Logical Methods in Computer Science
Vol. 13(2:4)2017, pp. 1–27
www.lmcs-online.org

Submitted May 1, 2016
Published May 11, 2017

CERTIFYING CONFLUENCE PROOFS VIA RELATIVE TERMINATION AND RULE LABELING [*]

JULIAN NAGELE [a], BERTRAM FELGENHAUER [b], AND HARALD ZANKL [c]

[a,b] Department of Computer Science, University of Innsbruck, Austria
e-mail address: {julian.nagele, bertram.felgenhauer}@uibk.ac.at

[c] Innsbruck, Austria
e-mail address: hzankl@gmail.com

Abstract. The rule labeling heuristic aims to establish confluence of (left-)linear term rewrite systems via decreasing diagrams. We present a formalization of a confluence criterion based on the interplay of relative termination and the rule labeling in the theorem prover Isabelle. Moreover, we report on the integration of this result into the certifier CeTA, facilitating the checking of confluence certificates based on decreasing diagrams. The power of the method is illustrated by an experimental evaluation on a (standard) collection of confluence problems.

1. Introduction

One of the most important properties in program verification is confluence, which ensures that different computation paths produce the same result, i.e., that normal forms are unique. Consequently automatable formal methods that can analyze confluence of programs are of great interest. One of the most widely used and successful formalisms for analyzing confluence is rewriting, a conceptually simple but powerful abstract model of computation, which uses directed equations, i.e., rewrite rules to replace equals by equals. The recent advances in confluence research have culminated in the Confluence Competition (CoCo) [2], where automated tools try to establish/refute confluence of rewrite systems.[1] As demonstrated in the termination competition,[2] rewriting is suitable for analyzing programs written in real-world programming languages such as C, Haskell, and Java. As the proofs produced by automated tools are often complex and large, there is interest in verifying them using an independent certifier. A certifier is a different kind of automated tool that reads proof certificates, which are for instance produced by automated confluence tools, and either accepts them as correct or rejects them as erroneous, thereby increasing credibility of

Key words and phrases: term rewriting, confluence, decreasing diagrams, certification.
[*] A preliminary version of this article appeared in RTA 2015.
[a,b] This research is supported by FWF (Austrian Science Fund) project P27528.

[1]`http://coco.nue.riec.tohoku.ac.jp`
[2]`http://termination-portal.org`

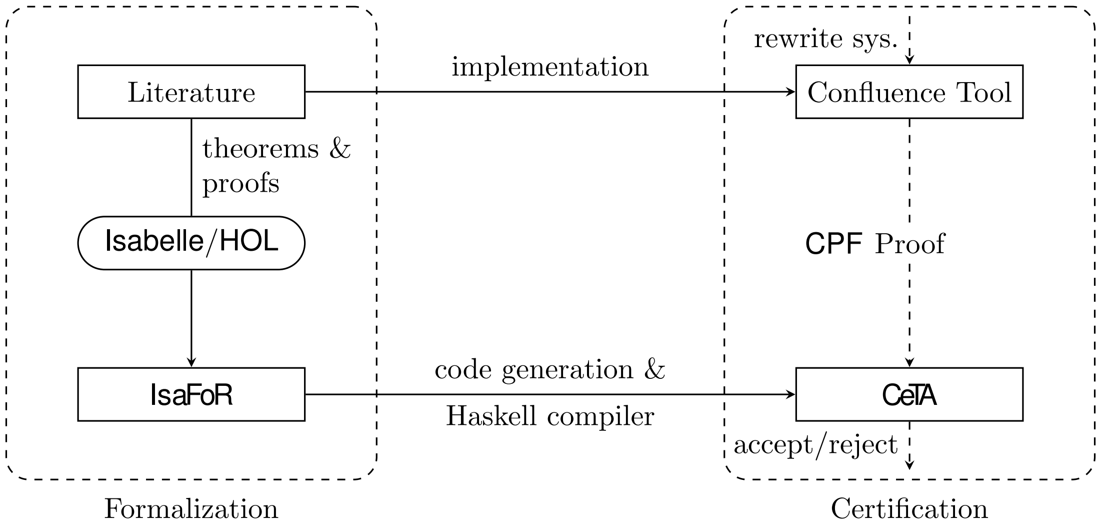


FIGURE 1. Certification of confluence proofs.

the proofs found by the confluence tools. To ensure correctness of the certifier itself, the predominant solution is to use proof assistants like Coq or Isabelle to first formalize the underlying theory in the proof assistant and then use this formalization to obtain verified, executable check functions for inspecting the certificates.

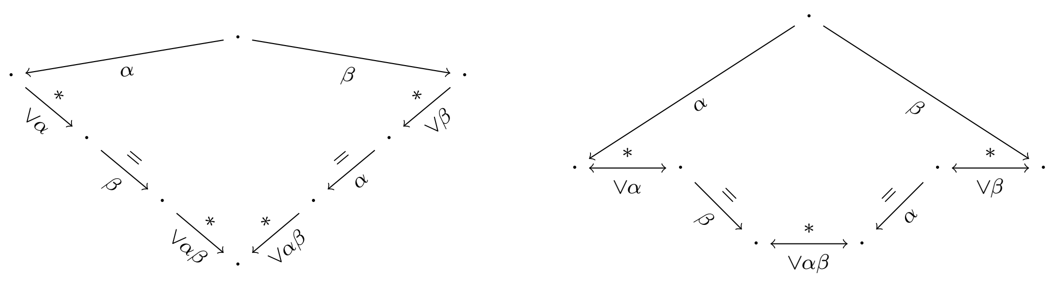
In this article we are concerned with the formalization and certification of a recent confluence result in the proof assistant Isabelle/HOL [22]. We based our formalization on the Isabelle Formalization of Rewriting (IsaFoR) [36], adding approximately 4200 lines of Isabelle code in Isar style. IsaFoR contains executable check functions for each formalized proof technique together with formal proofs that whenever such a check is accepted, the technique is applied correctly. The certifier CeTA is a trusted Haskell program obtained from IsaFoR, using Isabelle's code-generation facility [10], which ensures partial correctness [11]. CeTA reads and checks proof certificates in the certification problem format (CPF) [32].[3] The big picture of this automated, reliable confluence analysis is shown in Figure 1, the resources available from the URLs given in Footnotes 1 and 3 provide more details.

Decreasing diagrams [23] provide a complete characterization of confluence for abstract rewrite systems whose convertibility classes are countable. This confluence criterion states that a labeled abstract rewrite system is confluent if each of its local peaks can be *joined decreasingly* in the shape of Figure 2(A), where $\vee\alpha$ indicates that any label below α may be used, and $>$ is a well-founded order on the labels. We will introduce these notions formally in Section 2. As a criterion for abstract rewrite systems, decreasing diagrams can be applied to first- and higher-order rewriting, including term rewriting and the λ-calculus.

In this article we describe our formalization in Isabelle[4] of a recent powerful confluence result for term rewrite systems, exploiting relative termination and decreasing diagrams. Our aim was to formalize [41, Corollary 16] for a specific labeling (which assigns labels to rewrite steps that are suitable for establishing confluence by decreasing diagrams) culminating in Theorem 4.2. Actually the formal proof of Theorem 4.2 is obtained by instantiating the

[3] IsaFoR/CeTA and CPF are available at `http://cl-informatik.uibk.ac.at/software/ceta/`.

[4] In this article, the verb *formalize* refers exclusively to mechanized definitions and proofs in Isabelle/HOL.



(A) Valley version (B) Conversion version

Figure 2. Locally decreasing diagrams.

more general result of Theorem 5.10 appropriately. Hence, confluence proofs according to these theorems are now checkable by CeTA. We achieved this via the following steps:

- Formalize local decreasingness for abstract rewrite systems following [6, 7] (Section 3).
- Perform a detailed analysis of how local peaks can be joined for (left-)linear term rewrite systems. Note that this analysis has been performed in Isabelle and hence IsaFoR has been extended appropriately (Section 4.1).
- Use the detailed analysis of local peaks to formalize the notion of a labeling and a confluence result for term rewrite systems parametrized by a labeling (Section 4.2). In this way it is ensured that the formalization is reusable for other labelings stemming from [41].
- Instantiate the result from Section 4.2 to obtain concrete confluence results. We demonstrate how this instantiation can be done, culminating in a formal proof of Theorem 5.10 (Section 5).
- Finally we made our formalization executable to check proof certificates: we suitably extended CPF to represent proofs according to Theorem 5.10 and implemented dedicated check functions in our formalization, enabling CeTA to inspect, i.e., certify, such confluence proofs (Section 5.3).

To indicate which parts of the formalization are contributed by this article, it is stated explicitly whenever existing results from IsaFoR are (re)used. More generally, all results displayed as definition, lemma, or theorem in explicit LaTeX environments have been formalized and are thus contributions of this article and as such new additions to IsaFoR. In the presentation we use different levels of abstraction. Code listings represent the formalization literally as Isabelle source code, LaTeX environments display the formalization as written in Isabelle (but based on mathematical notation) and the intermediate text provides a high-level description of the formalization in words. The formalization constitutes our main contribution; it is not our goal to advance the theory of rewriting in this work.

The remainder of this article is organized as follows. Preliminaries are introduced in the next section. In Section 3 we describe the formalization of decreasing diagrams for abstract rewrite systems. Based on the results for abstract rewriting and the notion of a labeling, Section 4 formulates conditions that limit the attention to critical peaks rather than arbitrary local peaks in the case of first-order term rewriting. Section 5 instantiates these results with concrete labelings to obtain corollaries that ensure confluence. Section 6 presents an experimental evaluation, before we conclude in Section 7. The full formalization is available from the URL in Footnote 3.

This article is an extended version of [20]. Apart from improvements in the presentation, new contributions comprise the support for the *conversion version* of decreasing diagrams,[5] cf. [24], culminating in the new result of Theorem 5.10. The proof of Theorem 5.10 is more involved than the one of Theorem 4.2, the target of [20], which is obtained here by instantiating Theorem 5.10 appropriately. Also the formalization described in Section 3 is new. Compared to [20], the formalization of decreasing diagrams in Section 3 uses an algebraic approach based on involutive monoids, and is significantly smaller than the previous one. The main motivation for the replacement was to cover decreasing diagrams for commutation and Church-Rosser modulo, but those extensions are part of ongoing IsaFoR developments and are out of scope for this article. However, as is common software engineering practice, we eliminated all uses of the formalization presented in [20, Section 3], thus reducing future maintenance effort in IsaFoR.

2. Preliminaries

We assume familiarity with rewriting [4, 35] and decreasing diagrams [23]. Basic knowledge of Isabelle [22] is not essential but experience with an interactive theorem prover might be helpful.

Let $\mathcal{F}$ be a signature and $\mathcal{V}$ a set of variables disjoint from $\mathcal{F}$. By $\mathcal{T}(\mathcal{F},\mathcal{V})$, we denote the set of *terms* over $\mathcal{F}$ and $\mathcal{V}$. Let $\Box$ be a function symbol of arity zero and $\Box \notin \mathcal{F}$. A *context* is a term in $(\mathcal{F} \cup \{\Box\}, \mathcal{V})$, with a single occurrence of $\Box$, dubbed the *context hole*. *Positions* are strings of positive natural numbers, i.e., elements of $\mathbb{N}_+^*$. We write $q \leqslant p$ if $qq' = p$ for some position q', in which case $p \backslash q$ is defined to be q'. Furthermore $q < p$ if $q \leqslant p$ and $q \neq p$. Finally, $q \parallel p$ if neither $q \leqslant p$ nor $p < q$. Positions are used to address subterm occurrences. The *set of positions* of a term t is defined as $\mathcal{P}\mathsf{os}(t) = \{\epsilon\}$ if t is a variable and as $\mathcal{P}\mathsf{os}(t) = \{\epsilon\} \cup \{iq \mid 1 \leqslant i \leqslant n \text{ and } q \in \mathcal{P}\mathsf{os}(t_i)\}$ if $t = f(t_1,\ldots,t_n)$. The *subterm* of t at position $p \in \mathcal{P}\mathsf{os}(t)$ is defined as $t|_p = t$ if $p = \epsilon$ and as $t|_p = t_i|_q$ if $p = iq$ and $t = f(t_1,\ldots,t_n)$. We write $s[t]_p$ for the result of replacing the occurrence of $s|_p$ at position p with t in s. The set of *function symbol positions* $\mathcal{P}\mathsf{os}_{\mathcal{F}}(t)$ is $\{p \in \mathcal{P}\mathsf{os}(t) \mid t|_p \notin \mathcal{V}\}$ and $\mathcal{P}\mathsf{os}_{\mathcal{V}}(t) = \mathcal{P}\mathsf{os}(t) \setminus \mathcal{P}\mathsf{os}_{\mathcal{F}}(t)$.

