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_2} t$ iff $s \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_1} t$, such that every rewrite sequence in $(\mathcal{R}_1, \mathcal{E})$ can be transformed into a rewrite sequence in $(\mathcal{R}_2, \mathcal{E})$ of a special shape (cf. Lemma 6.4). The inference rules in Figure 6.1 define a relation $u \rightsquigarrow v$ on constants. We will see in a moment that $u \rightsquigarrow v$ iff $u \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_1} v$.
The result of the rewrite closure step is $(\mathcal{R}_2, \mathcal{E})$, where $\mathcal{R}_2 = \{u \to v \mid u \rightsquigarrow v\}$.

Lemma 6.4. $s \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_1} t$ *if and only if* $s \to^*_{\mathcal{E} \cup \mathcal{R}_2} \cdot \to^*_{\mathcal{E}^{-} \cup \mathcal{R}_2} t$.

Proof. Because of (base), we have $\mathcal{R}_1 \subseteq \mathcal{R}_2$, so that $\to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_1} \subseteq \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_2}$. On the other hand, all rules in Figure 6.1 are compatible with the requirement

$$\frac{u \to v \in \mathcal{R}_1}{u \rightsquigarrow v}\ \text{base} \qquad \frac{u \in \mathcal{F}_C}{u \rightsquigarrow u}\ \text{refl} \qquad \frac{u \rightsquigarrow v \quad v \rightsquigarrow w}{u \rightsquigarrow w}\ \text{trans}$$
$$\frac{u_1 \rightsquigarrow v_1 \quad u_2 \rightsquigarrow v_2 \quad \{u_1 \circ u_2 \approx u, v_1 \circ v_2 \approx v\} \subseteq \mathcal{E}}{u \rightsquigarrow v}\ \text{comp}$$

Figure 6.1: Inference rules for rewrite closure.

that $\to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} \subseteq \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_1}$. Therefore, the reachability relation is preserved, i.e., $\to^*_{\mathcal{E}^\pm \cup \mathcal{R}_1} = \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2}$.

First we show that for $u, v \in \mathcal{F}_C$, $u \rightsquigarrow v$ (and therefore $u \to_{\mathcal{R}_2} v$) whenever $u \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} v$. Assume that we have $u \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} v$ but not $u \rightsquigarrow v$. Let $u = t_0 \to \cdots \to t_n = v$ be the shortest sequence of $(\mathcal{E}^\pm \cup \mathcal{R}_2)$ steps from u to v, and pick u and v such that n is minimal. If $n = 0$ then $u = v$, and $u \rightsquigarrow v$ by (refl). If $n = 1$ then $u \to v \in \mathcal{R}_2$ since $\mathcal{E}$ only contains D-rules. If $t_i \in \mathcal{F}_C$ for any $0 < i < n$, then $u \rightsquigarrow t_i \rightsquigarrow v$ by minimality of $u \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} v$, and $u \rightsquigarrow v$ by transitivity (trans). In the remaining case, we have $t_i = u_i \circ v_i$ for all $0 < i < n$, and hence $u_1 \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} u_{n-1}$ and $v_1 \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} v_{n-1}$ since any root step would have a constant from $\mathcal{F}_C$ as source or target. But these two rewrite sequences have length at most $n - 2$, and therefore $u_1 \rightsquigarrow u_{n-1}$ and $v_1 \rightsquigarrow v_{n-1}$, implying $u \rightsquigarrow v$ by the (comp) rule. In all cases we found that $u \rightsquigarrow v$, a contradiction.

Now let $s \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_1} t$. Then $s \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} t$. Assume that this rewrite sequence is not of the shape $s \to^*_{\mathcal{E} \cup \mathcal{R}_2} \cdot \to^*_{\mathcal{E}^- \cup \mathcal{R}_2} t$, but has a minimal number of inversions between $\mathcal{E}$ and $\mathcal{E}^-$ steps (an inversion is any pair of an $\mathcal{E}$ step following an $\mathcal{E}^-$ step, not necessarily directly). Then it has a subsequence of the shape $s' \to_{p,\mathcal{E}^-} s'' \to^*_{\mathcal{R}_2} t'' \to_{q,\mathcal{E}} t'$, starting with an $\mathcal{E}^-$ step at p and a final $\mathcal{E}$ step at q. The cases $p < q$ or $p > q$ are impossible, because $\mathcal{E}$ contains only D-rules and $\mathcal{R}_2$ only C-rules (applying C-rules does not change the set of positions of a term).

If $p = q$ then $s''|_{pi} \to^*_{\mathcal{R}_2} t''|_{qi}$ for $i \in \{1, 2\}$, collecting all $\mathcal{R}_2$ steps at positions below p from $s'' \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} t''$. By (refl) and (trans) this implies $s''|_{pi} \rightsquigarrow t''|_{qi}$ for $i \in \{1, 2\}$. Consequently, we have $s'|_p \rightsquigarrow t'|_p$ by (comp). Hence we can delete the two $\mathcal{E}^\pm$ steps and the collected $\mathcal{R}_2$ steps, and replace them by an $\mathcal{R}_2$ step using the rule $s'|_p \to t'|_p$. This decreases the number of inversions between $\mathcal{E}$ and $\mathcal{E}^-$ steps, contradicting our minimality assumption. Otherwise, if $p \parallel q$, then we can reorder the rewrite sequence $s' \to^*_{\mathcal{E}^\pm \cup \mathcal{R}_2} t'$ as $s' \to^*_{>q,\mathcal{R}_2} \cdot \to_{q,\mathcal{E}} \cdot \to^*_{\mathcal{R}_2} \cdot \to_{p,\mathcal{E}^-} \cdot \to^*_{>p,\mathcal{R}_2} t'$, commuting mutually parallel rewrite steps. This reduces the number of inversions between $\mathcal{E}$ and $\mathcal{E}^-$ steps, and again we reach a contradiction. □

The size of $\mathcal{R}_2$ is bounded by $|\mathcal{F}_C|^2 = O(\|\mathcal{R}_0\|^2)$. Note that inference rules correspond naturally to Horn clauses, which are implications with a conjunction of positive literals on the left-hand side and at most one positive literal on the right-hand side. For example,

$$\frac{A \quad B}{C}$$

is equivalent to $A \wedge B \to C$. Therefore, we view the inference rules in Figure 6.1 as a system of Horn clauses with atoms of the form $u \rightsquigarrow v$ $(u, v \in \mathcal{F}_C)$.

rewrite-closure$(n, \mathcal{E}, \mathcal{R})$: Compute rewrite closure.
(Assumes that $\mathcal{F}_C = \{1, \ldots, n\}$, which can be achieved as part of the flattening step).

1. By scanning $\mathcal{E}$ once, compute arrays l and r such that $l[u] = \{(v, w) \mid u \circ v \to w \in \mathcal{E}\}$ and $r[v] = \{(u, w) \mid u \circ v \to w \in \mathcal{E}\}$.
2. Let $\mathcal{R}' = \varnothing \subseteq \{1, \ldots, n\}^2$ (represented by an array).
3. Process (refl): Call **add**(u, u) for $1 \leqslant u \leqslant n$.
4. Process (base): Call **add**(u, v) for $u \to v \in \mathcal{R}$.

add(u, v): Add $u \to v$ to $\mathcal{R}'$ and process implied (trans) and (comp) rules.

1. If $u \to v \in \mathcal{R}'$, return immediately.
2. Let $\mathcal{R}' = \mathcal{R}' \cup \{u \to b\}$.
3. Process (trans): For all $w \in \{1, \ldots, n\}$,
   - if $w \to u \in \mathcal{R}'$, call **add**(w, v).
   - if $v \to w \in \mathcal{R}'$, call **add**(u, w).
4. Process (comp):
   - For all $(u_2, u_r) \in l[u]$ and $(v_2, v_r) \in l[v]$, if $u_2 \to v_2 \in \mathcal{R}'$, call **add**$(u_r, v_r)$.
   - For all $(u_1, u_r) \in r[u]$ and $(v_1, v_r) \in r[v]$, if $u_1 \to v_1 \in \mathcal{R}'$, call **add**$(u_r, v_r)$.

Figure 6.2: Algorithm for rewrite closure.

This system can be solved in time proportional to the total size of the clauses [16], finding a minimal solution for the relation $\rightsquigarrow$. There are $|\mathcal{R}_1|$ instances of (base), $|\mathcal{F}_C|$ instances of (refl), $|\mathcal{F}_C|^3$ instances of (trans) and at most $|\mathcal{F}_C|^2$ instances of (comp), noting that u_1, u_2 are determined by u and v_1, v_2 are determined by v. Therefore, we can compute $\mathcal{R}_2$ in time $O(\|R_0\|^3)$.

Remark 6.5. In our implementation, we do not generate these Horn clauses explicitly. Instead, whenever we make a new inference $u \rightsquigarrow v$, we check all possible rules that involve $u \rightsquigarrow v$ as a premise. The result is a neat incremental algorithm (see Figure 6.2). From an abstract point of view, however, this is no different than solving the Horn clauses as stated above. This remark also applies to inference rules presented later.

Example 6.6 (continued from Example 6.3)**.** We present $\mathcal{R}_2$ and $\mathcal{R}_2'$ as tables, where non-empty entries correspond to the rules contained in each TRS. For example, $\mathsf{fa} \to \mathsf{b} \in \mathcal{R}_2'$ but $\mathsf{fa} \to \mathsf{b} \notin \mathcal{R}_2$. The letters indicate the inference rule used to derive the entry, while the superscripts indicate stage numbers—each

inference uses only premises that have smaller stage numbers.

$\mathcal{R}_2$	f	a	fa	b	fb	ffb
f	r^0					
a		r^0	b^0	b^0	t^2	t^1
fa			r^0		c^1	c^3
b				r^0		b^0
fb					r^0	
ffb						r^0

$\mathcal{R}_2'$	f	a	fa	b	fb	ffb	fffb
f	r^0						
a		r^0	b^0	b^0	t^2	t^1	t^3
fa			r^0	t^3	c^1	t^3	t^2
b				r^0	t^4	b^0	t^4
fb				t^2	r^0	t^2	c^1
ffb				t^4	c^3	r^0	c^3
fffb				b^0	t^4	t^1	r^0

6.2.3 Congruence Closure

We are also interested in the congruence closure of $(\mathcal{R}_1, \mathcal{E})$, because it allows us to decide when two terms are convertible. We calculate the congruence closure as the rewrite closure of $(\mathcal{R}_1^{\pm}, \mathcal{E})$ and call it $(\mathcal{C}, \mathcal{E})$. This step also takes $O(\|\mathcal{R}_0\|^3)$ time. By Lemma 6.4 we have

$$s \leftrightarrow^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_1} t \iff s \to^*_{\mathcal{E}^{\pm} \cup \mathcal{R}_1^{\pm}} t \iff s \to^*_{\mathcal{E} \cup \mathcal{C}} \cdot \to^*_{\mathcal{E}^{-} \cup \mathcal{C}} t \iff s \downarrow_{\mathcal{E} \cup \mathcal{C}} t$$

Note that $\mathcal{C}$ is symmetric and therefore, $\to_{\mathcal{E}^{-} \cup \mathcal{C}} = \mathcal{E} \cup \mathcal{C} \leftarrow$.

Remark 6.7. There are far more efficient methods for calculating the congruence closure (an almost linear time algorithm can be found in [8]), but the simple reduction to the rewrite closure is sufficient for our purposes, since the total asymptotic running time is unchanged.

Example 6.8 (continued from Example 6.6)**.** Since $\mathcal{C}$ is an equivalence relation, we just give its equivalence classes: $[\mathsf{f}]_{\mathcal{C}} = \{\mathsf{f}\}$ and $[\mathsf{a}]_{\mathcal{C}} = \{\mathsf{a}, \mathsf{fa}, \mathsf{b}, \mathsf{fb}, \mathsf{ffb}\}$. For $\mathcal{C}'$, we obtain $[\mathsf{f}]_{\mathcal{C}'} = [\mathsf{f}]_{\mathcal{C}}$ and $[\mathsf{a}]_{\mathcal{C}'} = [\mathsf{a}]_{\mathcal{C}} \cup \{\mathsf{fffb}\}$. Note that $\mathcal{C}$ and $\mathcal{C}'$ are the symmetric, transitive closures of $\mathcal{R}_2$ and $\mathcal{R}_2'$, respectively. This holds in general.

6.2.4 Confluence Conditions

So far we have flattened the TRS $\mathcal{R}_0$ and computed its rewrite and congruence closures, enabling us to check reachability and convertibility of any given terms efficiently. In this section we use these tools to decide confluence of $\mathcal{R}_0$.

We closely follow the approach in [73] and [29], which is based on the analysis of two convertible terms s, t and their normal forms with respect to a system of so-called *forward rules* of a rewrite closure. In our approach, these correspond to the system $\mathcal{E} \cup \mathcal{R}_2$. However, $\to_{\mathcal{E} \cup \mathcal{R}_2}$ is typically non-terminating, and we cannot use this idea directly. This problem is easy to overcome though. We define $\mathcal{A}; \mathcal{B} = \{\ell \to r \mid \ell \to m \in \mathcal{A} \text{ and } m \to r \in \mathcal{B}\}$ and $\to_{\mathcal{A}/\mathcal{B}} = \to^*_{\mathcal{B}} \cdot \to_{\mathcal{A}} \cdot \to^*_{\mathcal{B}}$. Note that $\to_{\mathcal{E}/\mathcal{R}_2}$ is terminating. We will use $\to_{\mathcal{E}/\mathcal{R}_2}$ in place of the forward reduction. This choice is justified by Lemma 6.10 below. We will abuse notation slightly and speak of $\mathcal{E}/\mathcal{R}_2$ normal forms.

Lemma 6.9. *Let $\mathcal{S}$ be a transitive, reflexive (as a relation) set of C-rules and $\mathcal{E}$ a set of D-rules. Then $\to^*_{\mathcal{E} \cup \mathcal{S}} = \to^*_{\mathcal{S}} \cdot \to^*_{\mathcal{E}; \mathcal{S}}$ and $\to^*_{\mathcal{E}^{-} \cup \mathcal{S}} = \to^*_{\mathcal{S}; \mathcal{E}^{-}} \cdot \to^*_{\mathcal{S}}$.*

Proof. We first show that $\to^*_{\mathcal{E}\cup\mathcal{S}} = \to^*_{\mathcal{S}} \cdot \to^*_{\mathcal{E};\mathcal{S}}$. Start with a rewrite sequence $s \to^*_{\mathcal{E}\cup\mathcal{S}} t$. Whenever an $\mathcal{S}$ step is followed by another $\mathcal{S}$ step at the same position, we can combine them using transitivity of $\mathcal{S}$. Note that since $\mathcal{E}$ only contains D-rules, no intermediate $\mathcal{E}$ step can overlap with either of the $\mathcal{S}$ steps. Once there are no more $\mathcal{S}$ steps that can be combined this way, we replace each $\mathcal{E}$ step that is followed by an $\mathcal{S}$ step at the same position by the corresponding $\mathcal{E};\mathcal{S}$ step. If there is no following $\mathcal{S}$ step, we add an identity $\mathcal{S}$ step (which exists by reflexivity of $\mathcal{S}$) first. It is easy to verify that the final rewrite sequence is of the desired shape.

For $\to^*_{\mathcal{E}^-\cup\mathcal{S}} = \to^*_{\mathcal{S};\mathcal{E}^-} \cdot \to^*_{\mathcal{S}}$ it suffices to note that by reversing the rewrite sequences this is equivalent to $\to^*_{\mathcal{E}\cup\mathcal{S}^-} = \to^*_{\mathcal{S}^-} \cdot \to^*_{\mathcal{E};\mathcal{S}^-}$. Since $\mathcal{S}^-$ is transitive and reflexive if $\mathcal{S}$ is, the claim reduces to the previous one. □

Lemma 6.10.

1. *If* $s \to^*_{\mathcal{E}^\pm\cup\mathcal{R}_1} t$ *then* $s \to^*_{\mathcal{E}/\mathcal{R}_2} \cdot \to^*_{\mathcal{R}_2;\mathcal{E}^-} \cdot \to^*_{\mathcal{R}_2} t$.
2. *If* $s \leftrightarrow^*_{\mathcal{E}^\pm\cup\mathcal{R}_1} t$ *then* $s \to^*_{\mathcal{C}} \cdot \to^*_{\mathcal{E};\mathcal{C}} \cdot {}^*_{\mathcal{E};\mathcal{C}}\!\leftarrow \cdot {}^*_{\mathcal{C}}\!\leftarrow t$.

Proof. 1. Assume that $s \to^*_{\mathcal{E}^\pm\cup\mathcal{R}_1} t$. By Lemma 6.4, this is equivalent to $s \to^*_{\mathcal{E}\cup\mathcal{R}_2} \cdot \to^*_{\mathcal{E}^-\cup\mathcal{R}_2} t$, or $s \to^*_{\mathcal{E}/\mathcal{R}_2} \cdot \to^*_{\mathcal{E}^-\cup\mathcal{R}_2} t$, which according to Lemma 6.9 is equivalent to $s \to^*_{\mathcal{E}/\mathcal{R}_2} \cdot \to^*_{\mathcal{R}_2;\mathcal{E}^-} \cdot \to^*_{\mathcal{R}_2} t$, noting that $\mathcal{R}_2^-$ is both reflexive and transitive by construction.

2. Assume that $s \leftrightarrow^*_{\mathcal{E}^\pm\cup\mathcal{R}_1} t$, i.e., $s \to_{\mathcal{E}^\pm\cup\mathcal{R}_1^\pm} t$. Again by Lemma 6.4, this is equivalent to $s \to^*_{\mathcal{E}\cup\mathcal{C}} \cdot {}^*_{\mathcal{E}\cup\mathcal{C}}\!\leftarrow t$. Since $\mathcal{C}$ is reflexive and transitive, the claim follows from Lemma 6.9. □

Let us assume that $\mathcal{R}_1 \cup \mathcal{E}^\pm$ is confluent, and that we have two convertible terms s and t. There are corresponding $\mathcal{E}/\mathcal{R}_2$ normal forms s' and t' for s and t, respectively. Now s' and t' are convertible, so that by Lemma 6.10(2), for some term r,

$$s' \to^*_{\mathcal{C}} \cdot \to^*_{\mathcal{E};\mathcal{C}} r \xleftarrow[\mathcal{E};\mathcal{C}]{*} \cdot \xleftarrow[\mathcal{C}]{*} t' \tag{6.1}$$

Furthermore, by confluence and Lemma 6.10(1), noting that the choice of s' and t' forces the $\to^*_{\mathcal{E}/\mathcal{R}_2}$ sequences to be empty, it follows that for their common reduct r',

$$s' \to^*_{\mathcal{R}_2;\mathcal{E}^-} \cdot \to^*_{\mathcal{R}_2} r' \xleftarrow[\mathcal{R}_2]{*} \cdot \xleftarrow[\mathcal{R}_2;\mathcal{E}^-]{*} t' \tag{6.2}$$

To capture the conditions on s' and t' (which are $\mathcal{E}/\mathcal{R}_2$ normal forms), we adapt the notion of *top-stabilizable* terms and constants from [29] to our purposes.

Definition 6.11. A term $u \circ v$ with $u, v \in \mathcal{F}_C$ is *top-stabilizable* if there exists an $\mathcal{E}/\mathcal{R}_2$ normal form s such that $s \to^*_{\mathcal{C}} \cdot \to^*_{\mathcal{E};C} u \circ v$. A constant $u \in \mathcal{F}_C$ is *top-stabilizable* if there exist $v, w \in \mathcal{F}_C$ such that $u \to_{\mathcal{C};\mathcal{E}^-} v \circ w$ and $v \circ w$ is top-stabilizable.

The equations (6.1,6.2) define two rewrite sequences from r to r' that consist solely of C- and inverse D-steps (note that we consider the rewrite sequences from (6.1) in reverse). This means that no rewrite step occurs below a preceding

rewrite step. In fact all rewrite steps modify a leaf of a term. Therefore we may assume without loss of generality that $r \in \mathcal{F}_C$. Looking at the surrounding rewrite steps in equation (6.1), we distinguish three cases depending on whether the sequence of $\mathcal{E};\mathcal{C}$ steps is empty or not.

1. $s' \to^*_{\mathcal{C}\cup\mathcal{E}} s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} r \ {}_{\mathcal{E};\mathcal{C}}\!\leftarrow t_1 \circ t_2 \ {}_{\mathcal{C}\cup\mathcal{E}}\!\overset{*}{\leftarrow} t'$. In this case $s_1 \circ s_2$, $t_1 \circ t_2$ must be top-stabilizable. Furthermore, for $i \in \{1,2\}$, the terms s_i and t_i are convertible via r_i, so that $s_i \downarrow_{\mathcal{C}\cup\mathcal{E}} t_i$ by Lemma 6.10.

2. $s' \to^*_{\mathcal{C}\cup\mathcal{E}} s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} t' \in \mathcal{F}_C$. (Note that we use the fact that $\mathcal{C}$ is an equivalence relation: $s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} \cdot \overset{*}{\mathcal{C}}\!\leftarrow t'$ implies $s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} t'$ if $t' \in \mathcal{F}_C$.) Then there must be $t_1, t_2 \in \mathcal{F}_C$ such that $t' \to_{\mathcal{R}_2;\mathcal{E}^-} t_1 \circ t_2$, and $s_i \downarrow_{\mathcal{C}\cup\mathcal{E}} t_i$ for $i \in \{1,2\}$. This case also covers $s' \ {}_{\mathcal{E};\mathcal{C}}\!\leftarrow t_1 \circ t_2 \ {}_{\mathcal{C}\cup\mathcal{E}}\!\overset{*}{\leftarrow} t'$ by symmetry.

3. $\mathcal{F}_C \ni s' \to_{\mathcal{C}} t' \in \mathcal{F}_C$. Then $s' \downarrow_{\mathcal{E}^-\cup\mathcal{R}_2} t'$, with common reduct r'.

Hence we have found the following necessary conditions for confluence of $\mathcal{R}_1 \cup \mathcal{E}^{\pm}$:

Definition 6.12. The *confluence conditions* for confluence of $\mathcal{R}_2 \cup \mathcal{E}^{\pm}$ are as follows.

1. If $s_1 \circ s_2$ and $t_1 \circ t_2$ are top-stabilizable for constants $s_1, s_2, t_1, t_2 \in \mathcal{F}_C$ such that $s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} r \ {}_{\mathcal{E};\mathcal{C}}\!\leftarrow t_1 \circ t_2$ then $s_i \downarrow_{\mathcal{C}\cup\mathcal{E}} t_i$ for $i \in \{1,2\}$.

2. If $s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} t'$ for $s_1, s_2, t' \in \mathcal{F}_C$ and top-stabilizable $s_1 \circ s_2$, then there must be $t_1, t_2 \in \mathcal{F}_C$ such that $t' \to_{\mathcal{R}_2;\mathcal{E}^-} t_1 \circ t_2$, and $s_i \downarrow_{\mathcal{C}\cup\mathcal{E}} t_i$ for $i \in \{1,2\}$.

3. If $\mathcal{F}_C \ni s' \to_{\mathcal{C}} t' \in \mathcal{F}_C$ then $s' \downarrow_{\mathcal{E}^-\cup\mathcal{R}_2} t'$.

Lemma 6.13. *The confluence conditions are necessary and sufficient for confluence of* $\mathcal{R}_1 \cup \mathcal{E}^{\pm}$.

Proof. Necessity has already been shown above. For sufficiency, assume that the confluence conditions are satisfied and there are convertible terms s and t with no common reduct. Then any corresponding $\mathcal{E}/\mathcal{R}_2$ normal forms do not have a common reduct either. Let s' and t' be convertible $\mathcal{E}/\mathcal{R}_2$ normal forms with no common reduct such that $|s'| + |t'|$ is minimal. Recall that $\to^*_{\mathcal{E}^{\pm}\cup\mathcal{R}_1} = \to^*_{\mathcal{E}^{\pm}\cup\mathcal{R}_2}$ so that $\mathcal{R}_1 \cup \mathcal{E}^{\pm}$ joinability and $\mathcal{R}_2 \cup \mathcal{E}^{\pm}$ joinability coincide. The same holds for convertibility. We will simply use the terms "joinable" and "convertible" for both $\mathcal{R}_1 \cup \mathcal{E}^{\pm}$ and $\mathcal{R}_2 \cup \mathcal{E}^{\pm}$. We distinguish three cases.

1. If $s', t' \in \mathcal{F}_C$, then by Lemma 6.10(2), $s' \to_{\mathcal{C}} t'$ (since s', t' are $\mathcal{E};\mathcal{C}$ normal forms and $\mathcal{C}$ is an equivalence relation) and we obtain a joining sequence from the third confluence condition, contradicting the non-joinability of s' and t'.

2. If $s' = s'_1 \circ s'_2 \notin \mathcal{F}_C$ and $t' \in \mathcal{F}_C$, then by Lemma 6.10(2) there is a rewrite sequence $s' \to^*_{\mathcal{C}\cup\mathcal{E}} s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} t'$ (again using that t' is an $\mathcal{E};\mathcal{C}$ normal form and that $\mathcal{C}$ is an equivalence relation). By the second confluence condition we obtain a term $t_1 \circ t_2$ such that $t' \to_{\mathcal{R}_2;\mathcal{E}^-} t_1 \circ t_2$, and t_i and

s_i are convertible for $i \in \{1, 2\}$. Therefore, t_1 and s'_1 are convertible. Furthermore, since $|t_1| + |s'_1| < |t'| + |s'|$, this implies that t_1 and s'_1 are joinable. Analogously, t_2 and s'_2 are also joinable, and therefore s' is joinable with $t_1 \circ t_2 \; {}_{\mathcal{R}_2;\mathcal{E}^-}\!\leftarrow t'$, contradicting our assumptions.

The case that $s' \in \mathcal{F}_C$ and $t' \notin \mathcal{F}_C$ is handled symmetrically.

3. If $s' = s'_1 \circ s'_2 \notin \mathcal{F}_C$ and $t' = t'_1 \circ t'_2 \notin \mathcal{F}_C$, then by Lemma 6.10(2), $s' \to^*_{\mathcal{E}\cup\mathcal{C}} r \; {}_{\mathcal{E}\cup\mathcal{C}}\!\overset{*}{\leftarrow} t'$. If $r = r_1 \circ r_2$ is not a constant, then s'_1 and t'_1 are convertible via r_1 and likewise s'_2 and t'_2 are convertible via r_2. However, one of these pairs cannot be joinable, and we obtain a smaller counterexample to confluence, a contradiction. Therefore, r must be a constant.

   Using Lemma 6.10(2) we obtain a rewrite proof $s' \to^*_{\mathcal{C}\cup\mathcal{E}} s_1 \circ s_2 \to_{\mathcal{E};\mathcal{C}} r \; {}_{\mathcal{E};\mathcal{C}}\!\leftarrow t_1 \circ t_2 \; {}_{\mathcal{C}\cup\mathcal{E}}\!\overset{*}{\leftarrow} t'$. From the first confluence condition, we conclude that s_1 and t_1 are convertible and therefore also s'_1 and t'_1. By minimality of $|s'| + |t'|$, s'_1 and t'_1 must be joinable. Likewise, s'_2 and t'_2 must also be joinable, from which we conclude that $s' = s'_1 \circ s'_2$ and $t' = t'_1 \circ t'_2$ are joinable as well, a contradiction.

This completes the proof. □

6.2.5 Computation of Confluence Conditions

The computation consists of two major steps: First we compute all top-stabilizable constants and terms of the form $u \circ v$. Then we check the three confluence conditions.

In order to compute the top-stabilizable constants and terms, we first need to find the $\mathcal{E}/\mathcal{R}_2$ normal forms of the shape $u \circ v$—denoted by $\mathsf{NF}(u \circ v)$. We can compute the complement of that set, i.e., the $\mathcal{E}/\mathcal{R}_2$ reducible terms of that shape using the following inference rules.

$$\frac{u \circ v \approx w \in \mathcal{E}}{\neg\mathsf{NF}(u \circ v)}\ \text{base} \qquad \frac{\{u_1 \to v_1, u_2 \to v_2\} \subseteq \mathcal{R}_2 \quad \neg\mathsf{NF}(v_1 \circ v_2)}{\neg\mathsf{NF}(u_1 \circ u_2)}\ \text{comp}$$

To obtain a cubic time algorithm, note that thanks to transitivity of $\mathcal{R}_2$, inferences made by (comp) need not be processed—if $\neg\mathsf{NF}(w_1 \circ w_2)$ implies $\neg\mathsf{NF}(v_1 \circ v_2)$ by (comp) and $\neg\mathsf{NF}(v_1 \circ v_2)$ implies $\neg\mathsf{NF}(u_1 \circ u_2)$ by (comp) then $\neg\mathsf{NF}(w_1 \circ w_2)$ implies $\neg\mathsf{NF}(u_1 \circ u_2)$ by (comp) as well. Therefore we simply consider each $\mathcal{E}$ rule (there are $O(\|\mathcal{R}_0\|)$ of these) in turn, and then make the corresponding inferences by the (comp) rule in $O(\|\mathcal{R}_0\|^2)$ time, for a total of $O(\|\mathcal{R}_0\|^3)$.

Let us turn to top-stabilizable terms and constants now. First note that any constant is an $\mathcal{E}/\mathcal{R}_2$ normal form. The top-stabilizable constants and terms can be found using another incremental computation. Every $\mathcal{E}/\mathcal{R}_2$ normal form is top-stabilizable. If $u \circ v$ is top-stabilizable and $u \circ v \to_{\mathcal{E}/\mathcal{C}} w$, then w is a top-stabilizable constant, and $u' \circ v$ and $u \circ v'$ are top-stabilizable terms whenever $u \to_{\mathcal{C}} u'$, $v \to_{\mathcal{C}} v'$. For any top-stabilizable constant w, $w \circ v$, $u \circ w$ for constants u, v are also top-stabilizable. Consequently, we obtain the following inference rules, where $\mathsf{TS}(u)$ and $\mathsf{TS}(v \circ w)$ assert that u and $v \circ w$ are top-stabilizable,

respectively, and $(i, i') \in \{(1,2),(2,1)\}$.

$$\frac{u_1 \circ u_2 \in \mathsf{NF}(\mathcal{E}/\mathcal{R}_2)}{\mathsf{TS}(u_1 \circ u_2)}\ \mathrm{nf} \qquad \frac{u_1 \circ u_2 \approx u \in \mathcal{E} \quad \mathsf{TS}(u_1 \circ u_2)}{\mathsf{TS}(u)}\ \mathrm{ts}_0 \qquad \frac{\mathsf{TS}(u_i)}{\mathsf{TS}(u_1 \circ u_2)}\ \mathrm{ts}_i$$

$$\frac{u \to v \in \mathcal{C} \quad \mathsf{TS}(v)}{\mathsf{TS}(u)}\ \mathrm{comp}_0 \qquad \frac{u_i \to v_i \in \mathcal{C} \quad u_{i'} = v_{i'} \quad \mathsf{TS}(v_1 \circ v_2)}{\mathsf{TS}(u_1 \circ u_2)}\ \mathrm{comp}_i$$

There are $O(\|\mathcal{R}_0\|^2)$ instances of (nf), $(\mathrm{ts}_{\{1,2\}})$ and (comp_0), and $O(\|\mathcal{R}_0\|^3)$ instances of (ts_0) and $(\mathrm{comp}_{\{1,2\}})$. Again these inference rules have the shape of Horn clauses and can be processed in time proportional to their total size, which is $O(\|\mathcal{R}_0\|^3)$.

Example 6.14 (continued from Example 6.8)**.** For $\mathcal{R}$ we have $\neg\mathsf{NF} = \{\mathsf{f} \circ \mathsf{b}, \mathsf{f} \circ \mathsf{fa}, \mathsf{f} \circ \mathsf{fb}, \mathsf{f} \circ \mathsf{a}\}$. Indeed $\mathsf{f} \circ \mathsf{ffb}$ is an $\mathcal{E}/\mathcal{R}_2$ normal form since using $\mathcal{R}_2$ it can only be rewritten to itself and it is not the left-hand side of any $\mathcal{E}$ rule. On the other hand, for $\mathcal{R}'$ we obtain $\neg\mathsf{NF}' = \neg\mathsf{NF} \cup \{\mathsf{f} \circ \mathsf{ffb}, \mathsf{f} \circ \mathsf{fffb}\}$. Note that normal forms also include terms like $\mathsf{f} \circ \mathsf{f}$ or $\mathsf{fa} \circ \mathsf{a}$ that have no correspondence in the original TRS.

In the $\mathcal{R}$ case, all terms of the form $u \circ v$ are top-stabilizable and so are all constants except for f. For $\mathcal{R}'$, only the normal forms are top-stabilizable.

With this pre-computation done, checking the confluence conditions becomes a straightforward matter. The only tricky part is checking joinability of constants in the third condition. This relation can be computed in a way strikingly similar to the rewrite closure from Section 6.2.2, using the following inference rules for computing $\downarrow = \downarrow_{\mathcal{E}^- \cup \mathcal{R}_2}$ on constants:

$$\frac{u \in \mathcal{F}_C}{u \downarrow u}\ \mathrm{refl} \qquad \frac{\{u_1 \circ u_2 \approx u, v_1 \circ v_2 \approx v\} \subseteq \mathcal{E} \quad u_1 \downarrow v_1 \quad u_2 \downarrow v_2}{u \downarrow v}\ \mathrm{comp}$$

$$\frac{u \to v \in \mathcal{R}_2 \quad v \downarrow w}{u \downarrow w}\ \mathrm{trans}_1 \qquad \frac{u \downarrow v \quad w \to v \in \mathcal{R}_2}{u \downarrow w}\ \mathrm{trans}_r$$

As with the previous inference rules, this is a system of Horn clauses. There are $O(\|\mathcal{E}\|^2) = O(\|\mathcal{R}_0\|^2)$ instances of (comp), $O(\|\mathcal{R}_0\|)$ instances of (refl) and $O(\|\mathcal{R}_0\|^3)$ instances of $(\mathrm{trans}_{l,r})$. Therefore, computing $\downarrow$ can be done in $O(\|\mathcal{R}_0\|^3)$ time.

Example 6.15 (continued from Example 6.14)**.** The joinability relations $\mathcal{R}$ and $\mathcal{R}'$ are given below. As in Example 6.6, the letters and superscripts indicate the rule being used to derive the entry and computation stage.

$\downarrow =$

	f	a	fa	b	fb	ffb
f	r^0					
a		r^0	t_l^1	t_l^1	t_l^1	t_l^1
fa		t_r^1	r^0	t_l^1	t_l^1	t_l^1
b		t_r^1	t_r^1	r^0		t_l^1
fb		t_r^1	t_r^1		r^0	
ffb		t_r^1	t_r^1	t_r^1		r^0

$\downarrow' =$

	f	a	fa	b	fb	ffb	fffb
f	r^0						
a		r^0	t_l^1	t_l^1	t_l^1	t_l^1	t_l^1
fa		t_r^1	r^0	t_l^1	c^2	c^2	c^2
b		t_r^1	t_r^1	r^0	t_r^1	c^3	c^3
fb		t_r^1	c^2	t_l^1	r^0	c^2	c^4
ffb		t_r^1	c^2	c^3	c^2	r^0	c^3
fffb		t_r^1	c^2	c^3	c^4	c^3	r^0

Cop	21	33	34	38	39	40	80	81	84	114	115	116
CR	×	✓	✓	×	×	✓	×	✓	✓	✓	✓	✓

Table 6.1: Confluence of Ground Cops.

system	$\mathcal{R}_{100}$	$\mathcal{R}_{200}$	$\mathcal{R}_{400}$	$\mathcal{R}_{800}$	$\mathcal{R}_{1600}$	$\mathcal{R}_{3200}$
time (s)	0.2 (×)	0.2 (×)	1.3 (×)	19.2 (×)	254.3 (×)	2321 (×)
system	$\mathcal{R}_{101}$	$\mathcal{R}_{201}$	$\mathcal{R}_{401}$	$\mathcal{R}_{801}$	$\mathcal{R}_{1601}$	$\mathcal{R}_{3201}$
time (s)	0.2 (✓)	0.2 (✓)	2.3 (✓)	30.1 (✓)	427.4 (✓)	3919 (✓)

Table 6.2: Runtimes for $\mathcal{R}_n$.

It is now easy to verify that $\mathcal{R}$ violates the third confluence condition ($\mathsf{fb} \to_{\mathcal{C}} \mathsf{ffb}$ but not $\mathsf{fb} \downarrow \mathsf{ffb}$), and therefore is not confluent. The other two confluence conditions are satisfied. $\mathcal{R}'$, on the other hand, satisfies all confluence conditions and is, therefore, confluent.