A *rewrite rule* is a pair of terms (l,r), written $l \to r$.[6] A rewrite rule $l \to r$ is *duplicating* if $|l|_x < |r|_x$ for some $x \in \mathcal{V}$, where the expression $|t|_x$ indicates the *number of occurrences* of the variable x in term t. A *term rewrite system* (TRS) is a signature together with a set of rewrite rules over this signature. In the sequel, signatures are left implicit. By $\mathcal{R}_{\mathsf{d}}$ and $\mathcal{R}_{\mathsf{nd}}$, we denote the *duplicating* and *non-duplicating rules* of a TRS $\mathcal{R}$, respectively. A *rewrite relation* is a binary relation on terms that is closed under contexts and substitutions. For a TRS $\mathcal{R}$ we define $\to_{\mathcal{R}}$ (often written as $\to$) to be the smallest rewrite relation that contains $\mathcal{R}$. As usual $\to^=$, $\to^+$, and $\to^*$ denote the reflexive, transitive, and reflexive and transitive closure of $\to$, respectively, while $\to^n$ denotes the n-fold composition of $\to$. A *relative TRS* $\mathcal{R}/\mathcal{S}$ is a pair of TRSs $\mathcal{R}$ and $\mathcal{S}$ with the induced rewrite relation $\to_{\mathcal{R}/\mathcal{S}} = \to_{\mathcal{S}}^* \cdot \to_{\mathcal{R}} \cdot \to_{\mathcal{S}}^*$, where $\cdot$ denotes relation composition. Sometimes we identify a TRS $\mathcal{R}$ with the relative TRS

[5]The conversion version of decreasing diagrams allows the use of conversions instead of forward rewrite sequences for parts with small labels, which can reduce the size of the diagrams considerably. This is illustrated in Figure 2(B). Note that the left diagram is an instance of the right one.

[6]We do not require the common *variable conditions*, i.e., the restriction that l is not a variable and all variables in r are contained in l.

$\mathcal{R}/\varnothing$ and vice versa, which is justified by $\to_{\mathcal{R}/\varnothing} = \to_{\mathcal{R}}$. A TRS $\mathcal{R}$ is *terminating (relative to a TRS $\mathcal{S}$)* if $\to_{\mathcal{R}}$ ($\to_{\mathcal{R}/\mathcal{S}}$) is well-founded. A TRS $\mathcal{R}$ is *confluent* if ${}_{\mathcal{R}}^{*}\!\!\leftarrow \cdot \to_{\mathcal{R}}^{*} \subseteq \to_{\mathcal{R}}^{*} \cdot {}_{\mathcal{R}}^{*}\!\!\leftarrow$.

A *critical overlap* $(l_1 \to r_1, p, l_2 \to r_2)_\mu$ of a TRS $\mathcal{R}$ consists of variants $l_1 \to r_1$ and $l_2 \to r_2$ of rewrite rules in $\mathcal{R}$ without common variables, a position $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(l_2)$, and a most general unifier μ of l_1 and $l_2|_p$. From a critical overlap $(l_1 \to r_1, p, l_2 \to r_2)_\mu$ we obtain a *critical peak* $l_2\mu[r_1\mu]_p \leftarrow l_2\mu \to r_2\mu$ and a *critical pair* $l_2\mu[r_1\mu]_p \leftarrow\!\rtimes\!\!\to r_2\mu$.[7]

If $l \to r \in \mathcal{R}$, p is a position, and σ is a substitution we call the triple $\pi = \langle p, l \to r, \sigma\rangle$ a *redex pattern*, and write p_π, l_π, r_π, σ_π for its position, left-hand side, right-hand side, and substitution, respectively. We write $\to^{\pi}$ (or $\to^{p_\pi, l_\pi \to r_\pi, \sigma_\pi}$) for a rewrite step at position p_π using the rule $l_\pi \to r_\pi$ and the substitution σ_π. A redex pattern π *matches* a term t if $t|_{p_\pi} = l_\pi\sigma_\pi$, in which case $t|_{p_\pi}$ is called a *redex*. Let π_1 and π_2 be redex patterns that match a common term. They are called *parallel*, written $\pi_1 \parallel \pi_2$, if $p_{\pi_1} \parallel p_{\pi_2}$. If $P = \{\pi_1, \pi_2, \ldots, \pi_n\}$ is a set of pairwise parallel redex patterns matching a term t, we denote by $t \mathrel{+\!\!\!\!\twoheadrightarrow}^{P} t'$ the *parallel rewrite step* from t to t' by P, i.e., $t \to^{\pi_1} \cdot \to^{\pi_2} \cdot \ldots \cdot \to^{\pi_n} t'$.

An *abstract rewrite system* (ARS) is a binary relation $\to$. Let I be an index set. We write $\{\to_\alpha\}_{\alpha \in I}$ to denote the ARS $\to$ where $\to$ is the union of $\to_\alpha$ for all $\alpha \in I$. Let $\{\to_\alpha\}_{\alpha \in I}$ be an ARS and let $>$ and $\geqslant$ be relations on I. Two relations $>$ and $\geqslant$ are called *compatible* if $\geqslant \cdot > \cdot \geqslant \subseteq >$. Given a relation $\succcurlyeq$ we write $\to_{\curlyvee \alpha_1 \cdots \alpha_n}$ for the union of all $\to_\beta$ with $\alpha_i \succcurlyeq \beta$ for some $1 \leqslant i \leqslant n$. Similarly, $\curlyvee S$ is the set of all β such that $\alpha \succcurlyeq \beta$ for some $\alpha \in S$. We call α and β *extended locally decreasing* (for $>$ and $\geqslant$) if ${}_{\alpha}\!\leftarrow \cdot \to_\beta \subseteq \leftrightarrow_{\curlyvee\alpha}^{*} \cdot \to_{\curlyvee\!\!\!/\,\beta}^{=} \cdot \leftrightarrow_{\curlyvee\alpha\beta}^{*} \cdot {}_{\curlyvee\!\!\!/\,\alpha}^{=}\!\!\leftarrow \cdot \leftrightarrow_{\curlyvee\beta}^{*}$. If there exist a well-founded order $>$ and a preorder $\geqslant$, such that $>$ and $\geqslant$ are compatible, and α and β are extended locally decreasing for all $\alpha, \beta \in I$ then the ARS $\{\to_\alpha\}_{\alpha \in I}$ is *extended locally decreasing* (for $>$ and $\geqslant$). We call an ARS *locally decreasing* (for $>$) if it is extended locally decreasing for $>$ and $=$, where the latter is the identity relation. In the sequel, we often refer to extended locally decreasing as well as to locally decreasing just by decreasing, whenever the context clarifies which concept is meant or the exact meaning is irrelevant. In the literature the above condition is referred to as *the conversion version of decreasing diagrams*. In its *valley version*, the condition reads as follows: ${}_{\alpha}\!\leftarrow \cdot \to_\beta \subseteq \to_{\curlyvee\alpha}^{*} \cdot \to_{\curlyvee\!\!\!/\,\beta}^{=} \cdot \to_{\curlyvee\alpha\beta}^{*} \cdot {}_{\curlyvee\alpha\beta}^{*}\!\!\leftarrow \cdot {}_{\curlyvee\!\!\!/\,\alpha}^{=}\!\!\leftarrow \cdot {}_{\curlyvee\beta}^{*}\!\!\leftarrow$. In the sequel we always refer to the conversion version of decreasing diagrams, except when stated otherwise explicitly.

In IsaFoR, the above notions are formalized essentially as described above. However, since the underlying logic, Isabelle/HOL, is typed, all notions are parametrized by some base types. For example, ARSs have an associated type for the objects being rewritten. In the case of $\{\to_\alpha\}_{\alpha \in I}$, there is one object type shared by all relations $\to_\alpha$ and another type for the labels. Terms and TRSs are parametric in the types of function symbols and variable symbols. Fortunately, types have almost no impact on the proofs. We will, therefore, not mention any types in the remainder of this article.

[7]We based the definition of critical pairs on the one in IsaFoR. Note that IsaFoR does not exclude root overlaps of a rule with (a variant of) itself as is commonly done in the literature. This allows to dispense with the variable condition that all variables in the right-hand side of a rule must also occur on the left. Moreover, if a TRS does satisfy the condition then all extra critical pairs that would normally be excluded are trivial.

3. Formalized Confluence Results for Abstract Rewriting

This section is concerned with the formalization of the following result from [12, Theorem 2]:

Lemma 3.1. *Every extended locally decreasing ARS is confluent.* □

It is interesting to note that in [13], the journal version of [12], extended local decreasingness is avoided by employing the *predecessor labeling.* Originally the authors used the *source labeling*, wherein a step $s \to t$ is labeled by s, and $\geqslant = \to^*$ for the weak order. Consequently, any rewrite step $s' \to t'$ with $s \to^* s'$ has a label s' with $s \geqslant s'$. In the predecessor labeling, a rewrite step can be labeled by an arbitrary predecessor, i.e., we have $s \to_u t$ if $u \to^* s$. Now, if $s \to_u t$ and $s \to^* s' \to t'$, we may label the step from s' to t' by u, the same label as $s \to_u t$. In this way, the need for the weak comparison $\geqslant$, and hence *extended* local decreasingness, is avoided. As the proof of Lemma 3.1 (see below) demonstrates, this idea works in general, i.e., extended local decreasingness can be traded for assigning more than one label to each rewrite step. In the context of automation, however, assigning just one (computable) label to every rewrite step is very attractive, as confluence tools must show critical peaks decreasing and confluence certifiers must check the related proof certificates.

In the formalization, we first establish results for local decreasingness, and then show that every extended locally decreasing ARS is also locally decreasing. For the first part, we closely follow the development of the *monotonic* order in [7]. Note that the proof development in [20] was based on [38], which formalized the valley and conversion versions of decreasing diagrams. The formalization presented in this section goes beyond that, providing results for commutation and for Church-Rosser modulo, see [7]. Despite covering more results, the development is still shorter than the replaced one. The key proof technique is that conversions are mapped to *French strings*, which abstract from the objects, keeping only the labels and directions of the rewrite steps indicated by accents: an acute accent ($\acute{\alpha}$) denotes a leftward step, while a grave accent ($\grave{\alpha}$) marks a rightward step. For example, the conversion $\mathsf{a} \; {}_{\alpha}\!\leftarrow \mathsf{b} \to_{\beta} \mathsf{c} \; {}_{\alpha}\!\leftarrow \mathsf{b} \to_{\gamma} \mathsf{d}$ would be represented by $\acute{\alpha}\grave{\beta}\acute{\alpha}\grave{\gamma}$. In fact we directly formalize *Greek strings*, which additionally allow macron accents ($\bar{\alpha}$) that may be used to represent equational steps for results on Church-Rosser modulo (cf. [7, Section 5]). This extension is not used for proving Lemma 3.1. Greek strings are compared by a well-founded order that shows that pasting local diagrams (which corresponds to replacing a substring $\acute{\alpha}\grave{\beta}$ by a new string representing the joining conversion in the decreasing diagram) terminates. Because Greek strings are simply lists of Greek letters, we can leverage existing results for lists from Isabelle/HOL, which is one reason why the formalization presented here is shorter. In contrast, if one were to work directly on conversions, the proofs would become cluttered by preconditions whenever two conversions are concatenated, stating that the last object of the first conversion must equal the initial object of the second conversion. Another reason that the formalization is now shorter is that the new proofs are much more algebraic (whereas diagram pasting as formalized in [38] has more of a geometric flavor), and thus better suited for interactive theorem provers.

The full formalization of Lemma 3.1 is part of the Archive of Formal Proofs [6]. There are few differences between the formalization and the paper version [7], and they are mostly of technical nature. For example, [7, Definition 19], which defines $\gg_{\bullet}$ (a well-founded order on Greek strings) in terms of $>$ (a fixed well-founded order on labels), is presented as a *"proper recursive definition"* in the paper. This idea is hard to convey to Isabelle (it would require an elaborate termination proof), so in the formalization, $\gg_{\bullet}$ is instead defined by its characterization as the least fixed point of the map $\Lambda_{>}$ from the proof of [7, Lemma 21],

```
adj_msog xs zs l =
   (case l of (y, ys) ⇒
                (y, case fst y of Acute ⇒ ys @ zs | Grave ⇒ xs @ ys | Macron ⇒ ys))
ms_of_greek as =
   mset (map (λx. case x of (xs, y, zs) ⇒ adj_msog xs zs (y, [])) (list_splits as))
letter_less r = {(a, b). (snd a, snd b) ∈ r}
nest r s = {(a, b). (ms_of_greek a, ms_of_greek b) ∈ mult (letter_less r <*lex*> s)}
greek_less r = lfp (nest r)
```

Listing 1. Definition of $(\gg_\bullet)^{-1}$ = ***greek_less*** $(<)$

(which states that $\gg_\bullet$ is a strict partial order). The resulting Isabelle definition is given in Listing 1. In a nutshell, ***ms_of_greek*** maps a Greek string x (which is a list of pairs of accents and labels) to a corresponding multiset ($\langle x\rangle^g$ in [7]), using auxiliary functions ***adj_msog*** and ***list_splits***. The latter returns all possible ways of splitting a list into a prefix, a single element and a suffix, e.g., ***list_splits*** $[1,2,3] = [([\,],1,[2,3]),([1],2,[3]),([1,2],3,[\,])]$. Next, (***letter_less*** $(<)$) lifts $<$ to Greek letters, and (***nest*** $(<)$) corresponds to $\Lambda_>$, employing the multiset extension ***mult*** and a lexicographic product. Finally, (***greek_less*** $(<)$) computes $(\gg_\bullet)^{-1}$ as a least fixed point using ***lfp***, which is part of Isabelle/HOL's library.