Putting everything together, we obtain the following theorem.

Theorem 6.16. *The confluence of a ground TRS $\mathcal{R}$ can be decided in cubic time.*

Proof. Let $n = \|\mathcal{R}\|$. We follow the process outlined above. First we curry $\mathcal{R}$ in linear time, obtaining $\mathcal{R}_0$ with $\|\mathcal{R}_0\| = O(n)$. Then we flatten $\mathcal{R}_0$, obtaining $(\mathcal{R}_1, \mathcal{E})$ with $\|\mathcal{E}\| = O(n)$ and $\|\mathcal{R}_1\| = O(n)$ in time $O(n \log(n))$. In the next step we compute the rewrite and congruence closures $(\mathcal{R}_2, \mathcal{E})$ and $(\mathcal{C}, \mathcal{E})$ of $(\mathcal{R}_1, \mathcal{E})$ in $O(n^3)$ time. Afterwards, we compute the $\mathcal{E}/\mathcal{R}_2$ normal forms $\mathsf{NF}(- \circ -)$, which as seen above takes $O(n^3)$ time. We then compute $\mathsf{TS}(-)$, $\mathsf{TS}(-,-)$ and $\downarrow_{\mathcal{E}^- \cup \mathcal{R}_2}$ in $O(n^3)$ time. Finally we check the three confluence conditions. For the first condition, we check each of the $O(n^2)$ pairs of rules $s_1 \circ s_2 \to_{\mathcal{E}} u, t_1 \circ t_2 \to_{\mathcal{E}} v$ with $u \to_{\mathcal{C}} v$. For the second condition, we consider the $O(n^3)$ triples such that $s_1 \circ s_2 \to_{\mathcal{E}} u \to_{\mathcal{C}} t' \to_{\mathcal{R}_2} v \;\mathcal{E}\!\leftarrow t_1 \circ t_2$. For the third condition we check all $O(n^2)$ pairs $s' \to_{\mathcal{C}} t'$. All these steps can be accomplished in $O(n^3)$ time. □

6.3 Experiments

We have implemented the above algorithm in the confluence tool CSI[2] [78], and tested it on the ground confluence problems from the Cops database.[3] The results are displayed in Table 6.1. There are no runtimes given because they are all negligible. Note though that even before implementing ground confluence in CSI, the tool could handle all these problems. The runtime improved from 14s to 3s for checking all the TRSs. In order to obtain runtime measurements, we considered the family of TRSs $\mathcal{R}_n = \mathcal{R} \cup \{\mathsf{f}^n(\mathsf{b}) \to \mathsf{b}\}$ extending $\mathcal{R}$ from Example 6.1. One can easily argue that the system $\mathcal{R}_n$ is confluent if and only if n is odd. (Since $\mathcal{R}$ is a subsystem, all terms are convertible. However, $\mathsf{f}(\mathsf{b})$

[2] `http://cl-informatik.uibk.ac.at/software/csi/`
[3] `http://coco.nue.riec.tohoku.ac.jp/cops/`

and b are only joinable if n is odd—otherwise the parity of k in the reducts $\mathsf{f}^k(\mathsf{b})$ is invariant.) The runtimes for various n are given in Table 6.2. Saigawa[4] fails on all these systems, while ACP[5] [6] can deal with the non-confluent examples thanks to the interpretation technique from [2], but fails on the confluent ones. The numbers from Table 6.2 do not agree well with the proven complexity bound. This is due to cache effects—as the input size increases, the intermediate arrays outgrow the first and second level caches. Note that for the last two columns, the factor is very close to 8, finally meeting expectations. The difference between odd and even n can be explained by the different size of the rewrite closures. All measurements were done on a 2.67GHz Intel i7-620M computer with 4GB RAM using a single core.

6.4 Conclusion

We have described an efficient algorithm for deciding the confluence of ground TRSs. In our opinion, this is a worthwhile addition to an automated confluence checker, since other methods fail on relatively simple ground TRSs. In fact, ACP can not handle either TRS from Example 6.1, and neither can Saigawa. Before adding the ground TRS code, CSI could not disprove confluence of $\mathcal{R}$, but it was able to prove confluence of $\mathcal{R}'$. It still failed on a close relative of $\mathcal{R}'$, namely the confluent ground TRS $\mathcal{R}_5 = \mathcal{R} \cup \{\mathsf{f}(\mathsf{f}(\mathsf{f}(\mathsf{f}(\mathsf{f}(\mathsf{b}))))) \to \mathsf{b}\}$.

A natural question is whether we can improve the bounds for the other known classes of TRSs with fixed maximum arity that have a known polynomial complexity for deciding complexity, foremost the class of shallow, left-linear TRSs. Our main improvement over [73] is the limitation to C-rules in the rewrite closure, effectively constraining the considered rules to relations between subterms of the original curried TRS. This does no longer work once we have variables in rules. Therefore, at present, we do not know how to improve the other results.

[4] version 1.6, `http://www.jaist.ac.jp/project/saigawa/`
[5] version 0.50, `http://www.nue.riec.tohoku.ac.jp/tools/acp/`

Chapter 7

Certifying Non-Confluence

Regular tree languages are a popular device for reachability analysis over term rewrite systems, with many applications like analysis of cryptographic protocols, or confluence and termination analysis. At the heart of this approach lies tree automata completion, first introduced by Genet for left-linear rewrite systems. Korp and Middeldorp introduced so-called quasi-deterministic automata to extend the technique to non-left-linear systems. In this chapter, we introduce the simpler notion of state-compatible automata, which are slightly more general than quasi-deterministic, compatible automata. This notion also allows us to decide whether a regular tree language is closed under rewriting, a problem which was not known to be decidable before.

The improved precision has a positive impact in applications which are based on reachability analysis, namely termination and confluence analysis.

Our results have been formalized in the theorem prover Isabelle/HOL. This allows to certify automatically generated proofs that are using tree automata techniques. However, the formalization itself is out of scope for this thesis. For details see [20], which forms the basis of this chapter.

7.1 Introduction

In this chapter we are largely concerned with over-approximations of the terms reachable from a regular tree language L_0 by rewriting using a term rewrite system $\mathcal{R}$, that is, we are interested in regular tree languages L such that $\mathcal{R}^*(L_0) \subseteq L$. Such over-approximations have been used, among other things, in the analysis of cryptographic protocols [25], for termination analysis [27, 46] and for establishing non-confluence of term rewrite systems [78].

Unfortunately, the question whether $\mathcal{R}^*(L_0) \subseteq L$ is undecidable in general. Tree automata completion, conceived by Genet et al. [23, 24], is based on the stronger requirements that $L_0 \subseteq L$ and L is itself closed under rewriting, i.e., $\mathcal{R}(L) \subseteq L$. This is accomplished by constructing L as the language accepted by a bottom-up tree automaton $\mathcal{A}$ that is *compatible* with $\mathcal{R}$:

Definition 7.1. A tree automaton $\mathcal{A}$ is *compatible* with a TRS $\mathcal{R}$ if for all state substitutions σ, rules $l \to r \in \mathcal{R}$ and states $q \in Q$, $l\sigma \to^*_{\mathcal{A}} q$ implies $r\sigma \to^*_{\mathcal{A}} q$.

The automaton is constructed by starting with an automaton that accepts L_0 and then identifying violations of the compatibility requirement, that is, state substitutions σ and rules $\ell \to r$ such that $\ell\sigma$ is accepted in some state q but $r\sigma$ is not. One then adds transitions and states to the automaton such that

$r\sigma$ is accepted in state q. If the process terminates with an automaton that is compatible with $\mathcal{R}$ and accepts a language L, then $L_0 \subseteq L$ by construction and L is closed under rewriting by $\mathcal{R}$ by the following result.

Theorem 7.2 ([24])**.** *Let the tree automaton $\mathcal{A}$ be compatible with the TRS $\mathcal{R}$. Then*

1. *if $\mathcal{R}$ is left-linear, then $\mathcal{L}(\mathcal{A})$ is closed under rewriting by $\mathcal{R}$, and*
2. *if $\mathcal{A}$ is deterministic, then $\mathcal{L}(\mathcal{A})$ is closed under rewriting by $\mathcal{R}$.*

□

Example 7.3. Let $\mathcal{R} = \{\mathsf{f}(x, x) \to x\}$ and $\mathcal{A}$ be the automaton with states $1, 2, 3$, final state 3, and transitions

$$\mathsf{a} \to 1 \qquad \mathsf{a} \to 2 \qquad \mathsf{f}(1, 2) \to 3$$

So $\mathcal{A}$ is non-deterministic and $\mathcal{R}$ is non-left-linear. Even though $\mathcal{A}$ is compatible with $\mathcal{R}$, $\mathcal{L}(\mathcal{A}) = \{\mathsf{f}(\mathsf{a}, \mathsf{a})\}$ is not closed under rewriting by $\mathcal{R}$, because $\mathsf{f}(\mathsf{a}, \mathsf{a})$ can be rewritten to a which is not in $\mathcal{L}(\mathcal{A})$.

However, demanding $\mathcal{A}$ to be deterministic if $\mathcal{R}$ is not left-linear may result in bad approximations.

Example 7.4. Let $\mathcal{R} = \{\mathsf{f}(x, x) \to \mathsf{b}, \mathsf{b} \to \mathsf{a}\}$ and $L_0 = \{\mathsf{f}(\mathsf{a}, \mathsf{a})\}$. The set of terms reachable from L_0, namely $\mathcal{R}^*(L_0) = \{\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{b}, \mathsf{a}\}$, is not accepted by any deterministic, compatible tree automaton. To see why, assume that such an automaton $\mathcal{A}$ exists, and let q be the state accepting $\mathsf{f}(\mathsf{a}, \mathsf{a})$. There must be transitions $\mathsf{a} \to q'$ (q' is unique because $\mathcal{A}$ is deterministic) and $\mathsf{f}(q', q') \to q$ in $\mathcal{A}$. By compatibility with the rules $\mathsf{f}(x, x) \to \mathsf{b}$ and $\mathsf{b} \to \mathsf{a}$, we must have transitions $\mathsf{b} \to q$, and $\mathsf{a} \to q$. Since we already have the transition $\mathsf{a} \to q'$, determinism implies $q' = q$. With the three transitions $\mathsf{a} \to q$, $\mathsf{b} \to q$, and $\mathsf{f}(q, q) \to q$, $\mathcal{A}$ accepts every term over the signature $\{\mathsf{f}, \mathsf{a}, \mathsf{b}\}$, which is not a very useful approximation of $\mathcal{R}^*(L_0)$.

To overcome this problem, Korp and Middeldorp introduced quasi-deterministic automata [46]. Indeed it is easy to find a quasi-deterministic automaton accepting $\mathcal{R}^*(L_0) = \{\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{b}, \mathsf{a}\}$ that is compatible with $\mathcal{R}$ from the previous example.

Example 7.5. Let $\mathcal{A}$ be an automaton with states $1, 2$, final state 2 and transitions

$$\mathsf{a} \to 1^* \qquad \mathsf{a} \to 2 \qquad \mathsf{b} \to 2^* \qquad \mathsf{f}(1, 1) \to 2^*$$

where the stars indicate the so-called designated states for each left-hand side. Then $\mathcal{A}$ is quasi-deterministic, compatible with $\mathcal{R}$ and $\mathcal{L}(\mathcal{A}) = \{\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{b}, \mathsf{a}\}$.

In this chapter, we concentrate on the compatibility requirement that ensures $\mathcal{R}(L) \subseteq L$. Since there may be bugs in the implementation of tree automata completion, it is important to independently certify whether $\mathcal{R}(L) \subseteq L$ is really satisfied. Such a certifier has already been developed in [12], but it is restricted

to left-linear systems and does not support the stronger quasi-deterministic automata. We extend this work by introducing *state-compatible* automata, which are deterministic but accomplish the effect of quasi-deterministic automata by relaxing the compatibility requirement instead. It turns out that as long as $\mathcal{R}$ has only non-collapsing rules, state-compatible automata and quasi-deterministic automata are equivalent. In the presence of collapsing rules, state-compatible automata can capture more approximations than quasi-deterministic ones.

We will further show that state-compatibility does not only ensure $\mathcal{R}(L) \subseteq L$, but it can also be utilized to obtain a decision procedure for the question whether a regular tree language is closed under rewriting—to the best of our knowledge, this result is new. We formalized these results within the theorem prover Isabelle/HOL [53], resulting in a formalized decision procedure for the question $\mathcal{R}(L) \subseteq L$. It is used to certify automatically generated proofs in two application areas that utilize tree automata: confluence analysis [78] and termination analysis via match-bounds [45, 46]. For the latter, we also had to adapt another central notion (raise-consistency) from compatible quasi-deterministic automata to our setting of state-compatible deterministic automata.

The improved precision of the decision procedure also has a positive impact within the application areas. To this end, we provide examples where the techniques of [45, 46, 78] are successfully applied when using our decision procedure, but where the techniques must fail if they are restricted to use the compatibility criteria of Genet or Korp and Middeldorp.

This chapter is structured as follows. In Section 7.2 we introduce the central notions of state-coherence and state-compatibility, and present the decision procedure for $\mathcal{R}(L) \subseteq L$. Section 7.3 is devoted to a comparison to quasi-deterministic automata. The next two sections demonstrate that the improved precision of the decision procedure is also helpful in applications. Here, Section 7.4 is on confluence analysis, and Section 7.5 deals with match-bounds. Finally, we conclude in Section 7.6.

7.2 State-Compatible Automata

In this section we will introduce state-compatible automata formally, and show how they ensure $\mathcal{R}(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{A})$. Furthermore, we describe a decision procedure for determining whether this inclusion holds for a given automaton $\mathcal{A}$ and TRS $\mathcal{R}$.

7.2.1 Definitions

Before we get down to definitions, let us briefly analyze the failure in Example 7.4. What happens there is that, by the compatibility requirement, all three terms in the rewrite sequence $\mathsf{f}(\mathsf{a},\mathsf{a}) \to_{\mathcal{R}} \mathsf{b} \to_{\mathcal{R}} \mathsf{a}$ have to be accepted in the same state. In conjunction with the determinism requirement, this is fatal. Consequently, because our goal is to obtain a deterministic automaton, we must allow a and b to be accepted in separate states, q_{a} and q_{b}. To track their connection by rewriting, we introduce a relation $\gg$ on states, such that $q_{\mathsf{b}} \gg q_{\mathsf{a}}$. In general,

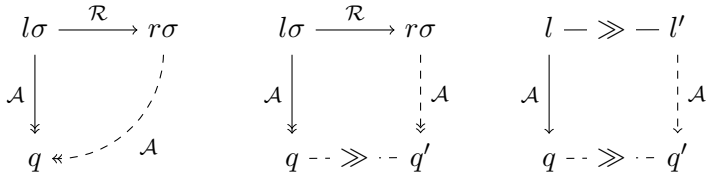


Figure 7.1: Compatibility, state-compatibility, and state-coherence.

we require $\gg$ to be state-compatible and state-coherent, which are defined as follows (see also Figure 7.1).

Definition 7.6. Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a tree automaton, and $\gg \subseteq Q \times Q$ be a relation on the states of $\mathcal{A}$. We say that $(\mathcal{A}, \gg)$ is *state-compatible with a TRS* $\mathcal{R}$ if for all state substitutions σ, rules $l \to r \in \mathcal{R}$ and states $q \in Q$, if $l\sigma \to^*_{\mathcal{A}} q$ then $r\sigma \to^*_{\mathcal{A}} q'$ for some $q' \in Q$ with $q \gg q'$. We say that $(\mathcal{A}, \gg)$ is *state-coherent* if $\{q' \mid q \in Q_f, q \gg q'\} \subseteq Q_f$, and if for all $f(q_1, \ldots, q_i, \ldots, q_n) \to q \in \Delta$ and $q_i \gg q_i'$ there is some $q' \in Q$ with $f(q_1, \ldots, q_i', \ldots, q_n) \to q' \in \Delta$ and $q \gg q'$.

The purpose of state-coherence is to deal with contexts in rewrite steps, as we will see in the proof of Theorem 7.9 below.

Example 7.7. Let $\mathcal{A}$ be an automaton with states $1, 2$ (both final), and transitions

$$\mathsf{a} \to 1 \qquad \mathsf{b} \to 2 \qquad \mathsf{f}(1,1) \to 2$$

Furthermore, let $2 \gg 2$ and $2 \gg 1$. Then $(\mathcal{A}, \gg)$ is state-coherent and state-compatible with $\mathcal{R} = \{\mathsf{f}(x,x) \to \mathsf{b}, \mathsf{b} \to \mathsf{a}\}$ and $\mathcal{L}(\mathcal{A}) = \{\mathsf{f}(\mathsf{a},\mathsf{a}), \mathsf{b}, \mathsf{a}\}$. Note that this automaton was obtained from the quasi-deterministic automaton from Example 7.5 by keeping only the transitions to designated states. We will see in Section 7.3 that this construction works in general.

Remark 7.8. If $(\mathcal{A}, \gg)$ is state-coherent, then $(\mathcal{A}, \gg^=)$ and $(\mathcal{A}, \gg^*)$ are also state-coherent. The same holds for state-compatibility with $\mathcal{R}$.

7.2.2 Soundness and Completeness

Next we prove the analogue of Theorem 7.2 for state-coherent, state-compatible automata.

Theorem 7.9. *Let $\mathcal{A}$ be a tree automaton such that $(\mathcal{A}, \gg)$ is state-coherent and state-compatible with the TRS $\mathcal{R}$ for some relation $\gg$. Then*

1. *if $\mathcal{R}$ is left-linear, then $\mathcal{L}(\mathcal{A})$ is closed under rewriting by $\mathcal{R}$, and*
2. *if $\mathcal{A}$ is deterministic, then $\mathcal{L}(\mathcal{A})$ is closed under rewriting by $\mathcal{R}$.*

Proof. Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$. First we show that whenever $l\tau \to^*_{\mathcal{A}} q$ for some substitution τ and rule $l \to r \in \mathcal{R}$, then there is a state $q' \in Q$ with $q \gg q'$ and $r\tau \to^*_{\mathcal{A}} q'$. By the assumptions, we can extract from $l\tau \to^*_{\mathcal{A}} q$ a state substitution σ such that $l\tau \to^*_{\mathcal{A}} l\sigma \to^*_{\mathcal{A}} q$: For each $x \in \mathcal{V}ar(l)$, we map x to the state reached

from $\tau(x)$ in the given sequence. The state is unique either by left-linearity, or because the given automaton is deterministic. By state-compatibility, we obtain a state q' such that $q \gg q'$ and $r\tau \to_{\mathcal{A}}^* r\sigma \to_{\mathcal{A}}^* q'$.

Using state-coherence we can show by structural induction on C that whenever $C[q] \to_{\mathcal{A}}^* q_\bullet$ and $q \gg q'$, then $C[q'] \to_{\mathcal{A}}^* q'_\bullet$ for some state $q'_\bullet$ with $q_\bullet \gg q'_\bullet$.

Finally, assume that $t \in \mathcal{L}(\mathcal{A})$ and $t \to_{\mathcal{R}} t'$. Then there exist a rule $l \to r \in \mathcal{R}$, a context C and a substitution τ such that $t = C[l\tau]$ and $t' = C[r\tau]$. We have a derivation $t = C[l\tau] \to_{\mathcal{A}}^* C[q] \to_{\mathcal{A}}^* q_\bullet \in Q_f$. By the preceding observations we can find states $q \gg q'$ and $q_\bullet \gg q'_\bullet$ such that $t' = C[r\tau] \to_{\mathcal{A}}^* C[q'] \to_{\mathcal{A}}^* q'_\bullet$. Note that by state-coherence, $q_\bullet \in Q_f$ implies $q'_\bullet \in Q_f$, so that $t' \in \mathcal{L}(\mathcal{A})$. □

Note that Theorem 7.9 generalizes Theorem 7.2 (choose $\gg$ to be the identity relation on states, which is always state-coherent). Moreover, the converse of Theorem 7.9 holds for trim, deterministic automata. We will prove this in Theorem 7.11 below, which allows us to derive our main decidability result in Corollary 7.12. But first let us show by example that the converse fails for some trim, non-deterministic automaton and ground TRS $\mathcal{R}$.

Example 7.10. Consider the TRS $\mathcal{R} = \{\mathsf{a} \to \mathsf{b}\}$ and the automaton $\mathcal{A}$ with states $0, 1, 2, 3$, final state 0, and transitions

$$\begin{array}{llll} \mathsf{a} \to 1 & \mathsf{b} \to 2 & \mathsf{f}(1) \to 0 & \mathsf{g}(1) \to 0 \\ & \mathsf{b} \to 3 & \mathsf{f}(2) \to 0 & \mathsf{g}(3) \to 0 \end{array}$$

This automaton accepts $\mathcal{L}(\mathcal{A}) = \{\mathsf{f}(\mathsf{a}), \mathsf{f}(\mathsf{b}), \mathsf{g}(\mathsf{a}), \mathsf{g}(\mathsf{b})\}$, which is closed under rewriting by $\mathcal{R}$. Assume that $(\mathcal{A}, \gg)$ is state-coherent and state-compatible with $\mathcal{R}$. By state-compatibility, $\mathsf{a} \to \mathsf{b}$ begets $1 \gg 2$ or $1 \gg 3$. If $1 \gg 2$, then state-coherence, considering the transition $\mathsf{g}(1) \to 0$, requires a transition with left-hand side $\mathsf{g}(2)$, which does not exist. Similarly, if $1 \gg 3$, then $\mathsf{f}(1) \to 0$ requires a transition with left-hand side $\mathsf{f}(3)$, which does not exist.

Theorem 7.11. *Let $\mathcal{A}$ be a trim, deterministic tree automaton such that $\mathcal{L}(\mathcal{A})$ is closed under rewriting by the TRS $\mathcal{R}$. Then there is a relation $\gg$ such that $(\mathcal{A}, \gg)$ is state-coherent and state-compatible with $\mathcal{R}$.*

Proof. Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$. We define $\gg$ as follows: $q \gg q'$ iff for some terms $t, t' \in \mathcal{T}(\mathcal{F})$, we have

$$q \xleftarrow[\mathcal{A}]{*} t \xrightarrow[\mathcal{R}]{} t' \xrightarrow[\mathcal{A}]{*} q' \tag{7.1}$$

Note that by virtue of $\mathcal{A}$ being deterministic, t and t' determine q and q' uniquely. We show that $(\mathcal{A}, \gg)$ is state-coherent and state-compatible.

1. (state-coherence) If $q \in Q_f$ and $q \gg q'$, then there exist terms t, t' satisfying (7.1). In particular, $q \in Q_f$ implies $t \in \mathcal{L}(\mathcal{A})$, and $t \to_{\mathcal{R}} t'$ implies $t' \in \mathcal{L}(\mathcal{A})$, because $\mathcal{L}(\mathcal{A})$ is closed under rewriting by $\mathcal{R}$. Because $\mathcal{A}$ is deterministic, t' determines q' uniquely, and $q' \in Q_f$ follows.

2. (state-coherence) Assume that $f(q_1, \dots, q_n) \to q \in \Delta$ and $q_i \gg q_i'$ for some index i and state q_i'. By (7.1) there are t_i, t_i' such that $q_i \xleftarrow[\mathcal{A}]{*} t_i \to_{\mathcal{R}} t_i' \to_{\mathcal{A}}^* q_i'$. Because all q_j are reachable, we can fix terms t_j with $t_j \to_{\mathcal{A}}^* q_j$ for $j \neq i$. The state q is productive, so there is a context C such that $C[q] \to_{\mathcal{A}}^* q_\bullet \in Q_f$. Let $t = f(t_1, \dots, t_n)$ and $t' = f(t_1, \dots, t_i', \dots, t_n)$. Then $C[t] \in \mathcal{L}(\mathcal{A})$ and $C[t] \to_{\mathcal{R}} C[t']$, hence $C[t'] \in \mathcal{L}(\mathcal{A})$ as well. Consequently, there are states $q', q_\bullet'$ such that

$$C[q] \xleftarrow[\mathcal{A}]{*} C[t] \xrightarrow[\mathcal{R}]{} C[t'] \xrightarrow[\mathcal{A}]{*} C[f(q_1, \dots, q_i', \dots, q_n)] \xrightarrow[\mathcal{A}]{} C[q'] \xrightarrow[\mathcal{A}]{*} q_\bullet' \in Q_f$$

   In particular, we have a transition $f(q_1, \dots, q_i', \dots, q_n) \to q' \in \Delta$, and $q \gg q'$.

3. (state-compatibility) Assume that $l\sigma \to_{\mathcal{A}}^* q$ for a state substitution σ. All states of $\mathcal{A}$ are reachable, so there is a substitution $\tau : \mathcal{V} \to \mathcal{T}(\mathcal{F})$ with $\tau(x) \to_{\mathcal{A}}^* \sigma(x)$ for all $x \in \mathcal{V}$. Furthermore, q is productive, so that for some context C, $C[q] \to_{\mathcal{A}}^* q_\bullet \in Q_f$. We have $C[l\tau] \in \mathcal{L}(\mathcal{A})$ and $C[l\tau] \to_{\mathcal{R}} C[r\tau]$. Consequently, $C[r\tau] \in \mathcal{L}(\mathcal{A})$ and for some states $q', q_\bullet'$,

$$C[q] \xleftarrow[\mathcal{A}]{*} C[l\sigma] \xleftarrow[\mathcal{A}]{*} C[l\tau] \xrightarrow[\mathcal{R}]{} C[r\tau] \xrightarrow[\mathcal{A}]{*} C[q'] \xrightarrow[\mathcal{A}]{*} q_\bullet' \in Q_f$$

   In particular, $r\tau \to_{\mathcal{A}}^* q'$. Recall that $\mathcal{A}$ is deterministic. Hence we can decompose this rewrite sequence as follows: $r\tau \to_{\mathcal{A}}^* r\sigma \to_{\mathcal{A}}^* q'$. We conclude by noting that $q \gg q'$ by the definition of $\gg$.

□

Corollary 7.12. *The problem $\mathcal{R}(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{A})$ is decidable.*

Proof. W.l.o.g. we may assume that $\mathcal{A}$ is deterministic. Using Proposition 2.64 we may also assume that $\mathcal{A}$ is trim. By Theorems 7.9 and 7.11 the problem reduces to whether there is some relation $\gg$ such that $(\mathcal{A}, \gg)$ is both state-compatible with $\mathcal{R}$ and state-coherent. But since there are only finitely many relations $\gg$ we can just test state-compatibility and state-coherence for each $\gg$. □

Remark 7.13. As a consequence of Theorem 7.11, regular languages accepted by state-coherent automata that are state-compatible with a fixed TRS $\mathcal{R}$ are closed under intersection and union. This can also be shown directly by a product construction.

7.2.3 Deciding $\mathcal{R}(\mathcal{L}(\mathcal{A})) \subseteq \mathcal{L}(\mathcal{A})$

In the remainder of this section we show that instead of testing all possible relations $\gg$, it suffices to construct a minimal one. We proceed as follows:

1. We assume that $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ is trim and deterministic. Note that given a non-deterministic automaton, we can compute an equivalent deterministic one in exponential time. Once we have a deterministic automaton, we can compute an equivalent trim one in polynomial time.

2. In the following steps we will find the smallest relation $\gg$ that makes $(\mathcal{A}, \gg)$ both state-compatible with $\mathcal{R}$ and state-coherent, if such a relation exists. Initially set $\gg = \emptyset$.

3. Consider each rule $l \to r \in \mathcal{R}$, each state substitution σ, and each state $q \in Q$ such that $l\sigma \to_\Delta^* q$. If there is some state q' with $r\sigma \to_\Delta^* q'$ then add (q, q') to $\gg$. Otherwise, $\mathcal{L}(\mathcal{A})$ is not closed under rewriting by $\mathcal{R}$, and the procedure terminates.

   At this point it is ensured that (any extension of) $\gg$ is state-compatible with $\mathcal{R}$.

4. In order to ensure state-coherence, we repeat the following process until $\gg$ is not increased any further.

   Whenever $q \gg q'$ and $f(q_1, \ldots, q_i = q, \ldots, q_n) \to q_\bullet \in \Delta$, then we look for a transition with left-hand side $f(q_1, \ldots, q_i' = q', \ldots, q_n)$ in Δ. If no such transition exists, state-coherence fails, and the algorithm terminates. Otherwise, let $q_\bullet' \in Q$ be the corresponding right-hand side and add $(q_\bullet, q_\bullet')$ to $\gg$.

   At this point $\gg$ is the smallest relation satisfying state-compatibility with $\mathcal{R}$ which additionally satisfies the second condition of state-coherence.

5. Assert for all $q \gg q'$ with final state q that also q' is final.

Step 3 which identifies the applicable instances of the state-compatibility constraint, consists of a polynomial number of NP queries, and step 4 can be performed in polynomial time. The whole procedure is, therefore, in the Δ_2^{P} (or P^{NP}) complexity class for deterministic automata as input.

Remark 7.14. Using [13, Exercise 1.12.2], which shows that it is NP-hard to decide whether an instance of a term l is accepted by a tree automaton $\mathcal{A}$, we can show that deciding whether the language accepted by a deterministic automaton is closed under rewriting by a given TRS is co-NP-hard. To wit, given a term l, a tree automaton $\mathcal{A}$, a fresh unary function $\star$ and a fresh constant $\diamond$, then $\star(\mathcal{L}(\mathcal{A})) = \{\star(x) \mid x \in \mathcal{L}(\mathcal{A})\}$ is closed under rewriting by $\star(l) \to \diamond$ if and only if no instance of l is accepted by $\mathcal{A}$.

7.3 Relation to Quasi-Deterministic Automata

In this section we relate deterministic state-coherent, state-compatible automata to quasi-deterministic automata by Korp and Middeldorp [46]. Our interest in quasi-deterministic automata is two-fold. First, they represent the state of the art in tree automata completion for non-left-linear TRSs. Secondly, the existing implementation of tree automata completion in $\mathsf{T_TT_2}$ and CSI is based on quasi-deterministic automata, and our goal is to certify the resulting non-confluence and termination proofs (cf. Sections 7.4 and 7.5). We show that given a compatible, quasi-deterministic automaton, we can extract a state-compatible, deterministic automaton accepting the same language, while

the opposite direction fails in the presence of collapsing rules. In our tools' implementation, due to Korp, this restriction is overcome by adding ϵ-transitions to the quasi-deterministic automata, and we will conclude the section with a description of this extension, which was hitherto unpublished.

First we recall the definitions of compatibility and quasi-determinism.

Definition 7.15 (Definition 18 of [46])**.** Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a tree automaton. For a left-hand side $l \in \mathrm{lhs}(\Delta)$ of a transition, we denote the set $\{q \mid l \to q \in \Delta\}$ of possible right-hand sides by $Q(l)$. We call $\mathcal{A}$ *quasi-deterministic* if for every $l \in \mathrm{lhs}(\Delta)$ there exists a *designated state* $p \in Q(l)$ such that for all transitions $f(q_1, \ldots, q_n) \to q \in \Delta$ and $i \in \{1, \ldots, n\}$ with $q_i \in Q(l)$, the transition $f(q_1, \ldots, q_{i-1}, p, q_{i+1}, \ldots, q_n) \to q$ belongs to Δ. Moreover, we require that $p \in Q_f$ whenever $Q(l)$ contains a final state.

For each $l \in \mathrm{lhs}(\Delta)$ we fix a designated state p_l satisfying the constraints of Definition 7.15. We denote the set of designated states by Q_d and the set $\{l \to p_l \mid l \in \mathrm{lhs}(\Delta)\}$ by Δ_d. The notion of compatibility used for quasi-deterministic tree automata is refined slightly from the standard one, Definition 7.1.

Definition 7.16 (Definition 23 of [46])**.** Let $\mathcal{R}$ be a TRS and L a language. Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a quasi-deterministic tree automaton. We say that $\mathcal{A}$ is *compatible* with $\mathcal{R}$ and L if $L \subseteq \mathcal{L}(\mathcal{A})$ and for each rewrite rule $l \to r \in \mathcal{R}$ and state substitution $\sigma \colon \mathcal{V}ar(l) \to Q_d$ such that $l\sigma \to^*_{\Delta_d} q$ it holds that $r\sigma \to^*_{\Delta} q$.

Example 7.5 exhibits a quasi-deterministic, quasi-compatible automaton.

We will show that for each quasi-deterministic automaton that is compatible with a TRS $\mathcal{R}$, there is a deterministic, state-coherent automaton that is state-compatible with $\mathcal{R}$ and accepts the same language. To this end, we need the following key lemma, a slight generalization of [46, Lemma 20], which shows that a quasi-deterministic automaton $\mathcal{A}$ is almost deterministic: all but the last step in a reduction can be performed using the deterministic Δ_d transitions.

Lemma 7.17. *Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a quasi-deterministic automaton. If $t \to^+_{\Delta} q$ then $t \to^*_{\Delta_d} \cdot \to_{\Delta} q$ for all terms $t \in \mathcal{T}(\mathcal{F} \cup Q)$ and states $q \in Q$.*

Proof. The proof is identical to the proof of [46, Lemma 20], except when t_i in $t = f(t_1, \ldots, t_n)$ is a state. In that case, we let $p_{l_i} = q_i = t_i$. □

Theorem 7.18. *Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a quasi-deterministic tree automaton that is compatible with $\mathcal{R}$. Then $\mathcal{A}' = (\mathcal{F}, Q_d, Q_f \cap Q_d, \Delta_d)$ makes $(\mathcal{A}', \gg)$ state-coherent and state-compatible with $\mathcal{R}$, where $q \gg q'$ if $q = q'$ or, for some left-hand side $l \in \mathrm{lhs}(\Delta)$, $q \in Q(l)$ and $q' = p_l$. Furthermore, $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$.*

Proof. Note that $\to_{\mathcal{A}} = \to_{\Delta}$ and $\to_{\mathcal{A}'} = \to_{\Delta_d}$.

1. (state-coherence) Assume that q is final in $\mathcal{A}'$, and $q \gg q'$. If $q = q'$ then q' is final, too. Otherwise, there is a left-hand side l such that $q \in Q(l)$ and $q' = p_l$ is the designated state of l. Since $Q(l)$ contains a final state (namely, q), q' must be final as well by Definition 7.15.

2. (state-coherence) Let $l = f(q_1, ..., q_i, ..., q_n)$ and $l' = f(q_1, ..., q_i', ..., q_n)$, where $q_i \gg q_i'$. Furthermore, let $l \to q \in \Delta_d$. If $q_i = q_i'$ then $l' \to q \in \Delta_d$ and $q \gg q$. Otherwise, there is a left-hand side $l^\bullet$ such that $q_i \in Q(l^\bullet)$ and $q_i' = p_{l^\bullet}$ is the designated state of $l^\bullet$. By Definition 7.15, there is a transition $l' \to q$ in Δ. Thus, l' is a left-hand side and $q \in Q(l')$. Furthermore, $l' \to p_{l'} \in \Delta_d$, and $q \gg p_{l'}$ follows.

3. (state-compatibility) Let σ be a state substitution and $l\sigma \to_{\Delta_d}^* q$. By compatibility, we have $r\sigma \to_\Delta^* q$. If r is a variable, we are done, noting that $q \gg q$. Otherwise, using Lemma 7.17, there is a left-hand side $l' \in \mathrm{lhs}(\mathcal{A})$ such that $r\sigma \to_{\Delta_d}^* l' \to_\Delta q$. Consequently, $r\sigma \to_{\Delta_d}^* \cdot \to_{\Delta_d} p_{l'}$, and since $q \in Q(l')$, we have $q \gg p_{l'}$.

4. (accepted language) $\mathcal{L}(\mathcal{A}') \subseteq \mathcal{L}(\mathcal{A})$ is obvious. To show $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')$, assume that $t \in \mathcal{L}(\mathcal{A})$, i.e., $t \to_\Delta^* q \in Q_f$. By Lemma 7.17, there is a left-hand side $l \in \mathrm{lhs}(\mathcal{A})$ such that $t \to_{\Delta_d}^* l \to_\Delta q$. As in the previous item we conclude that $t \to_{\Delta_d}^* p_l$, and $q \gg p_l$. The state p_l is final by state-coherence, so $t \in \mathcal{L}(\mathcal{A}')$ follows.

□

In the opposite direction, we have a positive result for non-collapsing TRSs.