A notable difference between the paper and the formalization concerns the proof of well-foundedness of the order $\gg_\bullet$ on Greek strings, where instead of simple termination, we employ Higman's lemma, which is more fundamental and was already available in the Archive of Formal Proofs [29, 30].

Theorem 3.2 (cf. [7, Theorem 23])**.** *If $>$ is a well-founded order on labels, then $\gg_\bullet$ is a well-founded, monotonic, partial order on Greek strings.*

Proof. We only show well-foundedness of $\gg_\bullet$ here. Because $>$ is well-founded, it can be extended to a well-order $>'$. Then $\geqslant'$ is a well-quasi-order. When applied to Greek letters, $\geqslant'$ compares the underlying labels. This lifted version is also a well-quasi-order. In the corresponding order on Greek strings $\gg'_\bullet$ we have $\hat\alpha \gg'_\bullet \epsilon$ for all labels α, and $\hat\alpha \gg'_\bullet \hat\beta$ whenever $\hat\alpha >' \hat\beta$. By monotonicity [7, Lemma 22], this implies $\geqslant'_{\mathsf{emb}} \subseteq (\gg'_\bullet)^=$, where $\geqslant'_{\mathsf{emb}}$ is the *string embedding relation* induced by $\succcurlyeq'$. By Higman's lemma, $\geqslant'_{\mathsf{emb}}$ is a well-quasi-order, and therefore $(\gg'_\bullet)^=$ is also a well-quasi-order. The strict part of $(\gg'_\bullet)^=$ is $\gg'_\bullet$, which is therefore a well-founded order, and so is its subset $\gg_\bullet$. □

In the rest of this section, we describe our formalized proof of Lemma 3.1. Given an ARS that is extended locally decreasing for $>$ and $\geqslant$, the proof in [12] constructs a single order $\succ$ on sets of labels and establishes local decreasingness of the ARS for $>$. We do the same, but use a simplified proof for the following key lemma, which allows us to use $>$ directly.

Lemma 3.3. *Every extended locally decreasing ARS is locally decreasing.*

The idea is to set $\Rightarrow_\alpha := \to_{\vee\!\!\!/\alpha}$ and show that $\Rightarrow$ is locally decreasing.[8] This establishes the claim because $\bigcup_{\alpha\in I} \Rightarrow_\alpha = \bigcup_{\alpha\in \vee\!\!\!/ I} \to_\alpha = \bigcup_{\alpha\in I}\to_\alpha$, and therefore, $\Rightarrow = \{\Rightarrow_\alpha\}_{\alpha\in I}$ and $\to = \{\to_\alpha\}_{\alpha\in I}$ are the same ARS, labeled in two different ways. The next example demonstrates some peculiarities of this approach.

[8]Compared to [12], we use $\vee\!\!\!/\alpha$ instead of $C_\alpha = (\vee\!\!\!/\alpha) \setminus (\vee\alpha)$.

$$\begin{tikzcd}
& \mathsf{a} \arrow[dl, "1"'] \arrow[dr, "2"] & \\
\mathsf{b} \arrow[dr, "1.5"'] & & \mathsf{c} \arrow[dl, "1"] \\
& \mathsf{d} &
\end{tikzcd}$$

(A) Extended locally decreasing

$$\begin{tikzcd}
& \mathsf{a} \arrow[dl, Rightarrow, "1"'] \arrow[dr, Rightarrow, "2"] & \\
\mathsf{b} \arrow[dr, Rightarrow, "2"'] & & \mathsf{c} \arrow[dl, Rightarrow, "1"] \\
& \mathsf{d} &
\end{tikzcd}$$

(B) Locally decreasing

$$\begin{tikzcd}
& \mathsf{a} \arrow[dl, Rightarrow, "1.5"'] \arrow[dr, Rightarrow, "2"] & \\
\mathsf{b} \arrow[dr, Rightarrow, "2"'] & & \mathsf{c} \arrow[dl, Rightarrow, "1.5"] \\
& \mathsf{d} &
\end{tikzcd}$$

(C) Locally decreasing

FIGURE 3. (Extended) locally decreasing peaks.

Example 3.4. Consider the ARS $\{\to_\alpha\}_{\alpha\in\{1,1.5,2\}}$ where $\to_1 = \{(\mathsf{a},\mathsf{b}),(\mathsf{c},\mathsf{d})\}$, $\to_{1.5} = \{(\mathsf{b},\mathsf{d})\}$, and $\to_2 = \{(\mathsf{a},\mathsf{c})\}$. This ARS is extended locally decreasing for the orders $> := \{(x,y) \mid x,y \in \mathbb{Q}_{\geqslant 0}, x-y \geqslant 1\}$ and $\geqslant_{\mathbb{Q}}$, as depicted in Figure 3(A). We have $\Rightarrow_2 = \{(\mathsf{a},\mathsf{b}),(\mathsf{c},\mathsf{d}),(\mathsf{a},\mathsf{c}),(\mathsf{b},\mathsf{d})\}$, $\Rightarrow_{1.5} = \{(\mathsf{b},\mathsf{d}),(\mathsf{a},\mathsf{b}),(\mathsf{c},\mathsf{d})\}$, and $\Rightarrow_1 = \{(\mathsf{a},\mathsf{b}),(\mathsf{c},\mathsf{d})\}$, where e.g. $\to_{1.5} \subseteq \Rightarrow_2$ since $2 \geqslant_{\mathbb{Q}} 1.5$. Consequently, $\Rightarrow\, = \,\Rightarrow_1 \cup \Rightarrow_{1.5} \cup \Rightarrow_2$. To establish local decreasingness of the related ARS $\{\Rightarrow_\alpha\}_{\alpha\in\{1,1.5,2\}}$ the peak $\mathsf{b}\ {}_1\!\Leftarrow \mathsf{a} \Rightarrow_2 \mathsf{c}$ (emerging from $\mathsf{b}\ {}_1\!\leftarrow \mathsf{a} \to_2 \mathsf{c}$) must be considered, which can be closed in a locally decreasing fashion via $\mathsf{b} \Rightarrow_2 \mathsf{d}\ {}_1\!\Leftarrow \mathsf{c}$ (based on $\mathsf{b} \to_{1.5} \mathsf{d}\ {}_1\!\leftarrow \mathsf{c}$), as in Figure 3(B). Note that $\mathsf{b} \Rightarrow_{1.5} \mathsf{d}\ {}_1\!\Leftarrow \mathsf{c}$ would not establish decreasingness for $>$. However, the construction also admits the peak $\mathsf{b}\ {}_{1.5}\!\Leftarrow \mathsf{a} \Rightarrow_2 \mathsf{c}$, for which there is no peak $\mathsf{b}\ {}_{1.5}\!\leftarrow \mathsf{a} \to_2 \mathsf{c}$ in the original ARS, as it does not contain the step $\mathsf{b}\ {}_{1.5}\!\leftarrow \mathsf{a}$. Still, this peak can be closed locally decreasing for $>$, cf. Figure 3(C), based on the steps in the diagram of Figure 3(A).

Proof of Lemma 3.3. We assume the ARS $\{\to_\alpha\}_{\alpha\in I}$ is extended locally decreasing for $>$ and $\geqslant$ and establish local decreasingness of the ARS $\{\Rightarrow_\alpha\}_{\alpha\in I}$ for $>$ by showing

$$\underset{\alpha}{\Leftarrow} \cdot \underset{\beta}{\Rightarrow} \subseteq \overset{*}{\underset{\curlyvee\alpha}{\Leftrightarrow}} \cdot \overset{=}{\underset{\beta}{\Rightarrow}} \cdot \overset{*}{\underset{\curlyvee\alpha\beta}{\Longleftrightarrow}} \cdot \overset{=}{\underset{\alpha}{\Leftarrow}} \cdot \overset{*}{\underset{\curlyvee\beta}{\Leftrightarrow}} \tag{3.1}$$

for $\alpha,\beta \in I$. Assume that $x\ {}_\alpha\!\Leftarrow \cdot \Rightarrow_\beta y$. By the definition of $\Rightarrow$, there are $\alpha' \leqslant \alpha$ and $\beta' \leqslant \beta$ such that $x\ {}_{\alpha'}\!\leftarrow \cdot \to_{\beta'} y$. Because $\to$ is extended locally decreasing, this implies that

$$x \overset{*}{\underset{\curlyvee\alpha'}{\longleftrightarrow}} \cdot \overset{=}{\underset{\curlyvee\!\!\!/\beta'}{\longrightarrow}} \cdot \overset{*}{\underset{\curlyvee\alpha'\beta'}{\longleftrightarrow}} \cdot \overset{=}{\underset{\curlyvee\!\!\!/\alpha'}{\longleftarrow}} \cdot \overset{*}{\underset{\curlyvee\beta'}{\longleftrightarrow}} y$$

Consider a label $\gamma \in \curlyvee\alpha'$, i.e., $\gamma < \alpha'$. By compatibility, this implies $\gamma < \alpha$, i.e., $\gamma \in \curlyvee\alpha$. Consequently, $\leftrightarrow_{\curlyvee\alpha'} \subseteq \Leftrightarrow_{\curlyvee\alpha}$, because $\leqslant$ is reflexive. Similarly, if $\gamma \in \curlyvee\!\!\!/\beta'$ then $\gamma \in \curlyvee\!\!\!/\beta$, because $\geqslant$ is transitive, and hence $\to^{=}_{\curlyvee\!\!\!/\beta'} \subseteq \Rightarrow^{=}_{\beta}$. Continuing in this fashion we obtain

$$x \overset{*}{\underset{\curlyvee\alpha}{\Leftrightarrow}} \cdot \overset{=}{\underset{\beta}{\Rightarrow}} \cdot \overset{*}{\underset{\curlyvee\alpha\beta}{\Longleftrightarrow}} \cdot \overset{=}{\underset{\alpha}{\Leftarrow}} \cdot \overset{*}{\underset{\curlyvee\beta}{\Leftrightarrow}} y$$

This establishes (3.1). Consequently, $\Rightarrow$ is locally decreasing. $\square$

4. Formalized Confluence Results for Term Rewriting

This section builds upon the result for ARSs from the previous section to prepare for confluence criteria for TRSs. First we recall the rule labeling [24], which maps each rewrite step to a natural number based on the rewrite rule applied in the step.

Definition 4.1. Let $i\colon \mathcal{R} \to \mathbb{N}$ be an index mapping, which associates to every rewrite rule a natural number. The function $\ell^i(s \to^\pi t) = i(l_\pi \to r_\pi)$ is called *rule labeling*. Labels due to the rule labeling are compared by $>_\mathbb{N}$ and $\geqslant_\mathbb{N}$.

Our aim is to formalize the following theorem, which is one of the main results of [41]:

Theorem 4.2 (Valley version of decreasing diagrams)**.** *A left-linear TRS is confluent if its duplicating rules terminate relative to its other rules and all its critical peaks are decreasing for the rule labeling.*

To support other confluence results, in the formalization we did not follow the easiest path, i.e., suit the definitions and lemmas directly to Theorem 4.2. Rather, we adopted the approach from [41], where all results are established via *labeling functions* (satisfying some abstract properties). Apart from avoiding a monolithic proof, this has the advantage that similar proofs need not be repeated for different labeling functions. Instead it suffices to establish that the concrete labeling functions satisfy some abstract conditions. In this approach decreasingness is established in three steps. The first step comprises joinability results for local peaks (Section 4.1). The second step (Section 4.2) formulates abstract conditions with the help of *labeling functions* that admit a finite characterization of decreasingness of local peaks. Finally, based on the previous two steps, the third step (Section 5) then obtains confluence results by instantiating the abstract labeling functions with concrete ones, e.g. the rule labeling. So only the third step needs to be adapted when formalizing new labeling functions, as steps one and two are unaffected. The content of this section is part of the `Decreasing_Diagrams2.thy` theory in IsaFoR.