Theorem 7.19. *Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a deterministic automaton and the relation $\gg \subseteq Q \times Q$ be such that $(\mathcal{A}, \gg)$ is state-coherent and state-compatible with $\mathcal{R}$. Furthermore, assume that $\mathcal{R}$ contains no collapsing rules. Then the automaton $\mathcal{A}' = (\mathcal{F}, Q, Q_f, \Delta')$ with $\Delta' = \{l \to q' \mid l \to q \in \Delta, q \gg^= q'\}$ is a quasi-deterministic automaton with designated states $p_l = q$ for $l \to q \in \Delta$, such that $\mathcal{A}'$ is compatible with $\mathcal{R}$ and accepts the same language as $\mathcal{A}$.*

Proof. Verifying that the construction results in a quasi-deterministic automaton that is compatible with $\mathcal{R}$ is straightforward. Note that applying Theorem 7.18 to $\mathcal{A}'$ results in some $(\mathcal{A}'', \gg'')$ with $\mathcal{L}(\mathcal{A}'') = \mathcal{L}(\mathcal{A}')$, where $\mathcal{A}''$ is $\mathcal{A}$ with states restricted to Q_d', the right-hand sides of Δ'. This restriction preserves the accepted language. Therefore, $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$. □

If $\mathcal{R}$ contains collapsing rules, quasi-deterministic, compatible automata may be weaker than state-coherent, state-compatible ones, as the following example demonstrates.

Example 7.20. Let $\mathcal{R} = \{\mathsf{f}(x, x) \to x\}$. The automaton $\mathcal{A}'$ over $\{\mathsf{f}, \mathsf{a}\}$ with states $1, 2$, both final, and transitions

$$\mathsf{a} \to 1 \qquad\qquad \mathsf{f}(1, 1) \to 2$$

accepts $L = \{\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{a}\}$. Furthermore, $(\mathcal{A}', \gg)$ is state-coherent and state-compatible with $\mathcal{R}$ if we let $2 \gg 1$.

Now assume that $\mathcal{A} = (\{\mathsf{f}, \mathsf{a}\}, Q, Q_f, \Delta)$ is a quasi-deterministic automaton and compatible with $\mathcal{R}$, and that $\mathsf{f}(\mathsf{a}, \mathsf{a}) \in \mathcal{L}(\mathcal{A})$. We will show that $\mathcal{A}$ accepts all terms over $\{\mathsf{f}, \mathsf{a}\}$. Note that since $\mathsf{f}(\mathsf{a}, \mathsf{a})$ is accepted, a must be a left-hand

side of $\mathcal{A}$. Let q be the designated state of a. By Lemma 7.17, we have a run $\mathsf{f}(\mathsf{a},\mathsf{a}) \to_{\Delta_d}^* \mathsf{f}(q,q) \to_\Delta q' \in Q_f$. Let $q^\bullet$ be the designated state of the left-hand side $\mathsf{f}(q,q)$. By quasi-determinism, $q^\bullet$ is a final state. Compatibility requires that $\mathsf{f}(q,q) \to_{\Delta_d} q^\bullet \;{}_\Delta^*\!\leftarrow q$, i.e., $q^\bullet = q$. So we have a final state q and two transitions $\mathsf{a} \to q$, $\mathsf{f}(q,q) \to q$, and $\mathcal{A}$ accepts all of $\mathcal{T}(\{\mathsf{f},\mathsf{a}\})$.

Remark 7.21. In his thesis [45], Korp generalizes Definition 7.15 (cf. [45, Definition 3.10]) by incorporating an auxiliary relation $\geqslant_{\phi_\mathcal{A}}$ that may be viewed as a precursor to our relation $\gg$. The modified definition permits smaller automata, which benefits implementations, but is more complicated than Definition 7.15. The modification also does not add expressive power. Indeed if $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ satisfies [45, Definition 3.10] using $\geqslant_{\phi_\mathcal{A}}$, then taking $\Delta' = \{l \to q \mid l \in \mathrm{lhs}(\Delta), \phi_\mathcal{A}(l) \geqslant q\}$, the automaton $\mathcal{A}' = (\mathcal{F}, Q, Q_f, \Delta')$ satisfies Definition 7.15, noting that $\phi_\mathcal{A}(l)$ is just another notation for the designated state p_l of l. Furthermore, $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$.

The actual implementation of quasi-deterministic automata in TTT2 and CSI by Korp is based on an extension with ϵ-transitions, which are transitions from states to states.

Definition 7.22. Let $\mathcal{A}_\epsilon = (\mathcal{F}, Q, Q_f, \Delta, \Delta_\epsilon)$ be a tree automaton with ϵ-transitions $\Delta_\epsilon \subseteq Q \times Q$. Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be the corresponding tree automaton with the ϵ-transitions removed. Then $\mathcal{A}_\epsilon$ is *extended quasi-deterministic* if $\mathcal{A}$ is quasi-deterministic and $l \to q' \in \Delta$ whenever $l \to q \in \Delta$ and $q \to q' \in \Delta_\epsilon$.

We let $\to_{\mathcal{A}_\epsilon} = \to_{\Delta \cup \Delta_\epsilon}$. The following lemma is key to justifying the extension.

Lemma 7.23. *Let $\mathcal{A}_\epsilon = (\mathcal{F}, Q, Q_f, \Delta, \Delta_\epsilon)$ be an extended quasi-deterministic automaton with ϵ-transitions. Then for terms $t \in \mathcal{T}(\mathcal{F})$, we have $t \to_{\mathcal{A}_\epsilon}^* q$ if and only if $t \to_\mathcal{A}^* q$.*

Proof. By structural induction on t. □

As a consequence, the language $\mathcal{L}(\mathcal{A}_\epsilon) = \{t \in \mathcal{T}(\mathcal{F}) \mid t \to_{\mathcal{A}_\epsilon}^* q, q \in Q_f\}$ accepted by $\mathcal{A}_\epsilon$ satisfies $\mathcal{L}(\mathcal{A}_\epsilon) = \mathcal{L}(\mathcal{A})$. Furthermore, Lemmas 20 and 21 and Theorem 24 of [46] remain true for extended quasi-compatible automata, and therefore $\mathcal{L}(\mathcal{A}_\epsilon)$ is closed under rewriting by $\mathcal{R}$ if $\mathcal{A}_\epsilon$ is extended quasi-deterministic and compatible with $\mathcal{R}$. In order to obtain a deterministic, state-coherent, state-compatible automaton, we apply Theorem 7.18, and then extend $\gg$ as necessary to ensure compatibility with collapsing rules from $\mathcal{R}$. This is easy in our implementation, because ϵ-transitions correspond to state instances of collapsing rules. More precisely, $q' \to q$ is added only if $l \to x \in \mathcal{R}$, $l\sigma \to_{\mathcal{A}_\epsilon}^* q$ and $\sigma(x) = q'$. A general approach is to observe that $(\mathcal{F}, Q, Q_f, \Delta_d)$ accepts the same language as $\mathcal{A}_\epsilon$, and then compute $\gg$ by the algorithm from Section 7.2.3.

7.4 Confluence

Tree automata have an obvious application for disproving (local) confluence. Given some peak $s \;{}_\mathcal{R}^*\!\leftarrow \cdot \to_\mathcal{R}^* t$ (or $s \;{}_\mathcal{R}\!\leftarrow \cdot \to_\mathcal{R} t$ for local confluence), one has

to prove that s and t are not joinable. To this end, it suffices to find suitable tree automata over-approximating descendants of s and t, respectively.

Observation 7.24. Let $\mathcal{A}_s$ and $\mathcal{A}_t$ be tree automata. If $s \in \mathcal{L}(\mathcal{A}_s)$, $t \in \mathcal{L}(\mathcal{A}_t)$, $\mathcal{L}(\mathcal{A}_s) \cap \mathcal{L}(\mathcal{A}_t) = \emptyset$, and both automata are closed under rewriting with $\mathcal{R}$, then s and t are not joinable w.r.t. $\mathcal{R}$.

Given the peak and both automata for a concrete TRS $\mathcal{R}$, by the decision procedure for closure under rewriting it is easy to check the conditions of Observation 7.24.

Notice that due to the precision of our criterion, we also can strengthen the power of confluence tools which are based on Observation 7.24 as demonstrated in the upcoming example.

Example 7.25. Consider the TRS $\mathcal{R}$ consisting of the following rules.

$$\begin{array}{lll} \mathsf{c} \to \mathsf{f}(\mathsf{a},\mathsf{b}) & \mathsf{f}(x,x) \to x & \mathsf{f}(\mathsf{a},\mathsf{a}) \to \mathsf{f}(\mathsf{b},\mathsf{b}) \\ \mathsf{c} \to \mathsf{f}(\mathsf{a},\mathsf{a}) & \mathsf{f}(x,y) \to \mathsf{f}(y,x) & \mathsf{f}(\mathsf{b},\mathsf{b}) \to \mathsf{f}(\mathsf{a},\mathsf{a}) \end{array}$$

Disproving local confluence is equivalent to finding a non-joinable critical pair. Note that $\mathcal{R}$ contains only one non-joinable critical pair, arising from the peak $\mathsf{f}(\mathsf{a},\mathsf{b}) \,{}_{\mathcal{R}}\!\leftarrow \mathsf{c} \to_{\mathcal{R}} \mathsf{f}(\mathsf{a},\mathsf{a})$.

Proving non-joinability of $\mathsf{f}(\mathsf{a},\mathsf{b})$ and $\mathsf{f}(\mathsf{a},\mathsf{a})$ is possible via Observation 7.24 and the following two automata: the first automaton has one final state 3, and consists of four transitions, and accepts the language $\{\mathsf{f}(\mathsf{a},\mathsf{b}), \mathsf{f}(\mathsf{b},\mathsf{a})\}$.

$$\mathsf{a} \to 1 \qquad \mathsf{b} \to 2 \qquad \mathsf{f}(1,2) \to 3 \qquad \mathsf{f}(2,1) \to 3$$

The second automaton has three final states 1, 2, and 3, and also contains four transitions. It's language is $\{\mathsf{f}(\mathsf{a},\mathsf{a}), \mathsf{f}(\mathsf{b},\mathsf{b}), \mathsf{a}, \mathsf{b}\}$.

$$\mathsf{a} \to 1 \qquad \mathsf{b} \to 2 \qquad \mathsf{f}(1,1) \to 3 \qquad \mathsf{f}(2,2) \to 3$$

By Observation 7.24, $\mathcal{R}$ is not confluent. In the following we will argue that it is impossible to show non-confluence this way when using quasi-deterministic automata and compatibility for closure under rewriting.

Assume there is some quasi-deterministic automaton $\mathcal{A}$ with transitions Δ which accepts $\mathsf{f}(\mathsf{a},\mathsf{a})$ and is compatible with $\mathcal{R}$. In the same way as in Example 7.20 one obtains transitions $\mathsf{a} \to q$ and $\mathsf{f}(q,q) \to q$, where q is a final state and the designated state of both a and $\mathsf{f}(q,q)$. Since $\mathcal{A}$ is closed under rewriting it must also contain $\mathsf{f}(\mathsf{b},\mathsf{b})$ and thus, by the same reasoning there are transitions $\mathsf{b} \to p$ and $\mathsf{f}(p,p) \to p$ for some final state p which is also the designated state of b and $\mathsf{f}(p,p)$.

Since $\mathcal{A}$ is compatible and $\mathsf{f}(\mathsf{a},\mathsf{a}) \to^*_{\Delta_d} q$ we deduce $\mathsf{f}(\mathsf{b},\mathsf{b}) \to^*_{\Delta} q$. By Lemma 7.17 we further conclude $\mathsf{f}(\mathsf{b},\mathsf{b}) \to^*_{\Delta_d} \mathsf{f}(p,p) \to_{\Delta} q$. Hence, both p and q are contained in $Q(\mathsf{f}(p,p))$ where p is the designated state of $\mathsf{f}(p,p)$. Thus, by the definition of quasi-compatible automata, we can exchange q by p in any left-hand side of a transition, so, from $\mathsf{f}(q,q) \to q \in \Delta$ we know that also $\mathsf{f}(q,p) \to q \in \Delta$. Thus we obtain the derivation $\mathsf{f}(\mathsf{a},\mathsf{b}) \to^*_{\Delta} \mathsf{f}(q,p) \to_{\Delta} q$ which shows $\mathsf{f}(\mathsf{a},\mathsf{b}) \in \mathcal{L}(\mathcal{A})$. Therefore, no automaton which accepts $\mathsf{f}(\mathsf{a},\mathsf{b})$ is disjoint from $\mathcal{A}$.

7.5 Match-Bounds

In this section we show how tree automata can be used to prove termination via match-bounds [27]. To this end, we first recapitulate the basic concepts and important results about match-bounds. Note that match-bounds require special treatment for non-left-linear TRSs (see Example 7.27). In [46], raise-consistent automata are used to ensure correctness. We present an adaptation of raise-consistency to our setting, leading to the new notion of state-raise-consistency. Finally, we explain how we treat quasi-compatibility, a weaker condition than compatibility that has been introduced specifically for match-bounds.

7.5.1 A Short Introduction to Match-Bounds

Match-bounds is a termination technique which is based on the following idea.

1. One considers an enriched signature where each original symbol of $\mathcal{F}$ is labeled by some natural number to yield a symbol of $\mathcal{F}' = \mathcal{F} \times \mathbb{N}$.

2. The rewrite rules of $\mathcal{R}$ over $\mathcal{F}$ are enriched by labels in a way that each rewrite step corresponds to an increase of labels.[1] The result is an enriched TRS $\mathcal{R}'$ over $\mathcal{F}'$. Possible enrichments are *match* and *roof* where the details are not relevant for this chapter.

3. One tries to show boundedness, i.e., whenever one picks some initial term where all labels in t are 0, then there must be some bound b so that the labels never exceed b when rewriting with $\mathcal{R}'$.

4. If boundedness is ensured, then termination follows as any infinite derivation would lead to an infinite increase in the labels by 2., which is impossible by 3.

Termination via match-bounds can easily be treated as a tree automata problem: the set of terms where every symbol is labeled by 0, $\mathsf{lift}_0(\mathcal{F})$, is accepted by a tree automaton, and moreover, tree automata completion of $\mathsf{lift}_0(\mathcal{F})$ under $\to_{\mathcal{R}'}$, if successful, yields a suitable bound b for Step 3, namely the largest label in the transitions of the resulting automaton $\mathcal{A}$.

In short, match-bounds can be summarized as follows.

Theorem 7.26. *If $\mathcal{R}'$ is a valid enrichment of $\mathcal{R}$, $\mathcal{R}$ is left-linear, $\mathcal{A}$ is a tree automaton, $\mathsf{lift}_0(\mathcal{F}) \subseteq \mathcal{L}(\mathcal{A})$, and $\mathcal{A}$ is closed under $\to_{\mathcal{R}'}$, then $\mathcal{R}$ is terminating.*

It is well known that the restriction to left-linearity is essential as otherwise not every rewrite step of $\mathcal{R}$ can be simulated by a corresponding step in $\mathcal{R}'$.

Example 7.27. For $\mathcal{R} = \{\mathsf{f}(x,x) \to \mathsf{f}(\mathsf{a},x)\}$ and both possible enrichments we obtain $\mathcal{R}' = \{\mathsf{f}_i(x,x) \to \mathsf{f}_{i+1}(\mathsf{a}_{i+1},x) \mid i \in \mathbb{N}\}$. Now the derivation

$$\mathsf{f}(\mathsf{a},\mathsf{a}) \to_{\mathcal{R}} \mathsf{f}(\mathsf{a},\mathsf{a}) \to_{\mathcal{R}} \mathsf{f}(\mathsf{a},\mathsf{a}) \to_{\mathcal{R}} \dots$$

[1] There is an exception concerning collapsing rules as these do not increase the labels. This is explained in detail in [27].

cannot be simulated in the enriched system since after one step

$$\mathsf{f}_0(\mathsf{a}_0, \mathsf{a}_0) \to_{\mathcal{R}'} \mathsf{f}_1(\mathsf{a}_1, \mathsf{a}_0)$$

there is a mismatch of the labels which cannot be repaired by $\mathcal{R}'$, and thus, the evaluation gets stuck.

To overcome this problem, we follow [46] and consider a special rewrite relation which allows to adjust labels for matching non-left-linear rules. In the following definition, base is the function which removes all labels of a term, and $s{\uparrow}t$ takes two terms with $\mathsf{base}(s) = \mathsf{base}(t)$ as input and performs a component-wise maximum on all labels. For example, $\mathsf{f}_1(\mathsf{a}_1, \mathsf{a}_3){\uparrow}\mathsf{f}_2(\mathsf{a}_0, \mathsf{a}_3) = \mathsf{f}_2(\mathsf{a}_1, \mathsf{a}_3)$. Moreover, for non-empty sets $S = \{s_1, \ldots, s_n\}$ we define $\mathsf{base}(S) = \{\mathsf{base}(s_1), \ldots, \mathsf{base}(s_n)\}$ and ${\uparrow}S = s_1{\uparrow}\ldots{\uparrow}s_n$.

Definition 7.28. Let $\mathcal{R}'$ be some TRS over an enriched signature $\mathcal{F}'$. We define the relation $\xrightarrow{\geqslant}_{\mathcal{R}'}$ as $s \xrightarrow{\geqslant}_{\mathcal{R}'} t$ if there is some rule $l \to r \in \mathcal{R}'$, $l = D\langle x_1, \ldots, x_n\rangle$ with all variables displayed, $s = C[D\langle s_1, \ldots, s_n\rangle]$, $S_i = \{s_j \mid 1 \leqslant j \leqslant n, x_j = x_i\}$ and $|\mathsf{base}(S_i)| = 1$ for each $1 \leqslant i \leqslant n$, $\tau = \{x_1/{\uparrow}S_1, \ldots, x_n/{\uparrow}S_n\}$, and $t = C[r\tau]$.

Note that τ in the previous definition is well-defined since whenever $x_i = x_j$ then $S_i = S_j$. Moreover, if $\mathcal{R}'$ is left-linear, then $\xrightarrow{\geqslant}_{\mathcal{R}'}$ is identical to $\to_{\mathcal{R}'}$.

Using this new relation it is possible to generalize Theorem 7.26 to arbitrary, possibly non-left-linear TRSs.

Theorem 7.29. *If $\mathcal{R}'$ is some valid enrichment of $\mathcal{R}$, $\mathcal{A}$ is some tree automaton, $\mathsf{lift}_0(\mathcal{F}) \subseteq \mathcal{L}(\mathcal{A})$, and $\mathcal{A}$ is closed under $\xrightarrow{\geqslant}_{\mathcal{R}'}$, then $\mathcal{R}$ is terminating.*

7.5.2 Adapting Raise-Consistency

The main difficulty when handling non-left-linear TRSs stems from the changed closure property where the standard rewrite relation $\to_{\mathcal{R}'}$ has been replaced by $\xrightarrow{\geqslant}_{\mathcal{R}'}$.

To handle this problem, the notion of raise-consistency was introduced in [46]. The basic idea is to ensure that whenever an automaton accepts terms s_1, s_2 with $\mathsf{base}(s_1) = \mathsf{base}(s_2)$, it also accepts $s_1{\uparrow}s_2$ in a related state, cf. Lemma 7.31, thereby allowing to perform a step $s \xrightarrow{\geqslant}_{\mathcal{R}'} t$ step by first replacing s by another term s' accepted by the automaton (namely $s' = C[l\tau]$ in terms of Definition 7.28), followed by a plain rewrite step using a rule from $\mathcal{R}'$.

In the following we first adapt the notion of raise-consistency to our setting leading to the notion of state-raise-consistency, and prove that state-raise-consistency in combination with state-compatibility and state-coherence ensures closure under $\xrightarrow{\geqslant}_{\mathcal{R}'}$. Furthermore, we show that raise-consistent quasi-deterministic automata can easily be turned into state-raise-consistent deterministic automata.

Definition 7.30. $(\mathcal{A}, \gg)$ is *state-raise-consistent* if $q \gg q'$ for any transitions $f_i(q_1, \ldots, q_n) \to q$ and $f_j(q_1, \ldots, q_n) \to q'$ of $\mathcal{A}$ with $i < j$.

Until Corollary 7.33 we assume a fixed deterministic automaton $\mathcal{A}$, a TRS $\mathcal{R}'$, and a relation $\gg$, where $(\mathcal{A}, \gg)$ is state-raise-consistent, state-coherent, and state-compatible w.r.t. $\mathcal{R}'$.

Lemma 7.31. *Let $S = \{s_1, \dots, s_n\}$ with $|\mathsf{base}(S)| = 1$. If $s_i \to_{\mathcal{A}}^* q_i$ for all $1 \leqslant i \leqslant n$ then there is some q such that $\uparrow S \to_{\mathcal{A}}^* q$ and $q_i \gg^* q$ for all $1 \leqslant i \leqslant n$*

Proof. We only consider the case $n = 2$ here, which is then easily generalized to arbitrary n. Let $\mathsf{base}(s_1) = \mathsf{base}(s_2)$, and $s_i \to_{\mathcal{A}}^* q_i$ for both $i = 1, 2$. We perform induction on s_1, so let $s_1 = f_a(t_1, \dots, t_k) \to_{\mathcal{A}}^* f_a(p_1, \dots, p_k) \to_{\mathcal{A}} q_1$ and $s_2 = f_b(t'_1, \dots, t'_k) \to_{\mathcal{A}}^* f_b(p'_1, \dots, p'_k) \to_{\mathcal{A}} q_2$ where $\mathsf{base}(t_i) = \mathsf{base}(t'_i)$ for all $1 \leqslant i \leqslant k$. By the induction hypothesis we obtain q'_i such that $t_i \uparrow t'_i \to_{\mathcal{A}}^* q'_i$, $p_i \gg^* q'_i$, and $p'_i \gg^* q'_i$ for each i. Thus,

$$s_1 \uparrow s_2 = f_{\max(a,b)}(t_1 \uparrow t'_1, \dots, t_k \uparrow t'_k) \xrightarrow[\mathcal{A}]{*} f_{\max(a,b)}(q'_1, \dots, q'_k)$$

Using $f_a(p_1, \dots, p_k) \to_{\mathcal{A}} q_1$, $p_i \gg^* q'_i$ and state-coherence, there is some $q_1^\bullet$ such that that $f_a(q'_1, \dots, q'_k) \to_{\mathcal{A}} q_1^\bullet$ and $q_1 \gg^* q_1^\bullet$. Similarly, we obtain $q_2^\bullet$ such that that $f_b(q'_1, \dots, q'_k) \to_{\mathcal{A}} q_2^\bullet$ and $q_2 \gg^* q_2^\bullet$.

It remains to prove existence of some $q^\bullet$ such that $f_{\max(a,b)}(q'_1, \dots, q'_k) \to_{\mathcal{A}} q^\bullet$ and $q_i^\bullet \gg^* q^\bullet$ for both $i = 1, 2$: then we would be able to derive the desired result $s_1 \uparrow s_2 \to_{\mathcal{A}}^* f_{\max(a,b)}(q'_1, \dots, q'_k) \to_{\mathcal{A}} q^\bullet$ and $q_i \gg^* q_i^\bullet \gg^* q^\bullet$ for both i. To show the existence of $q^\bullet$ we distinguish cases depending on how a and b compare.

If $a = b$, then $f_a(q'_1, \dots, q'_k) \to_{\mathcal{A}} q_1^\bullet$, and $f_b(q'_1, \dots, q'_k) \to_{\mathcal{A}} q_2^\bullet$ imply $q_1^\bullet = q_2^\bullet$ by determinism of $\mathcal{A}$. Hence, we can let $q^\bullet := q_1^\bullet = q_2^\bullet$ and are done. If $a < b$, then by state-raise-consistency we conclude $q_1^\bullet \gg q_2^\bullet$ and choose $q^\bullet := q_2^\bullet$ to derive the desired result: $f_{\max(a,b)}(q'_1, \dots, q'_k) = f_b(q'_1, \dots, q'_k) \to_{\mathcal{A}} q_2^\bullet = q^\bullet$ and $q_i^\bullet \gg^* q_2^\bullet = q^\bullet$ for both i. The final case, $a > b$, is symmetric to $a < b$. □

Lemma 7.32. *If $s \to_{\mathcal{A}}^* q$ and $s \xrightarrow{\geqslant}_{\mathcal{R}'} t$, then there exists some q' with $t \to_{\mathcal{A}}^* q'$ and $q \gg^* q'$.*

Proof. Since $s \xrightarrow{\geqslant}_{\mathcal{R}'} t$ there are $l \to r \in \mathcal{R}'$ and C, D, s_i, S_i, and τ satisfying the conditions of Definition 7.28. Hence, we can decompose $s \to_{\mathcal{A}}^* q$ into $s = C[D\langle s_1, \dots, s_n\rangle] \to_{\mathcal{A}}^* C[D\langle q_1, \dots, q_n\rangle] \to_{\mathcal{A}}^* C[p] \to_{\mathcal{A}}^* q$, where $s_i \to_{\mathcal{A}}^* q_i$ for each $1 \leqslant i \leqslant n$ and $D\langle q_1, \dots, q_n\rangle \to_{\mathcal{A}}^* p$. Since $|\mathsf{base}(S_i)| = 1$ we use Lemma 7.31 to obtain for each i some q'_i such that $\uparrow S_i \to_{\mathcal{A}}^* q'_i$ and $q_i \gg^* q'_i$. Moreover, since $S_i = S_j$ whenever $x_i = x_j$ we can ensure that $q'_i = q'_j$ whenever $x_i = x_j$. This allows us to define $\sigma = \{x_1/q'_1, \dots, x_n/q'_n\}$ and we conclude both $D\langle q'_1, \dots, q'_n\rangle = l\sigma$ and $\tau(x_i) \to_{\mathcal{A}}^* \sigma(x_i)$ for all $1 \leqslant i \leqslant n$.

From $D\langle q_1, \dots, q_n\rangle \to_{\mathcal{A}}^* p$, $q_i \gg^* q'_i$ and state-coherence we obtain a p_0 satisfying $l\sigma = D\langle q'_1, \dots, q'_n\rangle \to_{\mathcal{A}}^* p_0$ and $p \gg^* p_0$. From $l\sigma \to_{\mathcal{A}}^* p_0$ and state-compatibility we conclude that there is some p' such that $r\sigma \to_{\mathcal{A}}^* p'$ and $p_0 \gg p'$ and hence $p \gg^* p'$. In the same way, from $C[p] \to_{\mathcal{A}}^* q$, and $p \gg^* p'$ we obtain some q' such that $C[p'] \to_{\mathcal{A}}^* q'$ and $q \gg^* q'$.

This finally yields $t = C[r\tau] \to_{\mathcal{A}}^* C[r\sigma] \to_{\mathcal{A}}^* C[p'] \to_{\mathcal{A}}^* q'$ where $q \gg^* q'$. □

Corollary 7.33. *If $s \in \mathcal{L}(\mathcal{A})$ and $s \xrightarrow{\geqslant}_{\mathcal{R}'} t$, then $t \in \mathcal{L}(\mathcal{A})$.*

Therefore, we have the following criterion for termination of $\mathcal{R}$.

Corollary 7.34. *Let $\mathcal{R}$ be a TRS and $\mathcal{R}'$ a suitable enrichment for $\mathcal{R}$. If $\mathcal{A}$ is a deterministic tree automaton and $(\mathcal{A}, \gg)$ is state-raise-consistent, state-coherent, and state-compatible with $\mathcal{R}'$, and furthermore $\mathsf{lift}_0(\mathcal{F}) \subseteq \mathcal{L}(\mathcal{A})$ then $\mathcal{R}$ is terminating.*

Observe that Corollary 7.33 shows that state-coherence together with state-compatibility, state-raise-consistency, and determinism is a sufficient criterion for closure under rewriting with $\xrightarrow{\geqslant}_{\mathcal{R}'}$. However, it is not a necessary criterion, so there is no analogue of Theorem 7.11 where we replace $\to_{\mathcal{R}'}$ by $\xrightarrow{\geqslant}_{\mathcal{R}'}$ and add state-raise-consistency. This is demonstrated in the following example.

Example 7.35. Let $\mathcal{A}$ be the deterministic and trim automaton with transitions $\mathsf{a}_0 \to 0, \mathsf{a}_1 \to 1, \mathsf{f}_0(0) \to 2$ and final states $1, 2$. It accepts the language $L = \{\mathsf{a}_1, \mathsf{f}_0(\mathsf{a}_0)\}$. Let $\mathcal{R}' = \emptyset$. Then obviously $\mathcal{L}(\mathcal{A}) = L$ is closed under rewriting w.r.t. $\xrightarrow{\geqslant}_{\mathcal{R}'}$.

Now assume there is some relation $\gg$ such that $(\mathcal{A}, \gg)$ is both state-coherent and state-raise-consistent. From the latter and the transitions $\mathsf{a}_0 \to 0$ and $\mathsf{a}_1 \to 1$ we conclude $0 \gg 1$. In combination with the transition $\mathsf{f}_0(0) \to 2$ and state-coherence there must be some state q satisfying $\mathsf{f}_0(1) \to q$ and $2 \gg q$. However, since there is no transition with left-hand side $\mathsf{f}_0(1)$ we derived a contradiction, and thus there is no such relation $\gg$.

We present another example that shows that the match-bounds technique potentially suffers from this limitation.

Example 7.36. Consider the TRS $\mathcal{R} = \{\mathsf{f}(\mathsf{g}(\mathsf{g}(x))) \to \mathsf{f}(\mathsf{g}(x))\}$, whose enrichment is $\mathcal{R}' = \{\mathsf{f}_a(\mathsf{g}_b(\mathsf{g}_c(x))) \to \mathsf{f}_d(\mathsf{g}_d(x)) \mid d = \min(a, b, c) + 1\}$. Let the automaton $\mathcal{A}$ have states $1, 2$, both final, and the following transitions:

$$\begin{array}{lll} \mathsf{f}_0(1) \to 1 & \mathsf{g}_0(1) \to 1 & \mathsf{c}_0 \to 1 \\ \mathsf{f}_1(2) \to 1 & \mathsf{g}_1(1) \to 2 & \end{array}$$

With $\gg$ as equality, $\mathcal{A}$ is deterministic, state-coherent, and state-compatible with $\mathcal{R}'$, and therefore $\mathcal{R}$ is match-bounded.

However, there is no $(\mathcal{A}, \gg)$ that is deterministic, state-coherent, state-compatible with $\mathcal{R}'$, and also state-raise-consistent, and accepts $\mathsf{f}_0(\mathsf{g}_0^k(\mathsf{c}_0))$ for all $k > 0$. To see why, let q_i be the state accepting $\mathsf{g}_0^i(\mathsf{c}_0)$. There is a state q with $\mathsf{f}_0(\mathsf{g}_0(\mathsf{g}_0(q_i))) \to_{\mathcal{A}}^* q$, and by state-compatibility with $\mathcal{R}'$, there are states q', q'' such that $\mathsf{f}_1(\mathsf{g}_1(q_i)) \to_{\mathcal{A}} \mathsf{f}_1(q') \to_{\mathcal{A}} q''$, where $q \gg q''$. In particular, there are transitions $\mathsf{g}_1(q_i) \to q'$, and $\mathsf{g}_0(q_i) \to q_{i+1}$, which implies $q_{i+1} \gg q'$ by state-raise-consistency. By state-coherence this implies that whenever $\mathcal{A}$ accepts $C[\mathsf{g}_0(q_i)]$, then it also accepts $C[\mathsf{g}_1(q_i)]$. We have $\mathsf{f}_1(\mathsf{g}_1(\mathsf{g}_0^k(\mathsf{c}_0)))$ in $\mathcal{L}(\mathcal{A})$ by closure under rewriting, and by induction on j we can now show that $\mathsf{f}_1(\mathsf{g}_1^{j+1}(\mathsf{g}_0^{k-j}(\mathsf{c}_0))) \in \mathcal{L}(\mathcal{A})$. In particular, $\mathsf{f}_1(\mathsf{g}_1^{k+1}(\mathsf{c}_0))$ is accepted by $\mathcal{A}$ as well.

By iterating this construction (increasing all labels of f and g by 1 in each iteration) we can show that $\mathcal{A}$ accepts $\mathsf{f}_a(\mathsf{g}_a^k(\mathsf{c}_0))$ for all $k > 0$ and $a \in \mathbb{N}$, contradicting the fact that $\mathcal{A}$ is finite.

Despite incompleteness, state-raise-consistency still subsumes the criterion of raise-consistency for quasi-deterministic automata.

Theorem 7.37. *Let $\mathcal{A}$, $\mathcal{R}$, $\mathcal{A}'$, and $\gg$ be as in Theorem 7.18. If $\mathcal{A}$ is raise-consistent then $(\mathcal{A}', \gg)$ is state-raise-consistent.*

Proof. Let $f_i(q_1, \ldots, q_n) \to q \in \Delta_d$ and $f_j(q_1, \ldots, q_n) \to q' \in \Delta_d$ be rules of $\mathcal{A}'$ with $i < j$. Hence, both rules are also present in $\mathcal{A}$ and by raise-consistency we conclude that $f_j(q_1, \ldots, q_n) \to q \in \Delta$. By definition of Δ_d we know that $q' = Q(f_j(q_1, \ldots, q_n))$ is the designated state and thus, $q \gg q'$ by the definition of $\gg$. □

Similarly to Example 7.20 we further prove that the inclusion is strict by providing a rewrite system where match-boundedness cannot be proven using quasi-deterministic automata.

Example 7.38. Let $\mathcal{R} = \{\mathsf{f}(x, x) \to x, \mathsf{f}(\mathsf{f}(x, y), z) \to \mathsf{f}(\mathsf{a}, \mathsf{a})\}$ over signature $\mathcal{F} = \{\mathsf{a}, \mathsf{f}\}$. Then the enriched system is $\mathcal{R}' = \{\mathsf{f}_i(x, x) \to x, \mathsf{f}_i(\mathsf{f}_j(x, y), z) \to \mathsf{f}_k(\mathsf{a}_k, \mathsf{a}_k) \mid i, j, k \in \mathbb{N},\ k = 1 + \min(i, j)\}$. In particular, $\mathcal{R}'$ contains the rules $\mathsf{f}_i(x, x) \to x$ and $\mathsf{f}_i(\mathsf{f}_i(x, y), z) \to \mathsf{f}_{i+1}(\mathsf{a}_{i+1}, \mathsf{a}_{i+1})$ for all $i \in \mathbb{N}$.

The deterministic automaton $\mathcal{A}$ over $\{\mathsf{f}, \mathsf{a}\}$ with states $1, 2$, both final, and transitions

$$\begin{aligned} \mathsf{a}_0 &\to 2 & \mathsf{f}_0(p, q) &\to 1 \quad \text{for all } p, q \in \{1, 2\} \\ \mathsf{a}_1 &\to 2 & \mathsf{f}_1(2, 2) &\to 1 \end{aligned}$$

accepts all terms in $\mathsf{lift}_0(\mathcal{F})$ and is closed under rewriting with $\xrightarrow{\geq}_{\mathcal{R}'}$, as $(\mathcal{A}', \gg)$ is state-coherent and state-compatible with $\mathcal{R}'$ and state-raise-consistent if we let $1 \gg 1$ and $1 \gg 2$. Hence, by Theorem 7.29 and Corollary 7.33 termination of $\mathcal{R}$ is proved.

We further prove that a similar proof is impossible if we use compatible, quasi-deterministic automata. To this end, assume that $\mathcal{A} = (\mathcal{F} \times \mathbb{N}, Q, Q_f, \Delta)$ is a quasi-deterministic automaton which is compatible[2] with $\mathcal{R}$ and accepts all terms in $\mathsf{lift}_0(\mathcal{F})$. These conditions imply that $\mathcal{L}(\mathcal{A})$ is closed under rewriting with $\mathcal{R}'$. Obviously, $\mathsf{f}_0(\mathsf{f}_0(\mathsf{a}_0, \mathsf{a}_0), \mathsf{a}_0) \in \mathcal{L}(\mathcal{A})$ and since $\mathsf{f}_0(\mathsf{f}_0(\mathsf{a}_0, \mathsf{a}_0), \mathsf{a}_0) \to_{\mathcal{R}'} \mathsf{f}_1(\mathsf{a}_1, \mathsf{a}_1)$ and $\mathcal{L}(\mathcal{A})$ is closed under rewriting, we also have $\mathsf{f}_1(\mathsf{a}_1, \mathsf{a}_1) \in \mathcal{L}(\mathcal{A})$.

Since $\mathsf{f}_1(x, x) \to x \in \mathcal{R}'$, we can proceed in the same way as in Example 7.20 to show that $\mathcal{A}$ accepts all terms over $\{\mathsf{f}_1, \mathsf{a}_1\}$, i.e., $\mathsf{lift}_1(\mathcal{F})$.