Compared to [20], we formalized a conversion version of Theorem 4.2, which we will describe later (cf. Theorem 5.10). This generalization affects Sections 4.2 and 5.

4.1. Local Peaks.

As IsaFoR already supported Knuth-Bendix' criterion (see [31]), it contained results for joinability of local peaks and the critical pair theorem (the terms obtained by a local peak in a left-linear TRS are joinable or an instance of a critical pair). However, large parts of the existing formalization could not be reused directly as the established results lacked information required for ensuring decreasingness. For instance, to obtain decreasingness for the rule labeling in case of a variable peak, the rewrite rules employed in the joining sequences are crucial, but the existing formalization only states that such a local peak is joinable. On the other hand, the existing notion of critical pairs from IsaFoR could be reused as the foundation for critical peaks. Since the computation of critical pairs requires a formalized unification algorithm, extending IsaFoR admitted focusing on the tasks related to decreasingness.

Local peaks can be characterized based on the positions of the diverging rewrite steps. Either the positions are parallel, called a *parallel peak*, or one position is above the other. In the latter situation we further distinguish whether the lower position is at a function position, called a *function peak*, or at/below a variable position of the other rule's left-hand side, called a *variable peak*. More precisely, for a local peak

$$t = s[r_1\sigma_1]_p \leftarrow s[l_1\sigma_1]_p = s = s[l_2\sigma_2]_q \to s[r_2\sigma_2]_q = u \tag{4.1}$$

$$\begin{tikzcd} & s \arrow[dl, "r_1 \leftarrow l_1"'] \arrow[dr, "l_2 \rightarrow r_2"] & \\ t \arrow[dr, "l_2 \rightarrow r_2"'] & & u \arrow[dl, "r_1 \leftarrow l_1"] \\ & v & \end{tikzcd}$$

(A) Parallel peak

$$\begin{tikzcd} & & s \arrow[dll, "r_1 \leftarrow l_1"'] \arrow[drr, "l_2 \rightarrow r_2"] & & \\ t \arrow[dr] & & & & u \arrow[dl] \\ & \cdot & & \cdot & \\ & & ? & & \end{tikzcd}$$

(B) Function peak

$$\begin{tikzcd} & s \arrow[dl, "r_1 \leftarrow l_1"'] \arrow[dr, "l_2 \rightarrow r_2"] & \\ t \arrow[dr, "l_2 \rightarrow r_2"'] & & u \arrow[dl, "\parallel\, r_1 \leftarrow l_1"] \\ & v & \end{tikzcd}$$

(C) Variable peak

FIGURE 4. Three kinds of local peaks.

```
local_peaks ℛ =
  {((s, r₁, p₁, σ₁, True, t), (s, r₂, p₂, σ₂, True, u)) |s t u r₁ r₂ p₁ p₂ σ₁ σ₂.
   (s, t) ∈ rstep_r_p_s ℛ r₁ p₁ σ₁ ∧ (s, u) ∈ rstep_r_p_s ℛ r₂ p₂ σ₂}

parallel_peak ℛ p =
  (p ∈ local_peaks ℛ ∧
   (let ((s, r₁, p₁, σ₁, b, t), (s, r₂, p₂, σ₂, d, u)) = p in p₁ ⊥ p₂))

function_peak ℛ p =
  (p ∈ local_peaks ℛ ∧
   (let ((s, rl₁, p₁, σ₁, b, t), (s, rl₂, p₂, σ₂, d, u)) = p in
    ∃ r. p₁ <#> r = p₂ ∧ r ∈ poss (fst rl₁) ∧ is_Fun (fst rl₁ |_ r)))

variable_peak ℛ p =
  (p ∈ local_peaks ℛ ∧
   (let ((s, rl₁, p₁, σ₁, b, t), (s, rl₂, p₂, σ₂, d, u)) = p in
    ∃ r. p₁ <#> r = p₂ ∧ ¬ (r ∈ poss (fst rl₁) ∧ is_Fun (fst rl₁ |_ r))))
```

LISTING 2. Characterization of local peaks.

there are three possibilities (modulo symmetry):

(A): $p \parallel q$ (parallel peak),
(B): $q \leqslant p$ and $p \backslash q \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(l_2)$ (function peak),
(C): $q \leqslant p$ and $p \backslash q \notin \mathcal{P}\mathsf{os}_{\mathcal{F}}(l_2)$ (variable peak).

For the situation of a left-linear TRS these cases are visualized in Figure 4. It is easy to characterize parallel, function, and variable peaks in Isabelle (cf. Listing 2) but it requires tedious notation. E.g., the ***local_peaks*** of a TRS $\mathcal{R}$ are represented as the set of all pairs $(s, r_1, p_1, \sigma_1, \mathsf{True}, t)$ and $(s, r_2, p_2, \sigma_2, \mathsf{True}, u)$, with rewrite steps $s \rightarrow^{p_1, r_1, \sigma_1} t$ and $s \rightarrow^{p_2, r_2, \sigma_2} u$. The rewrite steps are formalized via the existing IsaFoR notion of ***rstep_r_p_s***, which represents a step $s \rightarrow_{\mathcal{R}}^{p, l \rightarrow r, \sigma} t$ as $(s,\ t) \in$ ***rstep_r_p_s*** $\mathcal{R}\ (l,\ r)\ p\ \sigma$. Here True and False are used to recall the direction of a step, i.e., $s \rightarrow t$ and $t \leftarrow s$, respectively. This distinction is important for steps that are part of conversions, because conversions may mix forward and backward steps. In the definitions of ***function_peak*** and ***variable_peak*** the symbol <#> is used, which is the IsaFoR operation for concatenating positions.

$p \in$ ***local_peaks*** $\mathcal{R} \Longrightarrow$
parallel_peak $\mathcal{R}\ p \lor$ ***variable_peak*** $\mathcal{R}\ p \lor$ ***variable_peak*** $\mathcal{R}$ (***snd*** p, ***fst*** p) $\lor$
function_peak $\mathcal{R}\ p \lor$ ***function_peak*** $\mathcal{R}$ (***snd*** p, ***fst*** p)

Listing 3. Cases of local peaks.

As the definition of function and variable peaks is asymmetric the five cases of local peaks can be reduced to the above three by mirroring those peaks. Then local peaks can be characterized as in Listing 3. Next we elaborate on the three cases.

Case 1: Parallel Peaks. Figure 4(A) shows the shape of a local peak where the steps take place at parallel positions. For a peak $t\ {}^{\pi_1}\!\!\leftarrow s \rightarrow^{\pi_2} u$ with $\pi_1 \parallel \pi_2$ we established that $t \rightarrow^{\pi_2} v\ {}^{\pi_1}\!\!\leftarrow u$, where opposing steps apply the same rule/substitution at the same position. The proof is straightforward and based on a decomposition of the terms into a context and the redex.

Case 2: Function Peaks. In general joining function peaks may involve rules not present in the divergence (as indicated by the question mark in Figure 4(B)). To reduce the burden of joining (infinitely many) function peaks to joining the (in case of a finite TRS finitely many) critical peaks, we formalized that every function peak is an instance of a critical peak.

Lemma 4.3. *Let $t\ {}^{p,l_1 \to r_1,\sigma_1}\!\!\leftarrow s \rightarrow^{q,l_2 \to r_2,\sigma_2} u$ with $qq' = p$, and $q' \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(l_2)$. Then there are a context C, a substitution τ, and a critical peak $l_2\mu[r_1\mu]_{q'} \leftarrow l_2\mu \rightarrow r_2\mu$ such that $s = C[l_2\mu\tau]$, $t = C[(l_2\mu[r_1\mu]_{q'})\tau]$, and $u = C[r_2\mu\tau]$.* □

This fact is folklore, see e.g. [35, Lemma 2.7.12]. We remark that this fact was already present (multiple times) in IsaFoR, but concealed in larger proofs, e.g. the formalization of orthogonality [19], and never stated explicitly.

As IsaFoR does not enforce that the variables of a rewrite rule's right-hand side are contained in its left-hand side, such rules are just also included in the critical peak computation.

Case 3: Variable Peaks. Variable overlaps (Figure 4(C)) can again be joined by the rules involved in the diverging step.[9] We only consider the case if $l_2 \to r_2$ is left-linear, as our main result assumes left-linearity. More precisely, if q' is the unique position in $\mathcal{P}\mathsf{os}_{\mathcal{V}}(l_2)$ such that $qq' \leqslant p$, $x = l_2|_{q'}$, and $|r_2|_x = n$ then we have $t \rightarrow_{l_2 \to r_2} v$, which is similar to the case for parallel peaks, as the redex $l_2\sigma$ becomes $l_2\tau$ but is not destroyed, and $u \rightarrow^n_{l_1 \to r_1} v$. To obtain this result we reason via parallel rewriting. The notion of parallel rewriting already supported by IsaFoR (employed to prove that orthogonal systems are confluent) does not keep track of, for example, the applied rules. Thus we augmented IsaFoR by a new version of parallel steps, which record the information (position, rewrite rule, substitution) of each rewrite step, i.e., the rewrite relation is decorated with the contracted redex patterns.

Definition 4.4. For a TRS $\mathcal{R}$ the *parallel rewrite relation* $\mathrel{+\!\!\!\!\twoheadrightarrow}$ is defined by the following inference rules.

$$\frac{}{x \overset{\varnothing}{\mathrel{+\!\!\!\twoheadrightarrow}} x} \qquad \frac{l \to r \in \mathcal{R}}{l\sigma \xrightarrow{\{\langle \epsilon, l \to r, \sigma\rangle\}} r\sigma} \qquad \frac{s_1 \overset{P_1}{\mathrel{+\!\!\!\twoheadrightarrow}} t_1 \quad \cdots \quad s_n \overset{P_n}{\mathrel{+\!\!\!\twoheadrightarrow}} t_n}{f(s_1,\ldots,s_n) \xrightarrow{(1P_1)\cup\cdots\cup(nP_n)} f(t_1,\ldots,t_n)}$$

[9]This includes rules having a variable as left-hand side.

$$\frac{}{(s,\ []) \in \textbf{\textit{conv}}\ \mathcal{R}}$$

$$\frac{(s,\ t) \in \textbf{\textit{rstep\_r\_p\_s}}\ \mathcal{R}\ r\ p\ \sigma \qquad (t,\ ts) \in \textbf{\textit{conv}}\ \mathcal{R}}{(s,\ (s,\ r,\ p,\ \sigma,\ \mathsf{True},\ t)\ \#\ ts) \in \textbf{\textit{conv}}\ \mathcal{R}}$$

$$\frac{(s,\ t) \in \textbf{\textit{rstep\_r\_p\_s}}\ \mathcal{R}\ r\ p\ \sigma \qquad (s,\ ts) \in \textbf{\textit{conv}}\ \mathcal{R}}{(t,\ (s,\ r,\ p,\ \sigma,\ \mathsf{False},\ t)\ \#\ ts) \in \textbf{\textit{conv}}\ \mathcal{R}}$$

Listing 4. Conversions.

Here for a set of redex patterns $P = \{\pi_1, \ldots, \pi_m\}$ by iP we denote $\{i\pi_1, \ldots, i\pi_m\}$ with $i\pi = \langle ip, l \to r, \sigma\rangle$ for $\pi = \langle p, l \to r, \sigma\rangle$.

To use this parallel rewrite relation for closing variable peaks we formalized the following auxiliary facts.[10]

Lemma 4.5. *The following properties of the parallel rewrite relation hold:*

(1) *For all terms s we have $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{\varnothing} s$.*
(2) *If $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{\varnothing} t$ then $s = t$.*
(3) *If $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{P} t$ and $q \in \mathcal{P}\mathsf{os}(u)$ then $u[s]_q \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{qP} u[t]_q$.*
(4) *We have $s \to^{\pi} t$ if and only if $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{\{\pi\}} t$.*
(5) *If $\sigma(x) \to^{\pi} \tau(x)$ and $\sigma(y) = \tau(y)$ for all $y \in \mathcal{V}$ with $y \neq x$ then $t\sigma \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{P} t\tau$ with $l_{\pi'} \to r_{\pi'} = l_{\pi} \to r_{\pi}$ for all $\pi' \in P$.*
(6) *If $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{\{\pi\} \cup P} t$ then there is a term u with $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{\{\pi\}} u \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{P} t$.*
(7) *If $s \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{\{\pi_1,\ldots,\pi_n\}} t$ then $s \to^{\pi_1} \cdots \to^{\pi_n} t$.* □

In principle the statements of Lemma 4.5 follow from the definitions using straightforward induction proofs, building upon each other in the order they are listed. However, the additional bookkeeping, required to correctly propagate the information attached to the rewrite relation, makes them considerably more involved than for the existing, agnostic notion of parallel rewriting.