Now, again we have a derivation $\mathsf{f}_1(\mathsf{f}_1(\mathsf{a}_1, \mathsf{a}_1), \mathsf{a}_1) \to_{\mathcal{R}'} \mathsf{f}_2(\mathsf{a}_2, \mathsf{a}_2)$ and by closure under rewriting we conclude $\mathsf{f}_2(\mathsf{a}_2, \mathsf{a}_2) \in \mathcal{L}(\mathcal{A})$ and afterwards derive $\mathsf{lift}_2(\mathcal{F}) \subseteq \mathcal{L}(\mathcal{A})$ as before. Iterating this reasoning yields $\bigcup_{i \in \mathbb{N}} \mathsf{lift}_i(\mathcal{F}) \subseteq \mathcal{L}(\mathcal{A})$. But this is impossible, since $\mathcal{A}$ is a finite automaton which can only have finitely many symbols in the transitions whereas $\bigcup_{i \in \mathbb{N}} \mathsf{lift}_i(\mathcal{F})$ contains infinitely many symbols.

7.5.3 Quasi-Compatibility

In [46, Section 5] the improvement of quasi-compatibility is introduced, which relaxes the compatibility criterion and therefore allows to reduce the size of

[2] We do not even require raise-consistency of $\mathcal{A}$.

the automata. While it is possible to also integrate this refinement into state-compatibility, we omit the details here. The main reason is that every quasi-(state-)compatible automata can easily be turned into an automaton which is also (state-)compatible by just adding more transitions, cf. the remark between Definition 32 and 33 in [46]. Thus, in the same way as we transform quasi-deterministic automata into deterministic automata within Section 7.3, we can also always transform quasi-compatible automata into compatible ones without losing power.

7.6 Conclusion

We have introduced the class of deterministic, state-coherent automata that are state-compatible with a TRS $\mathcal{R}$. We have shown that these automata capture precisely those regular tree languages that are closed under rewriting by $\mathcal{R}$, leading to a decision procedure for checking whether a regular language is closed under rewriting. Their simple definition allowed us to formalize most of our results on state-coherent, state-compatible automata within Isabelle/HOL. Also criteria for match-bounds—raise-consistency to ensure closure under $\xrightarrow{\geq}_{\mathcal{R}'}$—could easily be adapted to the corresponding notion state-raise-consistency. We further demonstrated via examples that the gain in precision carries over to the applications: our notions result in more powerful confluence and termination analysis.

As future work we plan to expand our formalization on match-bounds, e.g., by integrating results on relative rewriting, on match-bounds for dependency pairs, or match-bounds for complexity analysis. Another open question is whether the state-raise-consistency condition can be relaxed to cover more systems.

Chapter 8

Conclusion

In this thesis we have visited a wide range of confluence-related topics, from abstract rewrite systems and decreasing diagrams, labelings for applying the decreasing diagrams technique to term rewrite systems, layer systems, which mainly provide tools for decomposing TRSs into smaller TRSs for the purpose of establishing (non-)confluence, to very concrete techniques for deciding confluence of ground TRSs and closure under rewriting of tree automata for certifying non-confluence. This leaves a lot of opportunities for further research.

- In Chapter 3, we have investigated decreasing diagrams for showing the confluence and Church-Rosser modulo properties of abstract rewrite systems. It should be straightforward to adapt the labeling functions from Chapter 4 to the latter setting, but the details still have to be worked out.

- In Chapter 4, we started out with labeling individual steps, using critical pairs, and then parallel steps, using parallel critical pairs. A natural question is whether this extends to development steps. The most obvious approach is to consider conversions made of development steps. However, the corresponding notion of critical pairs does not lead to a finite set of critical pairs (consider the rule $\mathsf{f}(\mathsf{f}(x)) \to x$. We have $\mathsf{f}^{2k}(x) \mathrel{-\!\!\!\circledcirc} x$, and therefore
$$\mathsf{f}(x) \mathrel{\circledcirc\!\!\!-} \mathsf{f}(\mathsf{f}^{2k}(x)) = \mathsf{f}^{2k}(\mathsf{f}(x)) \mathrel{-\!\!\!\circledcirc} \mathsf{f}(x)$$
which gives rise to a development critical pair for each $k \in \mathbb{N}$. But there is another way to proceed: Rather than sticking to the undirected version of decreasing diagrams, we can use the commutation version, and use plain rewrite steps in one direction, while using development steps in the other direction. This was done to great effect by Okui in [58], where he introduces the notion of *simultaneous critical pairs*. It seems that Okui's result can be fruitfully generalized by adding labels to the rewrite steps.

- A technical challenge posed by labeling functions is that it is often beneficial to label different rewrite steps the same way. For example, the TRS
$$\mathsf{f}(\mathsf{f}(x)) \to \mathsf{g}(\mathsf{g}(x)) \qquad\qquad \mathsf{g}(\mathsf{g}(x)) \to \mathsf{f}(\mathsf{f}(x))$$
can be shown confluent by rule labeling, but only if both rules receive the same label. This means that adding new components to a labeling function can destroy confluence proofs by labeling. It would be great to have a truly incremental approach to labeling diagrams decreasingly.

- In Chapter 5, we have applied layer systems to confluence. A natural question is whether this extends to other properties like uniqueness of normal forms, or perhaps even to non-modular properties like termination under further restrictions.
- One big open problem for confluence tools is the treatment of non-left-linear TRSs. We have seen this in Chapter 5, where dealing with layered terms in the case of left-linear systems is quite straightforward, because fusion never interferes with rewriting sequences, whereas the non-left-linear systems caused a lot of trouble because of the need for rebalancing. Also in Chapter 4 the main reason for restricting most of the theory to left-linear systems is that the need for rebalancing makes joining variable overlaps decreasingly quite hard. There is some recent progress in this area, e.g. [69], but in general it seems that our understanding of confluence of non-left-linear TRSs is still very limited.

Finally, we plan to formalize some results of this thesis in IsaFoR, most notably the layer systems part. Some other parts have been formalized, namely the tree automata results (by René Thiemann), and parts of the labeling results (by Harald Zankl).

Publications

The following publications were produced during the course of my PhD studies (in order of appearance).

- H. Zankl, B. Felgenhauer, and A. Middeldorp. Labelings for decreasing diagrams. In *Proc. 22nd International Conference on Rewriting Techniques and Applications*, volume 10 of *Leibniz International Proceedings in Informatics*, pages 377–392, 2011.

- H. Zankl, B. Felgenhauer, and A. Middeldorp. CSI – A confluence tool. In *Proc. 23rd International Conference on Automated Deduction*, volume 6803 of *Lecture Notes in Artificial Intelligence*, pages 499–505, 2011.

- B. Felgenhauer, H. Zankl, and A. Middeldorp. Proving confluence with layer systems. In *Proc. 31st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 13 of *Leibniz International Proceedings in Informatics*, pages 288–299, 2011.

- B. Felgenhauer. Deciding confluence of ground term rewrite systems in cubic time. In *Proc. 23rd International Conference on Rewriting Techniques and Applications*, volume 15 of *Leibniz International Proceedings in Informatics*, pages 165–175, 2012.

- B. Felgenhauer. A proof order for decreasing diagrams. In *Proc. 1st International Workshop on Confluence*, pages 7–14, 2012.

- B. Felgenhauer. Binomial interpretations. In *Proc. 12th International Workshop on Termination*, 5 pages, 2012.

- B. Felgenhauer and V. van Oostrom. Proof orders for decreasing diagrams. In *Proc. 24th International Conference on Rewriting Techniques and Applications*, number 21 in Leibniz International Proceedings in Informatics, pages 174–189, 2013.

- B. Felgenhauer. Rule labeling for confluence of left-linear term rewrite systems. In *Proc. 2nd International Workshop on Confluence*, pages 23–27, 2013.

- B. Felgenhauer, Martin Avanzini, and Christian Sternagel. A Haskell library for term rewriting. In *Proc. 1st International Workshop on Haskell and Rewriting Techniques*, 6 pages, 2013.

- B. Felgenhauer and R. Thiemann. Reachability analysis with state-compatible automata. In *Proc. 8th International Conference on Language*

and Automata Theory and Applications, volume 8370 of *Lecture Notes in Computer Science*, pages 347–359, 2014.

- M. Avanzini and B. Felgenhauer. Type introduction for runtime complexity analysis. In *Proc. 14th International Workshop on Termination*, 5 pages, 2014.
- H. Zankl, B. Felgenhauer, and A. Middeldorp. Labelings for decreasing diagrams. *Journal of Automated Reasoning*, 54(2):101–133, 2015.
- B. Felgenhauer, A. Middeldorp, H. Zankl, and V. van Oostrom. Layer systems for proving confluence. *ACM Transactions on Computational Logic*, 36 pages, 2015, to appear.

Bibliography

[1] T. Aoto. Automated confluence proof by decreasing diagrams based on rule-labelling. In *Proc. 21st International Conference on Rewriting Techniques and Applications*, volume 6 of *Leibniz International Proceedings in Informatics*, pages 7–16, 2010.

[2] T. Aoto. Disproving confluence of term rewriting systems by interpretation and ordering. In *Proc. 9th International Workshop on Frontiers of Combining Systems*, volume 8152 of *Lecture Notes in Artificial Intelligence*, pages 311–326, 2013.

[3] T. Aoto and Y. Toyama. Extending persistency of confluence with ordered sorts. Technical Report IS-RR-96-0025F, School of Information Science, JAIST, 1996.

[4] T. Aoto and Y. Toyama. Persistency of confluence. *Journal of Universal Computer Science*, 3(11):1134–1147, 1997.

[5] T. Aoto and Y. Toyama. A reduction-preserving completion for proving confluence of non-terminating term rewriting systems. *Logical Methods in Computer Science*, 8(1:31):1–29, 2012.

[6] T. Aoto, J. Yoshida, and Y. Toyama. Proving confluence of term rewriting systems automatically. In *Proc. 20th International Conference on Rewriting Techniques and Applications*, volume 5595 of *Lecture Notes in Computer Science*, pages 93–102, 2009.

[7] C. Appel, V. van Oostrom, and J. G. Simonsen. Higher-order (non-)modularity. In *Proc. 21st International Conference on Rewriting Techniques and Applications*, volume 6 of *Leibniz International Proceedings in Informatics*, pages 17–32, 2010.

[8] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.

[9] L. Bachmair and N. Dershowitz. Equational inference, canonical proofs, and proof orderings. *Journal of the ACM*, 41(2):236–276, 1994.

[10] M. Bognar. A point version of decreasing diagrams. In *Proceedings Accolade 1996*, Dutch Graduate School in Logic, pages 1–14, 1997.

[11] M. Bognar and J. Klop. A note on some abstract confluence criteria. Technical Report IR-411, Vrije Universiteit Amsterdam, 1996.

[12] B. Boyer, T. Genet, and T. P. Jensen. Certifying a tree automata completion checker. In *Proc. 4th International Joint Conference on Automated Reasoning*, volume 5195 of *Lecture Notes in Computer Science*, pages 523–538, 2008.

[13] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, C. Löding, S. Tison, and M. Tommasi. *Tree Automata Techniques and Applications*. 2007. Available at `http://tata.gforge.inria.fr`.

[14] H. Comon, G. Godoy, and R. Nieuwenhuis. The confluence of ground term rewrite systems is decidable in polynomial time. In *Proc. 42nd Annual Symposium on Foundations of Computer Science*, pages 298–307, 2001.

[15] N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17:279–301, 1982.

[16] W. Dowling and J. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *Journal of Logic Programming*, 1(3):267–284, 1984.

[17] J. Endrullis, J. Waldmann, and H. Zantema. Matrix interpretations for proving termination of term rewriting. *Journal of Automated Reasoning*, 40(2-3):195–220, 2008.

[18] B. Felgenhauer. Deciding confluence of ground term rewrite systems in cubic time. In *Proc. 23rd International Conference on Rewriting Techniques and Applications*, volume 15 of *Leibniz International Proceedings in Informatics*, pages 165–175, 2012.

[19] B. Felgenhauer, A. Middeldorp, H. Zankl, and V. van Oostrom. Layer systems for proving confluence. *ACM Transactions on Computational Logic*, 2015. To appear.

[20] B. Felgenhauer and R. Thiemann. Reachability analysis with state-compatible automata. In *Proc. 8th International Conference on Language and Automata Theory and Applications*, volume 8370 of *Lecture Notes in Computer Science*, pages 347–359, 2014.

[21] B. Felgenhauer and V. van Oostrom. Proof orders for decreasing diagrams. In *Proc. 24th International Conference on Rewriting Techniques and Applications*, number 21 in Leibniz International Proceedings in Informatics, pages 174–189, 2013.

[22] B. Felgenhauer, H. Zankl, and A. Middeldorp. Proving confluence with layer systems. In *Proc. 31st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 13 of *Leibniz International Proceedings in Informatics*, pages 288–299, 2011.

[23] G. Feuillade, T. Genet, and V. V. T. Tong. Reachability analysis over term rewriting systems. *Journal of Automated Reasoning*, 33:341–383, 2004.

[24] T. Genet. Decidable approximations of sets of descendants and sets of normal forms. In *Proc. 9th International Conference on Rewriting Techniques and Applications*, volume 1379 of *Lecture Notes in Computer Science*, pages 151–165, 1998.

[25] T. Genet, Y.-M. Tang-Talpin, and V. V. T. Tong. Verification of copy-protection cryptographic protocol using approximations of term rewriting systems. In *Proc. WITS'03 (Workshop on Issues in the Theory of Security)*, 2003.

[26] A. Geser. *Relative Termination*. PhD thesis, Universität Passau, 1990. Available as technical report 91-03.

[27] A. Geser, D. Hofbauer, J. Waldmann, and H. Zantema. On tree automata that certify termination of left-linear term rewriting systems. *Information and Computation*, 205(4):512–534, 2007.

[28] G. Godoy, A. Tiwari, and R. Verma. On the confluence of linear shallow term rewrite systems. In *Proc. 20th International Symposium on Theoretical Aspects of Computer Science*, volume 2607 of *Lecture Notes in Computer Science*, pages 85–96, 2003.

[29] G. Godoy, A. Tiwari, and R. Verma. Deciding confluence of certain term rewriting systems in polynomial time. *Annals of Pure and Applied Logic*, 130(1-3):33–59, 2004.

[30] B. Gramlich. Confluence without termination via parallel critical pairs. In *Proc. 21st International Colloquium on Trees in Algebra and Programming*, volume 1059 of *Lecture Notes in Computer Science*, pages 211–225, 1996.

[31] N. Hirokawa and A. Middeldorp. Decreasing diagrams and relative termination. In *Proc. 5th International Joint Conference on Automated Reasoning*, volume 6173 of *Lecture Notes in Artificial Intelligence*, pages 487–501, 2010.

[32] N. Hirokawa and A. Middeldorp. Decreasing diagrams and relative termination. *Journal of Automated Reasoning*, 47(4):481–501, 2011.

[33] N. Hirokawa and A. Middeldorp. Commutation via relative termination. In *Proc. 2nd International Workshop on Confluence*, pages 29–33, 2013.

[34] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980.

[35] B. Jacobs. Involutive categories and monoids, with a GNS-correspondence. *Foundations of Physics*, pages 1–22, 2011.

[36] J. P. Jouannaud and J. Liu. From diagrammatic confluence to modularity. *Theoretical Computer Science*, 464:20–34, 2012.

[37] J.-P. Jouannaud and Y. Toyama. Modular Church-Rosser modulo: The complete picture. *International Journal of Software and Informatics*, 2(1):61–75, 2008.

[38] J.-P. Jouannaud and V. van Oostrom. Diagrammatic confluence and completion. In *Proc. 36th International Colloquium on Automata, Languages and Programming*, volume 5556 of *Lecture Notes in Computer Science*, pages 212–222, 2009.

[39] S. Kahrs. Confluence of curried term-rewriting systems. *Journal of Symbolic Computation*, 19(6):601–623, 1995.

[40] S. Kahrs. Personal communication, January 2011.

[41] R. Kennaway, J. Klop, M. Sleep, and F. Vries. Comparing curried and uncurried rewriting. *Journal of Symbolic Computation*, 21(1):15–39, 1996.

[42] A. Kitahara, M. Sakai, and Y. Toyama. On the modularity of confluent term rewriting systems with shared constructors. *Technical Reports of the Information Processing Society of Japan*, 95(15):11–20, 1995. In Japanese.

[43] J. Klop, A. Middeldorp, Y. Toyama, and R. de Vrijer. Modularity of confluence: A simplified proof. *Information Processing Letters*, 49:101–109, 1994.

[44] D. Knuth and P. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.

[45] M. Korp. *Termination Analysis by Tree Automata Completion*. PhD thesis, University of Innsbruck, 2010.

[46] M. Korp and A. Middeldorp. Match-bounds revisited. *Information and Computation*, 207(11):1259–1283, 2009.

[47] M. Korp, C. Sternagel, H. Zankl, and A. Middeldorp. The Tyrolean Termination Tool 2. In *Proc. 20th International Conference on Rewriting Techniques and Applications*, volume 5595 of *Lecture Notes in Computer Science*, pages 295–304, 2009.

[48] D. Lankford. On proving term rewrite systems are noetherian. Technical Report MTP-3, Louisiana Technical University, Ruston, LA, USA, 1979.

[49] C. Lüth. Compositional term rewriting: An algebraic proof of Toyama's theorem. In *Proc. 7th International Conference on Rewriting Techniques and Applications*, volume 1103 of *Lecture Notes in Computer Science*, pages 261–275, 1996.

[50] A. Middeldorp and H. Zantema. Simple termination of rewrite systems. *Theoretical Computer Science*, 175(1):127–158, 1997.

[51] G. Nelson and D. Oppen. Fast decision procedures based on congruence closure. *Journal of the ACM*, 27(2):356–364, 1980.

[52] M. Newman. On theories with a combinatorial definition of equivalence. *Annals of Mathematics*, 43(2):223–243, 1942.

[53] T. Nipkow, L. Paulson, and M. Wenzel. *Isabelle/HOL – A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer, 2002.

[54] E. Ohlebusch. *Modular Properties of Composable Term Rewriting Systems*. PhD thesis, Universität Bielefeld, 1994.

[55] E. Ohlebusch. On the modularity of confluence of constructor-sharing term rewriting systems. In *Proc. 19th International Colloquium on Trees in Algebra and Programming*, volume 787 of *Lecture Notes in Computer Science*, pages 261–275, 1994.

[56] E. Ohlebusch. Church-Rosser theorems for abstract reduction modulo an equivalence relation. In *Proc. 9th International Conference on Rewriting Techniques and Applications*, volume 1379 of *Lecture Notes in Computer Science*, pages 17–31, 1998.