Now for reasoning about variable peaks as in case (C) on page 10 we decompose $u = s[r_2\sigma]_q$ and $v = s[r_2\tau]_q$ where $\sigma(y) = \tau(y)$ for all $y \in \mathcal{V} \setminus \{x\}$ and $\sigma(x) \to_{p\backslash qq', l_1 \to r_1} \tau(x)$. From the latter by item (5) we obtain $r_2\sigma \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{P} r_2\tau$, where all redex patterns in P use $l_1 \to r_1$. Then by item (3) we get $s[r_2\sigma]_q \mathrel{+\!\!\!\!\!\!\twoheadrightarrow}^{qP} s[r_2\tau]_q$ and finally $s[r_2\sigma]_q \to^{n}_{l_1 \to r_1} s[r_2\tau]_q$ with $n = |qP| = |P|$ by item (7).

4.2. Local Decreasingness.

This section presents a confluence result (cf. Corollary 4.12) based on decreasingness of the critical peaks. Abstract conditions, via the key notion of a labeling, will ensure that parallel peaks and variable peaks are decreasing. Furthermore, these conditions imply that decreasingness of the critical peaks implies decreasingness of the function peaks. For establishing (extended) local decreasingness, a label must be attached to rewrite steps. To facilitate the computation of labels, in the formalization conversions provide

[10]In a typical mathematical proof, these facts would be considered as self-evident. However, Isabelle is not so easily convinced, so a proof is required. So rather than being an interesting result, Lemma 4.5 serves to illustrate the level of detail that formalizations often require.

information about the intermediate terms, applied rules, etc. In Listing 4 the definition of conversions as an inductive set is provided. The first case states that the (empty) conversion starting from a term s is a conversion. The second case states that if $s \to_{\mathcal{R}} t$ (using rule r at position p with substitution σ) and there is a conversion starting from t (where the list ts collects the details on the conversion), then there also is a conversion starting from s and the details for this conversion are collected in the list where the tuple $(s, r, p, \sigma, \mathsf{True}, t)$ is added to the list ts. While the first two cases of Listing 4 amount to an inductive definition of rewrite sequences, together with the third case (which considers a step $t \leftarrow s$ and a conversion starting from s), conversions are obtained. Here True and False are used to recall the direction of a step, as in Listing 2.

Furthermore, labels are computed by a *labeling (function)*, having (local) information about the rewrite step (such as source and target term, applied rewrite rule, position, and substitution) it is expected to label. For reasons of readability in this presentation we employ the mathematical notation (e.g., $\to^*$ for sequences and $\leftrightarrow^*$ for conversions, etc.) with all information implicit but remark that the formalization works on sequences and conversions with explicit information (as in Listing 4).

Definition 4.6. A *labeling* is a function ℓ from rewrite steps to a set of labels such that for all contexts C and substitutions σ the following properties are satisfied:

- If $\ell(s \to^{\pi_1} t) > \ell(u \to^{\pi_2} v)$ then $\ell(C[s\sigma] \to^{C[\pi_1\sigma]} C[t\sigma]) > \ell(C[u\sigma] \to^{C[\pi_2\sigma]} C[v\sigma])$
- If $\ell(s \to^{\pi_1} t) \geqslant \ell(u \to^{\pi_2} v)$ then $\ell(C[s\sigma] \to^{C[\pi_1\sigma]} C[t\sigma]) \geqslant \ell(C[u\sigma] \to^{C[\pi_2\sigma]} C[v\sigma])$
- $\ell(s \to^{\pi} t) = \ell(t \; {}^{\pi}\!\!\leftarrow s)$

Here $C[\pi\sigma]$ denotes $\langle qp, l \to r, \tau\sigma\rangle$ for $\pi = \langle p, l \to r, \tau\rangle$ and $C|_q = \Box$.

As the co-domain of a labeling is a set of labels, a labeling according to Definition 4.6 maps a single rewrite step to a single label, which is different from how (some) ARSs are labeled in Section 3.

Example 4.7. The rule labeling is a labeling in our sense.

In presence of a labeling, conversions can be labeled at any time. This avoids lifting many notions (such as rewrite steps, local peaks, rewrite sequences, etc.) and results from rewriting to labeled rewriting.

In the next definition a labeling is applied to conversions via the equations $\ell(t \leftrightarrow^0 t) = \varnothing$, $\ell(s \to^{\pi} t \leftrightarrow^* u) = \{\ell(s \to^{\pi} t)\} \cup \ell(t \leftrightarrow^* u)$, and $\ell(s \; {}^{\pi}\!\!\leftarrow t \leftrightarrow^* u) = \{\ell(s \; {}^{\pi}\!\!\leftarrow t)\} \cup \ell(t \leftrightarrow^* u)$.

Definition 4.8. A local peak $t \; {}^{\pi_1}\!\!\leftarrow s \to^{\pi_2} u$ is *extended locally decreasing* (for a labeling ℓ and orders $>$ and $\geqslant$) if there is a local diagram, i.e., $t \leftrightarrow^* t' \to^= t'' \leftrightarrow^* u'' \; {}^=\!\!\leftarrow u' \leftrightarrow^* u$ such that its labels are extended locally decreasing, i.e.,

- $\ell(t \leftrightarrow^* t') \subseteq \curlyvee\ell(t \; {}^{\pi_1}\!\!\leftarrow s)$,
- $\ell(t' \to^= t'') \subseteq \curlyvee\!\!\!/\,\ell(s \to^{\pi_2} u)$,
- $\ell(t'' \leftrightarrow^* u'') \subseteq \curlyvee\ell(t \; {}^{\pi_1}\!\!\leftarrow s \to^{\pi_2} u)$,
- $\ell(u'' \; {}^=\!\!\leftarrow u') \subseteq \curlyvee\!\!\!/\,\ell(t \; {}^{\pi_1}\!\!\leftarrow s)$, and
- $\ell(u' \leftrightarrow^* u) \subseteq \curlyvee\ell(s \to^{\pi_2} u)$.

Following [38], we separate (local) diagrams (where rewriting is involved) from decreasingness (where only the labels are involved). In contrast to the valley version of decreasing diagrams, as employed in [20], where a sequence $\to^*$ can be decomposed into $\to^*_{\curlyvee\alpha} \cdot \to^=_{\curlyvee\!\!\!/\beta} \cdot \to^*_{\curlyvee\alpha\beta}$ based on the sequence of labels, this is no longer possible for the

```
eldc R ℓ r p =
  (∃ cl₁ sl cl₂ cr₁ sr cr₂. ldc_trs R p cl₁ sl cl₂ cr₁ sr cr₂ ∧
     eld_conv ℓ r p cl₁ sl cl₂ cr₁ sr cr₂)

ldc_trs R p cl₁ sl cl₂ cr₁ sr cr₂ =
  (p ∈ local_peaks R ∧ cl₁ ∈ conv R ∧ sl ∈ seq R ∧ cl₂ ∈ conv R ∧
   cr₁ ∈ conv R ∧ sr ∈ seq R ∧ cr₂ ∈ conv R ∧
   get_target (fst p) = first cl₁ ∧ last cl₁ = first sl ∧
   last sl = first cl₂ ∧ get_target (snd p) = first cr₁ ∧
   last cr₁ = first sr ∧ last sr = first cr₂ ∧ last cl₂ = last cr₂)

eld_conv ℓ r p cl₁ sl cl₂ cr₁ sr cr₂ =
  (ELD_1 r (ℓ (fst p)) (ℓ (snd p)) (map ℓ (snd cl₁)) (map ℓ (snd sl))
    (map ℓ (snd cl₂)) ∧
   ELD_1 r (ℓ (snd p)) (ℓ (fst p)) (map ℓ (snd cr₁)) (map ℓ (snd sr))
    (map ℓ (snd cr₂)))

ELD_1 r β α σ₁ σ₂ σ₃ =
  (set σ₁ ⊆ ds (fst r) {β} ∧ length σ₂ ≤ 1 ∧ set σ₂ ⊆ ds (snd r) {α} ∧
   set σ₃ ⊆ ds (fst r) {α, β})
```

Listing 5. Extended local decreasingness.

conversion version (as there is no guarantee that the step $\to^{=}_{\curlyvee\beta}$ is oriented correctly). Hence we made the decomposition explicit for the conversion version.

The corresponding predicate in IsaFoR is given in Listing 5 where extended local decreasingness (***eldc***) of a local peak p is expressed via the existence of conversions cl_1, cl_2, cr_1, cr_2 and possibly empty steps sl and sr that close the divergence caused by the local peak p in the shape of a local diagram (***ldc_trs***).[11] Here ***get_target*** gets the target of a rewrite step and ***first*** and ***last*** get the first and last element of a conversion or rewrite sequence, respectively. Moreover the labels of the underlying conversions are required to be extended locally decreasing (***eld_conv***, ***ELD_1***). Here ***ds*** $(\preccurlyeq)$ S is the notation for $\curlyvee S$ and r is the pair of relations $(>, \geqslant)$.

Then a function peak is extended locally decreasing if the critical peaks are.

Lemma 4.9. *Let ℓ be a labeling and let all critical peaks of a TRS $\mathcal{R}$ be extended locally decreasing for ℓ. Then every function peak of $\mathcal{R}$ is extended locally decreasing for ℓ.*

Proof. As every function peak is an instance of a critical peak (see Lemma 4.3), the result follows from ℓ being a labeling (Definition 4.6). □

The notion of compatibility (between a TRS and a labeling) admits a finite characterization of extended local decreasingness.

Definition 4.10. Let ℓ be a labeling. We call ℓ *compatible* with a TRS $\mathcal{R}$ if all parallel peaks and all variable peaks of $\mathcal{R}$ are extended locally decreasing for ℓ.

The key lemma then establishes that if ℓ is compatible with a TRS, then all local peaks are extended locally decreasing.

[11] The conversions cl_2 and cr_2 could be merged into a single conversion. However, in proofs the symmetric version permits to reason about one side and obtain the other side for free.

Lemma 4.11. *Let ℓ be a labeling that is compatible with a TRS $\mathcal{R}$. If the critical peaks of $\mathcal{R}$ are extended locally decreasing for ℓ, then all local peaks of $\mathcal{R}$ are extended locally decreasing for ℓ.*

Proof. The cases of variable and parallel peaks are taken care of by compatibility. The case of function peaks follows from the assumption in connection with Lemma 4.9. The symmetric cases for function and variable peaks can be resolved by mirroring the local diagrams. □

Representing a TRS $\mathcal{R}$ over the signature $\mathcal{F}$ and variables $\mathcal{V}$ as the ARS over objects $\mathcal{T}(\mathcal{F},\mathcal{V})$ and relations $\to_\alpha = \{(s,t) \mid s \to^\pi t \text{ and } \ell(s \to^\pi t) = \alpha\}$ for some labeling ℓ, Lemma 3.1 immediately applies to TRSs. To this end, extended local decreasingness formulated via explicit rewrite sequences (with labeling functions) has to be mapped to extended local decreasingness on families of (abstract rewrite) relations; we omit the technical details here. Finally, we are ready to formalize a confluence result for first-order term rewriting.

Corollary 4.12. *Let ℓ be a labeling compatible with a TRS $\mathcal{R}$, and let $>$ and $\geqslant$ be a well-founded order and a compatible preorder, respectively. If the critical peaks of $\mathcal{R}$ are extended locally decreasing for ℓ and $>$ and $\geqslant$ then $\mathcal{R}$ is confluent.* □

Concrete confluence criteria are then obtained as instances of the above result by instantiating ℓ. For example, Theorem 4.2 is obtained by showing that its relative termination assumption in combination with the rule labeling yields a compatible labeling for left-linear TRSs.

5. Checkable Confluence Proofs

In this section we instantiate Corollary 4.12 to obtain concrete confluence results; first for linear TRSs (Section 5.1) and then for left-linear TRSs (Section 5.2). Afterwards we discuss the design of the certificates, checkable by CeTA, in Section 5.3. The first two subsections are part of `Decreasing_Diagrams2.thy` in IsaFoR, whereas the executable versions from Section 5.3 can be found in `Rule_Labeling_Impl.thy`.

5.1. Checkable Confluence Proofs for Linear TRSs.

The rule labeling admits a confluence criterion for linear TRSs based on the formalization established so far.

Lemma 5.1.
(1) *The rule labeling is a labeling.*
(2) *Parallel peaks are extended locally decreasing for the rule labeling.*
(3) *Variable peaks of a linear TRS are extended locally decreasing for the rule labeling.*
(4) *The rule labeling is compatible with a linear TRS.*
(5) *A linear TRS is confluent if its critical peaks are extended locally decreasing for the rule labeling.*

Proof. Item (1) follows from Definition 4.6 and Definition 4.1. For items (2) and (3) we employ the analysis of parallel and variable peaks from Section 4.1, respectively. Item (4) is then a consequence of items (2) and (3). Finally, item (5) amounts to an application of Corollary 4.12. □

$$\begin{tikzcd}
 & & \mathsf{a} \arrow[dll, "1"'] \arrow[drr, "1"] & & \\
\mathsf{b} & \mathsf{c} \arrow[l, "0"] \arrow[r, "0"'] & \mathsf{a} \arrow[dr, "1"'] & & \mathsf{d} \arrow[dl, equal] \\
 & & & \mathsf{d} &
\end{tikzcd}$$

(A)

$$\begin{tikzcd}
 & \mathsf{c} \arrow[dl, "0"'] \arrow[dr, "0"] & \\
\mathsf{b} \arrow[rr, "0"'] & & \mathsf{a}
\end{tikzcd}$$

(B)

FIGURE 5. Decreasingness of critical peaks in Example 5.2.

We demonstrate the rule labeling on a simple example.

Example 5.2. Consider the TRS consisting of the following rules, where subscripts indicate labels for the rule labeling:

$$\mathsf{a} \to_1 \mathsf{b} \qquad \mathsf{a} \to_1 \mathsf{d} \qquad \mathsf{b} \to_0 \mathsf{a} \qquad \mathsf{c} \to_0 \mathsf{a} \qquad \mathsf{c} \to_0 \mathsf{b}$$

The critical peaks are decreasing for the rule labeling, as depicted in Figure 5.

Eventually, we remark that for the rule labeling extended local decreasingness directly implies local decreasingness, as $\geqslant_{\mathbb{N}}$ is the reflexive closure of $>_{\mathbb{N}}$.

5.2. **Checkable Confluence Proofs for Left-linear TRSs.** That a locally confluent terminating left-linear TRS is confluent can be established in the flavor of Lemma 5.1. The restriction to left-linearity arises from the lack of considering non-left-linear variable peaks in Section 4.1. As the analysis of such a peak would not give further insights we pursue another aim in this section, i.e., the mechanized proof of Theorem 4.2.

It is well known that the rule labeling ℓ^i is in general not compatible with left-linear TRSs, cf. [13]. Thus, to obtain extended local decreasingness for variable peaks, in Theorem 4.2 the additional relative termination assumption is exploited. To this end, we use the source labeling.

Definition 5.3. The *source labeling* maps a rewrite step to its source, i.e., $\ell^{\mathsf{src}}(s \to^{\pi} t) = s$. Labels due to the source labeling are compared by the orders $\to^{+}_{\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}}$ and $\to^{*}_{\mathcal{R}}$.

The relative termination assumption of Theorem 4.2 makes all variable peaks of a left-linear TRS extended locally decreasing for the source labeling. These variable peaks might not be extended locally decreasing for the rule labeling, as the step $u \mathrel{+\!\!\!\!\twoheadrightarrow}_{\{l_1 \to r_1\}} v$ in Figure 4(C) yields $u \to^n v$ for n possibly larger than one. Hence we introduce weaker versions of decreasingness and compatibility.

Definition 5.4. A diagram of the shape $t \;{}_{\alpha}\!\leftarrow s \to_{\beta}^{l_2 \to r_2} u$, $t \to_{\vee\!\!/\beta} v \;{}_{\vee\!\!/\alpha}^{\;n}\!\leftarrow u$ is called *weakly extended locally decreasing* if $n \leqslant 1$ whenever r_2 is linear. We call a labeling ℓ *weakly compatible* with a TRS $\mathcal{R}$ if parallel and variable peaks are weakly extended locally decreasing for ℓ.

Following [41], the aim is to establish that the lexicographic combination of a compatible labeling with a weakly compatible labeling (e.g., $\ell^{\mathsf{src}} \times \ell^i$) is compatible with a left-linear

TRS. While weak extended local decreasingness could also be defined in the spirit of extended local decreasingness (with a more complicated join sequence or involving conversions), the chosen formulation suffices to establish the result, eases the definition, and simplifies proofs. As this notion is only used to reason about parallel and variable peaks, which need not be processed by automatic tools, the increased generality would not benefit automation.

Based on the peak analysis of Section 4.1, the following results are formalized (such properties must be proved for each labeling function):

Lemma 5.5. *Let $\mathcal{R}$ be a left-linear TRS.*

(1) *Parallel peaks are weakly extended locally decreasing for the rule labeling.*
(2) *Variable peaks of $\mathcal{R}$ are weakly extended locally decreasing for the rule labeling.*
(3) *The rule labeling is weakly compatible with $\mathcal{R}$.*

Proof. Results (1) and (2) are established by labeling the rewrite steps in the corresponding diagrams based on the characterizations of parallel and variable peaks of Figures 4(A) and 4(C), respectively. Item (3) follows from (1) and (2) together with Lemma 5.1(1). □

Similar results are formalized for the source labeling.

Lemma 5.6. *Let $\mathcal{R}$ be a left-linear TRS whose duplicating rules terminate relative to the other rules.*

(1) *The source labeling is a labeling.*
(2) *Parallel peaks are extended locally decreasing for the source labeling.*
(3) *Variable peaks of $\mathcal{R}$ are extended locally decreasing for the source labeling.*
(4) *The source labeling is compatible with $\mathcal{R}$.*

Proof. As rewriting is closed under contexts and substitutions, item (1) holds. Items (2) and (3) are established along the lines of the proof of Lemma 5.5. Finally, item (4) follows from (2) and (3) in combination with Definition 4.10. □

Using this lemma, we proved the following results for the lexicographic combination of the source labeling with another labeling.

Lemma 5.7. *Let $\mathcal{R}$ be a left-linear TRS whose duplicating rules terminate relative to the other rules and ℓ be a labeling weakly compatible with $\mathcal{R}$.*

(1) *Then $\ell^{\mathsf{src}} \times \ell$ is a labeling.*
(2) *Then $\ell^{\mathsf{src}} \times \ell$ is compatible with $\mathcal{R}$.* □

For reasons of readability we have left the orders $>$ and $\geqslant$ that are required for (weak) compatibility implicit and just mention that the lexicographic extension (as detailed in [41]) preserves the required properties. Finally, we prove Theorem 4.2.

Proof of Theorem 4.2. From Lemma 5.5(3) in combination with Lemma 5.7 we obtain that $\ell^{\mathsf{src}} \times \ell^i$ is a labeling compatible with a left-linear TRS, provided the relative termination assumption is satisfied. By assumption, the critical peaks are (extended locally) decreasing for the rule labeling ℓ^i. As along a rewrite sequence labels with respect to ℓ^{src} never increase, the critical peaks are extended locally decreasing for $\ell^{\mathsf{src}} \times \ell^i$. We conclude the proof by an application of Corollary 4.12. □

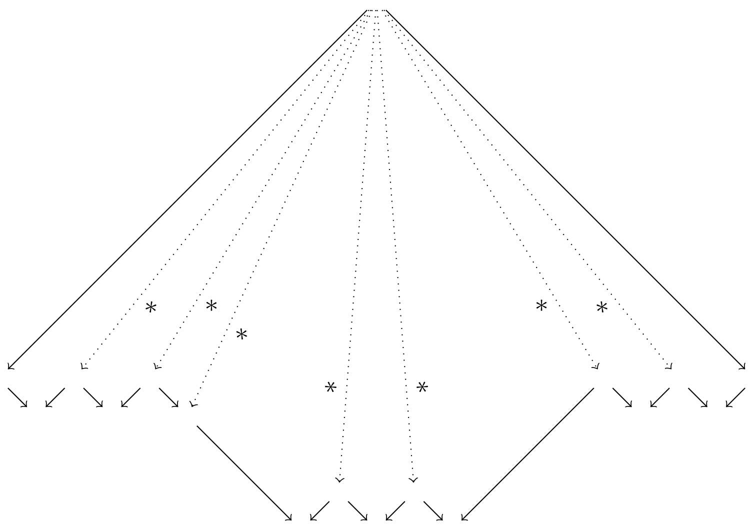


FIGURE 6. Additional property for the conversion version of Theorem 4.2.

Actually a stronger result than Theorem 4.2 has been established, as $\ell^{\mathsf{src}} \times \ell^i$ might show more critical peaks decreasing than ℓ^i alone.

Regarding a variant of Theorem 4.2 based on the conversion version of decreasing diagrams the statement "*As along a rewrite sequence labels with respect to ℓ^{src} never increase, . . .*" in the above proof is no longer appropriate, as rewrite sequences are replaced by conversions. Consequently, in contrast to the valley version, the source labeling might destroy decreasingness, as shown by the next example.

Example 5.8 (Example 5.2 continued)**.** Although the critical diagram in Figure 5(A) is decreasing for ℓ^i (cf. Example 5.2) it is not decreasing for $\ell^{\mathsf{src}} \times \ell^i$ as the label of the step $\mathsf{a} \to_{(\mathsf{a},1)} \mathsf{b}$ is not larger than the label of the step $\mathsf{b} \;_{(\mathsf{c},0)}\!\leftarrow \mathsf{c}$. The problem is that $\mathsf{a} \to^* \mathsf{c}$ does not hold.

To forbid the situation highlighted in Example 5.8, the property that the labels of the conversions are smaller or equal to the source of the local peak must be ensured.

Definition 5.9. A diagram of the shape $t_1 \leftarrow s \to t_n$, $t_1 \leftrightarrow t_2 \leftrightarrow \cdots \leftrightarrow t_n$ has the *fan property* if $s \to^* t_i$ for $1 \leqslant i \leqslant n$.

The fan property is sketched in Figure 6, where the solid arcs indicate the diagram and the dotted arcs the additional conditions. Our formalization covers the following result, which is new in theory and IsaFoR:

Theorem 5.10 (Conversion version of decreasing diagrams)**.** *A left-linear TRS is confluent if its duplicating rules terminate relative to its other rules and all its critical peaks have local diagrams that both have the fan property and are decreasing for the rule labeling.*

Proof. That $\ell^{\mathsf{src}} \times \ell^i$ is a labeling, is obtained as in the proof of Theorem 4.2. The fan property ensures that a critical peak that is decreasing for the rule labeling ℓ^i is also decreasing for $\ell^{\mathsf{src}} \times \ell^i$. We conclude by an application of Corollary 4.12. □

The following example demonstrates that the fan property is necessary for Theorem 5.10 to be correct. Note that the TRS in Example 5.8 is confluent and can hence only motivate the fan property but cannot show its indispensability.

Example 5.11. Consider the TRS $\mathcal{R}$ consisting of the following rules, where subscripts indicate labels for the rule labeling:

$$\mathsf{a} \to_2 \mathsf{b} \qquad \mathsf{f}(\mathsf{a},\mathsf{b}) \to_1 \mathsf{f}(\mathsf{a},\mathsf{a}) \qquad \mathsf{f}(\mathsf{b},\mathsf{a}) \to_1 \mathsf{f}(\mathsf{a},\mathsf{a}) \qquad \mathsf{f}(\mathsf{a},\mathsf{a}) \to_1 \mathsf{c} \qquad \mathsf{g}(x) \to_0 \mathsf{f}(x,x)$$

Then $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$ is easily seen to be terminating, for example using the lexicographic path order with a quasi-precedence that equates a and b and has $\mathsf{g} > \mathsf{f} > \mathsf{c}$. There are four critical peaks, all of which are decreasing with respect to the given rule labeling:

$$\begin{array}{ll}
\mathsf{f}(\mathsf{a},\mathsf{b}) \;{}_2\!\leftarrow \mathsf{f}(\mathsf{a},\mathsf{a}) \to_1 \mathsf{c} & \qquad \mathsf{f}(\mathsf{a},\mathsf{b}) \to_1 \mathsf{f}(\mathsf{a},\mathsf{a}) \to_1 \mathsf{c} \\
\mathsf{f}(\mathsf{b},\mathsf{a}) \;{}_2\!\leftarrow \mathsf{f}(\mathsf{a},\mathsf{a}) \to_1 \mathsf{c} & \qquad \mathsf{f}(\mathsf{b},\mathsf{a}) \to_1 \mathsf{f}(\mathsf{a},\mathsf{a}) \to_1 \mathsf{c} \\
\mathsf{f}(\mathsf{b},\mathsf{b}) \;{}_2\!\leftarrow \mathsf{f}(\mathsf{a},\mathsf{b}) \to_1 \mathsf{f}(\mathsf{a},\mathsf{a}) & \qquad \mathsf{f}(\mathsf{b},\mathsf{b}) \;{}_0\!\leftarrow \mathsf{g}(\mathsf{b}) \;{}_2\!\leftarrow \mathsf{g}(\mathsf{a}) \to_0 \mathsf{f}(\mathsf{a},\mathsf{a}) \\
\mathsf{f}(\mathsf{b},\mathsf{b}) \;{}_2\!\leftarrow \mathsf{f}(\mathsf{b},\mathsf{a}) \to_1 \mathsf{f}(\mathsf{a},\mathsf{a}) & \qquad \mathsf{f}(\mathsf{b},\mathsf{b}) \;{}_0\!\leftarrow \mathsf{g}(\mathsf{b}) \;{}_2\!\leftarrow \mathsf{g}(\mathsf{a}) \to_0 \mathsf{f}(\mathsf{a},\mathsf{a})
\end{array}$$

Nevertheless, $\mathcal{R}$ is not confluent: $\mathsf{f}(\mathsf{b},\mathsf{b}) \leftarrow \mathsf{f}(\mathsf{a},\mathsf{b}) \leftarrow \mathsf{f}(\mathsf{a},\mathsf{a}) \to \mathsf{c}$ is a conversion between two distinct normal forms. Note that the local diagrams for the final two critical peaks violate the fan property: $\mathsf{g}(\mathsf{a})$ is not reachable from $\mathsf{f}(\mathsf{a},\mathsf{b})$ nor $\mathsf{f}(\mathsf{b},\mathsf{a})$.