[57] E. Ohlebusch. *Advanced Topics in Term Rewriting*. Springer, 2002.

[58] S. Okui. Simultaneous critical pairs and Church-Rosser property. In *Proc. 9th International Conference on Rewriting Techniques and Applications*, volume 1379 of *Lecture Notes in Computer Science*, pages 2–16, 1998.

[59] V. van Oostrom. Confluence by decreasing diagrams. *Theoretical Computer Science*, 126(2):259–280, 1994.

[60] V. van Oostrom. *Confluence for Abstract and Higher-Order Rewriting*. PhD thesis, Vrije Universiteit, Amsterdam, 1994.

[61] V. van Oostrom. Developing developments. *Theoretical Computer Science*, 175(1):159–181, 1997.

[62] V. van Oostrom. Confluence by decreasing diagrams – converted. In *Proc. 19th International Conference on Rewriting Techniques and Applications*, volume 5117 of *Lecture Notes in Computer Science*, pages 306–320, 2008.

[63] V. van Oostrom. Modularity of confluence constructed. In *Proc. 4th International Joint Conference on Automated Reasoning*, volume 5195 of *Lecture Notes in Artificial Intelligence*, pages 348–363, 2008.

[64] V. van Oostrom. Confluence via critical valleys. In *Proc. 6th International Workshop on Higher-Order Rewriting*, pages 9–11, 2012.

[65] M. Oyamaguchi and Y. Ohta. A new parallel closed condition for Church-Rosser of left-linear term rewriting systems. In *Proc. 8th International Conference on Rewriting Techniques and Applications*, volume 1232 of *Lecture Notes in Computer Science*, pages 187–201, 1997.

[66] D. Plaisted. Polynomial time termination and constraint satisfaction tests. In *Proc. 5th International Conference on Rewriting Techniques and Applications*, volume 690 of *Lecture Notes in Computer Science*, pages 405–420, 1993.

[67] B. Rosen. Tree-manipulating systems and Church-Rosser theorems. *Journal of the ACM*, 20(1):160–187, 1973.

[68] A. Stump, H. Zantema, G. Kimmell, and R. Omar. A rewriting view of simple typing. *Logical Methods in Computer Science*, 9(1:4):1–29, 2012.

[69] T. Suzuki, T. Aoto, and Y. Toyama. Confluence proofs of term rewriting systems based on persistency. *Computer Software*, 30(3):148–162, 2013. in Japanese.

[70] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[71] R. Thieman, G. Allais, and J. Nagele. On the formalization of termination techniques based on multiset orderings. In *Proc. 23rd International Conference on Rewriting Techniques and Applications*, volume 15 of *Leibniz International Proceedings in Informatics*, pages 339–354, 2012.

[72] R. Thiemann and C. Sternagel. Certification of termination proofs using CeTA. In *Proc. 22nd International Conference on Theorem Proving in Higher Order Logics*, volume 5674 of *Lecture Notes in Computer Science*, pages 452–468, 2009.

[73] A. Tiwari. Deciding confluence of certain term rewriting systems in polynomial time. In *Proc. 17th IEEE Symposium on Logic in Computer Science*, pages 447–457, 2002.

[74] Y. Toyama. On the Church-Rosser property of term rewriting systems. Technical Report 17672, NTT ECL, 1981.

[75] Y. Toyama. On the Church-Rosser property for the direct sum of term rewriting systems. *Journal of the ACM*, 34(1):128–143, 1987.

[76] Y. Toyama. Commutativity of term rewriting systems. In K. Fuchi and L. Kott, editors, *Programming of Future Generation Computers II*, pages 393–407. North-Holland, 1988.

[77] U. Waldmann. Semantics of order-sorted specifications. *Theoretical Computer Science*, 94(1):1–35, 1992.

[78] H. Zankl, B. Felgenhauer, and A. Middeldorp. CSI – A confluence tool. In *Proc. 23rd International Conference on Automated Deduction*, volume 6803 of *Lecture Notes in Artificial Intelligence*, pages 499–505, 2011.

[79] H. Zankl, B. Felgenhauer, and A. Middeldorp. Labelings for decreasing diagrams. *Journal of Automated Reasoning*, 54(2):101–133, 2015.

[80] H. Zantema. Termination of term rewriting: Interpretation and type elimination. *Journal of Symbolic Computation*, 17(1):23–50, 1994.

Appendix A

CSI – A Confluence Tool

CSI is an automatic confluence prover developed in Innsbruck, near the Confluence of the rivers Sill and Inn. The tool has a web presence at

`http://cl-informatik.uibk.ac.at/software/csi/`

where the software can be tried out, and downloaded in binary form and as source code. The tool CSI is based on the termination prover $\mathsf{T_TT_2}$, which is also developed in Innsbruck.

In this appendix, we give a brief overview of CSI. Further information can be found in the $\mathsf{T_TT_2}$ and CSI system descriptions [47, 78].

A.1 Getting Started

The simplest way to use CSI is via the web frontend,[1] where one can enter or upload term rewrite systems in a standard TRS format, and use CSI to check confluence of the given system. For an example, see Figure A.1. Because confluence is an undecidable property, there are three different answers: `YES` if confluence could be established; `NO` if non-confluence was proved; and `MAYBE` (or a timeout) if no definitive answer could be found.

Alternatively, one can use the binaries available from the CSI website or compile CSI from source code. The tool can then be invoked using the `csi.sh` script, see Figure A.2.

The CSI tool has a built-in help, available by running `csi.sh --help`.

A.2 Features

Because CSI is based on $\mathsf{T_TT_2}$, the two tools have a lot of features in common. There are a number of advantages to building a confluence tool on top of a termination tool. First of all, the tools can share fundamentals like representations and parsers for TRSs. Secondly, termination proofs are often part of confluence proofs, for example when using the Knuth-Bendix criterion, or when using labelings (Chapter 4) based on relative termination. The features of CSI are, in a nutshell:

1. It has *processors* that turn input problems into a list of (easier) output problems that need to be solved.

[1] See `http://cl-informatik.uibk.ac.at/software/csi/`

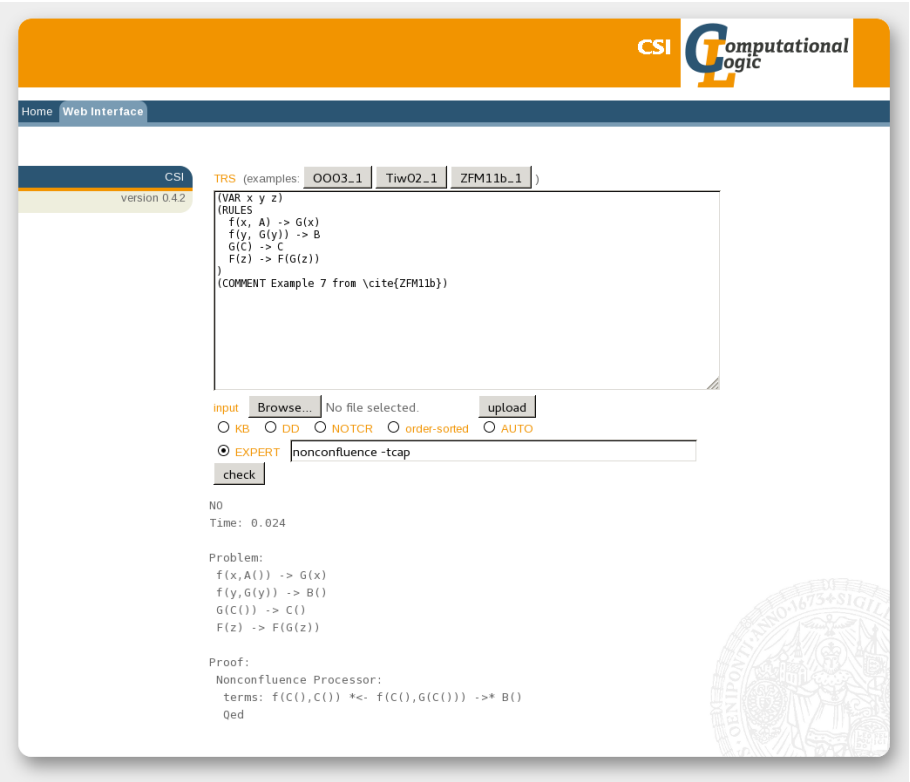


Figure A.1: The CSI web frontend.

2. Processors can be combined into complex strategies using a strategy language.
3. Input in TRS or CPF (XML) format; output of proofs in text or CPF format.

CSI comes with a predefined strategy `AUTO`, which can be found in `cr.conf` in the binary distribution and source code. This strategy has been designed to be flexible and powerful, and it should be sufficient for normal usage. For advanced users and the curious, he remainder of this section gives a glimpse into the design and syntax of the strategy language.

A.2.1 Processors

A processor takes an input problem (which may be a confluence problem, a non-confluence problem, a termination or non-termination problem, and a few other types), and, if successful, produces a (potentially empty) list of subproblems that remain to be solved. Let us describe a selection of processors that are relevant to proving confluence. Note that most processors take options that can be listed by using `csi.sh -s '<processor> -help' foo.trs`. A complete list of processors can be obtained by running `csi.sh -P`.

- `cr -kb`. This processor implements the Knuth-Bendix criterion. It takes a confluence problem as input, and checks whether all its critical pairs are

```
# ./csi.sh 243.trs*
NO

Problem:
 a() -> h(g(b()))
 a() -> h(c())
 b() -> g(b())
 h(g(x)) -> g(h(x))
 g(x) -> h(x)

Proof:
 Nonconfluence Processor:
  terms: h(g(b())) *<- a() ->* h(c())
  Qed

  first automaton:
   final states: {9}
   transitions:
    h(11) -> 11,32,9
    h(32) -> 9*
    h(9) -> 9*
    h(10) -> 10,11,32
    b() -> 10*
    g(32) -> 9*
    g(9) -> 9*
    g(11) -> 32,11
    g(10) -> 10,11

  second automaton:
   final states: {7}
   transitions:
    h(8) -> 7*
    c() -> 8*
```

* `243.trs` is Cops 243 from `http://coco.nue.riec.tohoku.ac.jp/cops/`

Figure A.2: Using CSI from the command line.

joinable. If successful, it returns the same TRS as a termination problem, which can then be dealt with by termination processors.

- `sorted -order`. Perform an order-sorted decomposition. This processor takes a confluence problem as input. If there is an order-sorted sort attachment that allows the TRS to be decomposed, it produces the decomposed TRS as a list of subproblems. Otherwise, it fails.
- `nonconfluence`. Try to establish non-confluence. It takes a non-confluence problem as input and tries to produce a witness for non-confluence, i.e. a conversion $s \leftrightarrow t$ (in practice it only looks for peaks, essentially starting with critical pairs) such that s and t are not joinable. Non-confluence can be established using tcap (`nonconfluence -tcap`) or using tree automata (`nonconfluence -tree`).
- `cr -dup`. This takes a (left-linear) confluence problem as input and tries to join all critical pairs. If successful, it produces a *diagram problem* as output, which consists of $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$, together with a set of joining valleys for the critical pairs.
- `rule_labeling`. This processor works on diagram problems; it computes a rule labeling with the goal of making the critical pair diagrams decreasing. It produces a new diagram problem (with the set of labels extended) as output.
- `decreasing`. This processor takes a diagram problem, checks that there is no termination problem left to solve, and whether all the critical diagrams are joined decreasingly. If that works out, the problem is marked as confluent.

The last three processors may seem obscure; but note that, if they are used in sequence, they implement the rule-labeling technique for left-linear TRSs. (Left-linear TRSs are non-duplicating, hence $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ is trivially terminating.) These three steps are separated so that all labelings from Chapter 4 can be implemented without reimplementing the basic pre- and postprocessing each time.

A.2.2 The Strategy Language

The strategy allows combining processors into more complex processors. Its syntax has the following grammar, where p is a basic processor, n denotes a count, t is a floating point number designating a timeout, and c is a condition.

$$\begin{aligned} s ::= {} & p \mid \texttt{(}\, s \,\texttt{)} \mid s \,\texttt{;}\, s \mid s \mid s \mid s \,\texttt{||}\, s \mid s\,\texttt{?} \mid s\,\texttt{+} \mid s\,\texttt{*} \mid s\,\texttt{*}\,n \mid s\,\texttt{[}\,t\,\texttt{]} \mid s\,\texttt{!} \\ & \mid \texttt{if}\ c\ \texttt{then}\ s\ \texttt{else}\ s \end{aligned}$$

The semantics are as follows.

- p. Invokes a basic processor (e.g., `cr -kb`).

- `(` s `)`. Same as s.
- s_1 `;` s_2. Sequencing. The processor s_1 is executed. If this is successful, s_2 is performed for every resulting subproblem.
- s_1 `|` s_2. Alternative. s_1 is tried; if s_1 is unsuccessful, s_2 is tried instead.
- s_1 `||` s_2. Parallel alternative. The subprocessors s_1 and s_2 are tried in parallel. The combined processor succeeds if either subprocessor succeeds, and the resulting subproblems are taken from the first subprocessor to succeed.
- s `?`. Option. The processor s is executed. If it succeeds then the subproblems produced by s are taken as subproblems of s `?`. Otherwise, the input problem is returned unmodified.
- s `*`. Repeat processor multiple times. The processor s is executed. If it succeeds, the produced subproblems are all subjected to s `*`, whereas upon failure, the input problem is returned. For example, `sorted -order*` will decompose a confluence problem as much as possible by iterated application of the order-sorted decomposition.
- s `+`. Repeat one or more times. This is similar to s `*`, but the combined processor fails if the first application of s fails.
- s `*` n. This is also similar to s `*`, but puts an upper bound of the number of times s is applied to subproblems.
- s `[` t `]`. Timeout. The processor s is run for at most t seconds. If t seconds elapse without s succeeding, the processor is terminated, and s `[` t `]` fails.
- s `!`. Finalize. This checks that after s has run successfully, no subproblems are left, and fails otherwise.
- `if` c `then` s_1 `else` s_2. Conditional. This construct can be used to check various properties of the input problem, like `trs` (verify that the input problem a TRS), `left-linear` (verify that the input TRS left-linear), etc. A complete list of available predicates is printed by `csi.sh -P`.

A.3 Experiments

In this section, we try to assess the power of the techniques described in this thesis, and our tool. To this end, we performed experiments using the 276 TRSs from the Cops database[2] of confluence problems, as of January 2015.

Figure A.3 shows the gain in power by adding various techniques to a basic strategy that essentially consists of the Knuth-Bendix criterion, strong closedness (for linear TRSs) and parallel closedness (for left-linear TRSs). We then enabled the order-sorted decomposition (os) based on the layer framework (Chapter 5),

[2] `http://coco.nue.riec.tohoku.ac.jp/cops/`

	basic	+os	+lab	+os,lab	+ground	ncr	+aut
yes	73	80	104	114	84	0	0
no	0	0	0	0	7	29	47
maybe	203	196	172	162	185	247	229

Figure A.3: Power gain through various techniques in CSI.

	full	-os	-lab	-par	-ground	-aut
yes	155	150	139	154	154	155
no	47	47	47	47	47	31
maybe	74	79	90	75	75	90

Figure A.4: Power loss through various techniques in CSI.

the labeling technique (lab) for decreasing diagrams (Chapter 4), and the decision procedure for ground TRSs (Chapter 6). Furthermore we compare a basic non-confluence technique (with the non-joinability test based on tcap) to the automata based non-confluence check.

In a similar vain, Figure A.4 shows the loss in power when certain techniques are removed from CSI's full strategy. The techniques removed are the order-sorted decomposition (os), the labeling (lab), only the parallel version of the labeling (par), the decision procedure for ground TRSs (ground), and the automata based non-confluence check (aut).

It is interesting to note that the full strategy can handle almost all ground TRSs handled by the decision procedure. The remaining example is

$$\mathsf{a} \to \mathsf{b} \qquad \mathsf{a} \to \mathsf{f}(\mathsf{a}) \qquad \mathsf{b} \to \mathsf{f}(\mathsf{f}(\mathsf{b})) \qquad \mathsf{f}^{64}(\mathsf{b}) \to \mathsf{b}$$

which was specifically designed to thwart confluence proofs that work by joining critical pairs. (It is $\mathcal{R}_{64}$ from Section 6.3.)

The labeling for parallel steps is also fairly weak compared to the standard labeling technique. The example gained is Example 4.56, which has the redeeming property of not being constructed for this purpose; furthermore, none of the competitor tools (ACP, CoLL, Saigawa) could show confluence of that TRS at the time of writing this.

A typical example gained by the order-sorted decomposition is

$$\mathsf{f}(x,y) \to x \qquad \mathsf{f}(x,y) \to \mathsf{f}(x,\mathsf{g}(y)) \qquad \mathsf{g}(x) \to \mathsf{h}(x)$$
$$\mathsf{F}(\mathsf{g}(x),x) \to \mathsf{F}(x,\mathsf{g}(x)) \qquad \mathsf{F}(\mathsf{h}(x),x) \to \mathsf{F}(x,\mathsf{h}(x))$$

from [4], where after decomposing the TRS, rules involving f end up in a different component than those involving F, separating the (non-left-linear, non-terminating) system into a left-linear TRS and a terminating TRS, which are two classes for which powerful confluence techniques exist.

Finally, a system where the tree automata based non-joinability criterion is beneficial is

$$\mathsf{f}(\mathsf{a},\mathsf{a}) \to \mathsf{c} \qquad \mathsf{f}(\mathsf{b},x) \to \mathsf{f}(x,x) \qquad \mathsf{f}(x,\mathsf{b}) \to \mathsf{f}(x,x) \qquad \mathsf{a} \to \mathsf{b}$$

	CSI	ACP	CoLL	Saigawa
yes	155	186	63	126
no	47	52	0	27
maybe	74	26	141	82

Figure A.5: Comparing CSI to ACP, CoLL and Saigawa.

from [32], which has a non-joinable peak

$$\mathsf{c} \leftarrow \mathsf{f}(\mathsf{a},\mathsf{a}) \rightarrow \mathsf{f}(\mathsf{b},\mathsf{a}) \rightarrow \mathsf{f}(\mathsf{b},\mathsf{b})$$

where the right term $\mathsf{f}(\mathsf{b},\mathsf{b})$ can only reach itself. However, since that term allows rule applications at the root, tcap is not strong enough to handle this TRS, while tree automata have no trouble at all.

As a final comparison, Figure A.5 compares CSI to its competitor tools, showing that it currently holds a solid second place.