Finally, we remark that in the formalization Theorem 4.2 is obtained by instantiating Theorem 5.10. To this end, we formalized that the fan property holds vacuously whenever the local diagram is a valley. The direct proof of Theorem 4.2 on page 17 does not have a correspondence in the formalization but it already conveys the proof idea of the more complex proof of Theorem 5.10.

5.3. Certificates.

Next we discuss the design of the certificates for confluence proofs via Theorem 5.10, i.e., how they are represented in CPF, and the executable checker to verify them. The main structure of a certificate in CPF consists of an input, in our case the TRS whose confluence should be certified, and a proof. For the proof, a minimal certificate could just claim that the considered rewrite system can be shown decreasing via the rule labeling. However, this is undecidable, even for locally confluent systems [13]. Hence the certificate contains the following entries: the TRS $\mathcal{R}$, the index function i, (candidates for) the joining conversions for each critical peak, an upper bound on the number of rewrite steps required to check the fan property, and, in case $\mathcal{R}$ is not right-linear, a relative termination proof for $\mathcal{R}_{\mathsf{d}}/\mathcal{R}_{\mathsf{nd}}$. The labels in the joining conversions are not required in the certificate, since CeTA has to check, i.e., compute them anyway. The same holds for the critical peaks.[12] Note that the (complex) reasoning required for parallel and variable peaks does not pollute the certificates. The outline of a certificate for a confluence proof according to Theorem 5.10 is shown in Figure 7.[13] Besides elements for structuring the certificate (`decreasingDiagrams`, `ruleLabelingConv`), additions to CPF, for representing proofs via Theorem 5.10, are the elements to represent the rule labeling (`ruleLabelingFunction`) and the joining conversions `convertibleCriticalPeaks`. Conversions and rewrite sequences themselves were already representable in CPF (`conversion`, `rewriteSequence`) and easy to reuse. To check such a

[12]Note that IsaFoR already comes with a formalized unification algorithm that was easy to reuse.

[13]In Figure 7 some boilerplate nodes and details are omitted—a full certificate in CPF for Example 5.2 is available at `http://cl-informatik.uibk.ac.at/experiments/2016/rule_labeling/rule_labeling_conv.proof.xml`.

```
certificationProblem
├── input
│   └── trs
│       ├── rule
│       │   ├── lhs
│       │   └── rhs
│       └── ...
└── proof
    └── decreasingDiagrams
        ├── relativeTerminationProof
        └── ruleLabelingConv
            ├── ruleLabelingFunction
            │   ├── ruleLabelingFunctionEntry
            │   │   ├── rule
            │   │   └── label
            │   └── ...
            └── convertibleCriticalPeaks
                ├── convertibleCriticalPeak
                │   ├── source
                │   ├── conversionLeft
                │   │   ├── conversion
                │   │   ├── rewriteSequence
                │   │   └── conversion
                │   └── conversionRight
                │       ├── conversion
                │       ├── rewriteSequence
                │       └── conversion
                └── ...
```

FIGURE 7. Structure of a Rule Labeling Certificate in CPF.

certificate CeTA performs the following steps:

(1) Parse the certificate. To parse the CPF elements that were newly introduced we extended CeTA's parser accordingly.
(2) Check the proof for relative termination. Luckily, CeTA already supports a wide range of relative termination techniques, so that here we just needed to make use of existing machinery.
(3) Compute all critical peaks of the rewrite system specified in the certificate.
(4) For each computed critical peak, find and check a decreasing joining conversion given in the certificate. To check decreasingness (cf. Listing 5) we require the decomposition of the joining conversions to be explicit in the certificate. As the confluence tools that generate certificates might use different renamings than CeTA when computing critical pairs, the conversions given in the certificate are subject to a variable renaming. Thus, after computing the critical peaks, CeTA has to consider the joining conversions modulo

method	success	CoCo '13	CoCo '14	CoCo '15	CoCo '16
Knuth-Bendix	27(+16)	✓	✓	✓	✓
(weak) orthogonality	4(+37)	✓	✓	✓	✓
strong closedness	28(+18)	✓	✓	✓	✓
Lemma 5.1(5)	44(+ 6)	✗	✓	✓	✓
parallel closedness	16(+36)	✗	✗	✗	✓
Theorem 4.2	49(+ 7)	✗	✗	✓	✓
redundant rules	—	✗	✗	✓	✓
$\sum$		46	59	76	77

Table 1. Experimental results for 164 TRSs from Cops.

renaming of variables. Then checking that they form a local diagram and that the labels are extended locally decreasing is straightforward.

(5) Check the fan property. The certificate contains an upper bound on the number of rewrite steps required to reach the terms in the conversions from the source of the peak. This ensures termination of CeTA when checking the existence of suitable rewrite sequences.

CeTA also supports checking decreasingness using the valley version of decreasing diagrams, i.e., certifying applications of Theorem 4.2. In that case splitting the joining sequences in the certificate is not required: for every critical peak just two rewrite sequences need to be provided. CeTA can automatically find a split if one exists: given two natural numbers α and β and a sequence σ of natural numbers, is there a split $\sigma = \sigma_1\sigma_2\sigma_3$ such that $\sigma_1 \subseteq \curlyvee\alpha$, $\sigma_2 \subseteq \curlyvee\!\!\!/\beta$ with length of σ_2 at most one, and $\sigma_3 \subseteq \curlyvee\alpha\beta$? The checker employs a simple, greedy approach. That is, we pick the maximal prefix of σ with labels smaller α as σ_1. If the next label is less or equal to β we take it as σ_2 and otherwise we take the empty sequence for σ_2. Finally, the remainder of the sequence is σ_3. A straightforward case analysis shows that this approach is complete, i.e., otherwise no such split exists.

6. Experiments

In this section we compare the techniques from this paper with the other confluence criteria supported by CeTA via an experimental evaluation. For experiments we considered the 164 TRSs in Cops[14] that stem from the literature—this is also the pool from which the benchmarks for CoCo are drafted—and used the confluence tool CSI [40] to obtain certificates in CPF for confluence proofs. ACP [3] can also produce confluence certificates in CPF, but at the moment they are a subset of the ones reported by CSI. All generated certificates have been certified by CeTA. The largest certificate (for Cops #60) has 760 KB and lists 182 candidate joins for showing the 34 critical peaks decreasing. The certificate is checked within 1.1 seconds. We remark that no confluence tool besides CSI has solved Cops #60 so far, stressing the importance of a certified proof.

Next we elaborate on the impact of the new contributions. Experimental results for various confluence criteria supported by CeTA are shown in Table 1.[15] We track the progress

[14]The Confluence Problems database is available at `http://cops.uibk.ac.at`.

[15]Details are available from `http://cl-informatik.uibk.ac.at/experiments/2016/rule_labeling`.

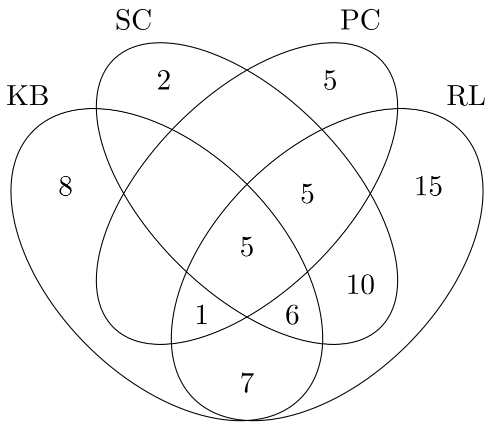


FIGURE 8. Overlap between solved Cops.

of certification for confluence along CoCo. The CeTA version from CoCo 2013 incorporated (weak) orthogonality [26], Knuth-Bendix' criterion [16], and Huet's result on strongly closed critical pairs [14]. As already employed for CoCo 2014, we included the data for Theorem 4.2 restricted to linear TRSs, i.e., Lemma 5.1(5). Due to the formalization described in this article, since CoCo 2015, also Theorem 4.2 is supported. Since 2015 CeTA also supports a pre-processing technique for showing confluence, namely addition and removal of redundant rules [17]. The idea of that technique is to add or remove rules that can be simulated by the other rules of a TRS, which reflects confluence and often makes other criteria applicable. The gain for the direct methods when combined with this technique is shown in parentheses in the second column of Table 1. Recently also Huet's result on parallel closed critical pairs [14], including Toyama's extension [37], was formalized and added to CeTA by the first author [18]. Note that CeTA can also certify various methods for non-confluence [19], resulting in 36 certified non-confluence proofs.

On our test-bed Theorem 4.2 can establish confluence of as many systems as all other direct methods (weak orthogonality, Knuth-Bendix' criterion, strong and parallel closedness) combined (49 systems), admits about 25% increase in power (64 vs. 49) when used in combination with them and by itself shows confluence of about two thirds of all systems that are certifiably confluent by CSI+CeTA (77 vs. 49). Figure 8 shows the systems solved by Knuth-Bendix' criterion (KB), strong closedness (SC), parallel closedness (PC), and Theorem 4.2 (RL) in relation to each other. (We omit weak orthogonality and Lemma 5.1(5), since they are subsumed by parallel closedness and Theorem 4.2 respectively.) Overall the decreasing diagrams based rule labeling technique turned out much stronger than the other, simple, syntactic criteria. However, this increase in power comes at a cost: the certificates are usually larger and much harder to check for a human, making automatic, reliable certification a necessity.

We are not aware of a confluence tool that can yet produce certificates according to Theorem 5.10. However, in the case of linear TRSs, saigawa supports the rule labeling for the conversion version of decreasing diagrams [13], which is covered by Theorem 5.10 but not by Theorem 4.2. Producing checkable certificates for these proofs would be an easy exercise for saigawa's authors.

7. Conclusion

Finally we discuss related work, comment on the existing formalization of rewriting in IsaFoR, and conclude with a short summary.

7.1. **Related Work.**

Formalizing confluence criteria has a long history in the λ-calculus. Huet [15] proved a stronger variant of the parallel moves lemma in Coq. Isabelle/HOL was used in [21] to prove the Church-Rosser property of β, η, and $\beta\eta$. For β-reduction the standard Tait/Martin-Löf proof as well as Takahashi's proof [34] were formalized. The first mechanically verified proof of the Church-Rosser property of β-reduction was done using the Boyer-Moore theorem prover [28]. The formalization in Twelf [25] was used to formalize the confluence proof of a specific higher-order rewrite system in [33].

Next we discuss related work for term rewriting. Newman's lemma (for abstract rewrite systems) and Knuth and Bendix' critical pair theorem (for first-order rewrite systems) have been proved in [27] using ACL2. An alternative proof of the latter in PVS, following the higher-order structure of Huet's proof, is presented in [9]. PVS is also used in the formalization of the lemmas of Newman and Yokouchi in [8]. Knuth and Bendix' criterion has also been formalized in Coq [5] and Isabelle/HOL [31]. The strong and parallel closedness conditions of Huet [14] have been formalized by the first author in Isabelle/HOL [18], where reasoning similar to the one in Section 4.1 is used to close variable and parallel peaks. However, for Huet's criteria it suffices to construct a common reduct while for our setting every rewrite step has to be made explicit in order to compute the labels and show local decreasingness.

7.2. **Assessment.**

First we explain why Theorem 4.2 is an adequate candidate for a formalization. On the one hand, regarding the aspect of automation, it is easily implementable as the relative termination requirement can be outsourced to external (relative) termination provers and the rule labeling heuristic has already been implemented successfully [1, 12]. Furthermore, it is a powerful criterion as demonstrated by the experimental evaluation in Section 6. On the other hand, regarding the aspect of formalization, it is challenging because it involves the combination of different labeling functions (in the sense of [41]). Hence, in our formalization Theorem 4.2 is not established directly, but obtained as a corollary of more general results. In particular Lemma 5.7 is based on a more general result, which allows different labeling functions to be combined lexicographically. This paves the way for reusing the formalization described here when tackling the remaining criteria in [41], which are based on more flexible combinations of labeling functions, and use labelings besides the source labeling (Lemma 5.6) and the rule labeling (Lemma 5.1). For example, labels can also be defined based on the path to a rewrite step, or the redex that is being contracted. In order to certify the corresponding proofs, we will also have to extend the CPF format with encodings of those labelings.

The required characterization of (closing) local peaks (cf. Figure 4) provides full information about the rewrite steps involved in the joining sequences. As this characterization is the basis for many confluence criteria—not necessarily relying on decreasing diagrams—this result aids future certification efforts. We anticipate that the key result for closing variable peaks for the left-linear case (cf. Section 4.1) does not rely on the annotated version of parallel rewriting, but as [41] also supports labelings based on parallel rewriting, the developed machinery should be useful for targeting further confluence results from [41].

Needless to say, parallel rewriting is handy on its own. The formalization described in this article covers a significant amount of the results presented in [41]. As explained, additional concepts (e.g., the annotated version of parallel rewriting) were formalized with preparation of the remaining criteria in mind. However, for some results that are not covered yet (e.g., persistency), we anticipate that even formalizing the preliminaries requires significant effort.

Next we discuss the usefulness of existing formalizations for this work. The existing machinery of IsaFoR provided invaluable support. We regard our efforts to establish an annotated version of parallel rewriting not as a shortcoming of IsaFoR, but as a useful extension to it. On the contrary, we could employ many results from IsaFoR without further ado, e.g., completeness of the unification algorithm (to compute critical peaks), plain rewriting (to connect parallel steps with single steps), and the support for relative termination. That Lemma 4.3 occurred several times in IsaFoR can be traced to textbook proofs, e.g. [4], where this result is not made explicit either. Instead it is established in the scope of a larger proof of the critical pair lemma. Still, in later proofs the result is used as if it would have been established explicitly. In IsaFoR these proofs have been duplicated, but as formalization papers typically come with code refactoring these deficiencies have been fixed. Note that the duplicated proofs have actually never been published. Ultimately our aim in the formalization was to follow paper proofs as closely as possible. The benefit of this choice is that this way, shortcomings in existing proofs can be identified and eradicated. As our formalization covers and combines results from various sources, the notions used in the papers had to be connected. As already mentioned, while different notations are typically identified in paper proofs, in the formalization this step has to be made explicit. To avoid this drawback in the future our recommendation is to strive for more standard notation, also in paper proofs.

Finally, differences to [41] are addressed. The concepts of an L-labeling and an LL-labeling from [41] have been generalized to the notion of a labeling *compatible* with a TRS while weak-LL-labelings are represented via *weakly compatible* labelings here.[16] This admits the formulation of the abstract conditions such that a labeling ensures confluence (cf. Corollary 4.12) independent from the TRS being (left-)linear. Furthermore we present a generalization of Theorem 4.2 to the conversion version of decreasing diagrams, namely Theorem 5.10.

7.3. Summary and Conclusion.

In this article we presented the formalization of a result establishing confluence of left-linear term rewrite systems based on relative termination and the rule labeling. While our formalization admits stronger results (in order to prepare for further results from [41]), we targeted Theorem 4.2, whose statement (in contrast to its proof) does not require the complex interplay of relative termination and the rule labeling. Hence this criterion is easily implementable for automated confluence tools, admitting the use of external termination provers. Our formalization subsumes the (original) criterion for the rule labeling (cf. Lemma 5.1(5)), which is applicable to linear systems only. Dealing with non-right-linear systems required an analysis of non-right-linear variable peaks, and of the interplay with the relative termination condition. Furthermore, whereas plain rule labeling can be proved correct by decreasing diagrams, the involvement of the source labeling

[16]The definitions of L-labelings and LL-labelings spell out the shape of the standard joining valley for a variable peak for linear and left-linear TRSs, respectively, and then impose restrictions on the occurring labels that ensure compatibility.

means that extended decreasing diagrams are required. Hence the proof of Theorem 4.2 is significantly more involved than the one of Lemma 5.1(5).

Despite the fact that any confluence proof by the conversion version of decreasing diagrams can be completed into a confluence proof by the valley version using the same labels (cf. the proof of [24, Theorem 3]), conversions can be (significantly) shorter than valleys [24, Example 8]. Regarding the conversion version of decreasing diagrams, in automated tools the main obstacle is finding suitable conversions. Albeit simple heuristics (cf. [13, Section 4]) have been proposed to limit the explosion in the search space when considering conversions, most automated confluence provers still favor the valley version. While those heuristics suffice for the rule labeling, Theorem 5.10 shows that in a more complex setting, conversions must satisfy additional restrictions, which make the search for suitable conversions even more challenging. Maybe the recent approach [39] to construct conversions can solve some of these challenges.

Acknowledgments

We thank Christian Sternagel and René Thiemann for insightful discussion and the reviewers for helpful comments.

References

[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. doi:10.4230/LIPIcs.RTA.2010.7.

[2] T. Aoto, N. Hirokawa, J. Nagele, N. Nishida, and H. Zankl. Confluence Competition 2015. In *Proc. 25th International Conference on Automated Deduction*, volume 9195 of *Lecture Notes in Artificial Intelligence*, pages 101–104, 2015. doi:10.1007/978-3-319-21401-6_5.

[3] 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. doi:10.1007/978-3-642-02348-4_7.

[4] F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998. doi:10.2277/0521779200.

[5] E. Contejean, P. Courtieu, J. Forest, O. Pons, and X. Urbain. Automated certified proofs with CiME3. In *Proc. 22nd International Conference on Rewriting Techniques and Applications*, volume 10 of *Leibniz International Proceedings in Informatics*, pages 21–30, 2011. doi:10.4230/LIPIcs.RTA.2011.21.

[6] B. Felgenhauer. Decreasing diagrams II. *Archive of Formal Proofs*, August 2015. Formal proof development, `http://www.isa-afp.org/entries/Decreasing-Diagrams-II.shtml`.

[7] B. Felgenhauer and V. van Oostrom. Proof orders for decreasing diagrams. In *Proc. 24th International Conference on Rewriting Techniques and Applications*, volume 21 of *Leibniz International Proceedings in Informatics*, pages 174–189, 2013. doi:10.4230/LIPIcs.RTA.2013.174.

[8] A.L. Galdino and M. Ayala-Rincón. A formalization of Newman's and Yokouchi's lemmas in a higher-order language. *Journal of Formalized Reasoning*, 1(1):39–50, 2008. doi:10.6092/issn.1972-5787/1347.

[9] A.L. Galdino and M. Ayala-Rincón. A formalization of the Knuth-Bendix(-Huet) critical pair theorem. *Journal of Automated Reasoning*, 45(3):301–325, 2010. doi:10.1007/s10817-010-9165-2.

[10] F. Haftmann. *Code Generation from Specifications in Higher Order Logic.* Dissertation, Technische Universität München, München, 2009. urn:nbn:de:bvb:91-diss-20091208-886023-1-1.

[11] F. Haftmann and T. Nipkow. Code generation via higher-order rewrite systems. In *Proc. 10th International Symposium on Functional and Logic Programming*, volume 6009 of *Lecture Notes in Computer Science*, pages 103–117. Springer, 2010. doi:10.1007/978-3-642-12251-4_9.

[12] 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. doi:10.1007/978-3-642-14203-1_41.

[13] N. Hirokawa and A. Middeldorp. Decreasing diagrams and relative termination. *Journal of Automated Reasoning*, 47(4):481–501, 2011. doi:10.1007/s10817-011-9238-x.

[14] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980. doi:10.1145/322217.322230.

[15] G. Huet. Residual theory in lambda-calculus: A formal development. *Journal of Functional Programming*, 4(3):371–394, 1994. doi:10.1017/S0956796800001106.

[16] D.E. Knuth and P.B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970. doi:10.1016/B978-0-08-012975-4.50028-X.

[17] J. Nagele, B. Felgenhauer, and A. Middeldorp. Improving automatic confluence analysis of rewrite systems by redundant rules. In *Proc. 26th International Conference on Rewriting Techniques and Applications*, volume 36 of *Leibniz International Proceedings in Informatics*, pages 257–268, 2015. doi:10.4230/LIPIcs.RTA.2015.257.

[18] J. Nagele and A. Middeldorp. Certification of classical confluence results for left-linear term rewrite systems. In *Proc. 7th International Conference on Interactive Theorem Proving*, volume 9807 of *Lecture Notes in Computer Science*, pages 290–306, 2016. doi:10.1007/978-3-319-43144-4_18.

[19] J. Nagele and R. Thiemann. Certification of confluence proofs using CeTA. In *Proc. 3rd International Workshop on Confluence*, pages 19–23, 2014. arXiv:1505.01337.

[20] J. Nagele and H. Zankl. Certified rule labeling. In *Proc. 26th International Conference on Rewriting Techniques and Applications*, volume 36 of *Leibniz International Proceedings in Informatics*, pages 269–284, 2015. doi:10.4230/LIPIcs.RTA.2015.269.

[21] T. Nipkow. More Church-Rosser proofs. *Journal of Automated Reasoning*, 26(1):51–66, 2001. doi:10.1023/A:1006496715975.

[22] T. Nipkow, L.C. Paulson, and M. Wenzel. *Isabelle/HOL – A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer, 2002. doi:10.1007/3-540-45949-9.

[23] V. van Oostrom. Confluence by decreasing diagrams. *Theoretical Computer Science*, 126(2):259–280, 1994. doi:10.1016/0304-3975(92)00023-K.

[24] 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. doi:10.1007/978-3-540-70590-1_21.

[25] F. Pfenning. A proof of the Church-Rosser theorem and its representation in a logical framework. Technical Report CMU-CS-92-186, School of Computer Science, Carnegie Mellon University, 1992.

[26] B.K. Rosen. Tree-manipulating systems and Church-Rosser theorems. *Journal of the ACM*, 20(1):160–187, 1973. doi:10.1145/321738.321750.

[27] J.-L. Ruiz-Reina, J.-A. Alonso, M.-J. Hidalgo, and F.-J. Martín-Mateos. Formal proofs about rewriting using ACL2. *Annals of Mathematics and Artificial Intelligence*, 36(3):239–262, 2002. doi:10.1023/A:1016003314081.

[28] N. Shankar. A mechanical proof of the Church-Rosser theorem. *Journal of the ACM*, 35(3):475–522, 1988. doi:10.1145/44483.44484.

[29] C. Sternagel. Well-quasi-orders. *Archive of Formal Proofs*, April 2012. Formal proof development, `http://isa-afp.org/entries/Well_Quasi_Orders.shtml`.

[30] C. Sternagel. Certified Kruskal's tree theorem. *Journal of Formalized Reasoning*, 7(1):45–62, 2014. doi:10.6092/issn.1972-5787/4213.

[31] C. Sternagel and R. Thiemann. Formalizing Knuth-Bendix orders and Knuth-Bendix completion. In *Proc. 24th International Conference on Rewriting Techniques and Applications*, volume 21 of *Leibniz International Proceedings in Informatics*, pages 287–302, 2013. doi:10.4230/LIPIcs.RTA.2013.287.

[32] C. Sternagel and R. Thiemann. The certification problem format. In *Proc. 11th International Workshop on User Interfaces for Theorem Provers*, volume 167 of *Electronic Proceedings in Theoretical Computer Science*, pages 61–72, 2014. doi:10.4204/EPTCS.167.8.

[33] K. Støvring. Extending the extensional lambda calculus with surjective pairing is conservative. *Logical Methods in Computer Science*, 2(2:1):1–14, 2006. doi:10.2168/LMCS-2(2:1)2006.

[34] M. Takahashi. Parallel reductions in λ-calculus. *Information and Computation*, 118(1):120–127, 1995. doi:10.1006/inco.1995.1057.

[35] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[36] 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. doi:10.1007/978-3-642-03359-9_31.

[37] 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.

[38] H. Zankl. Confluence by decreasing diagrams – formalized. In *Proc. 24th International Conference on Rewriting Techniques and Applications*, volume 21 of *Leibniz International Proceedings in Informatics*, pages 352–367, 2013. doi:10.4230/LIPIcs.RTA.2013.352.

[39] H. Zankl. Automating the conversion version of decreasing diagrams for first-order rewrite systems, 2016. Draft, 1 page, `http://cl-informatik.uibk.ac.at/users/hzankl/new/publications/ZA16_01.pdf`.

[40] 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. doi:10.1007/978-3-642-22438-6_38.

[41] H. Zankl, B. Felgenhauer, and A. Middeldorp. Labelings for decreasing diagrams. *Journal of Automated Reasoning*, 54(2):101–133, 2015. doi:10.1007/s10817-014-9316-y.