Formalizing Simultaneous Critical Pairs for Confluence of Left-Linear Rewrite Systems

Christina Kirk
University of Innsbruck
Innsbruck, Austria
christina.kirk@student.uibk.ac.at

Aart Middeldorp
University of Innsbruck
Innsbruck, Austria
aart.middeldorp@uibk.ac.at

Abstract

We report on the formalization of a sufficient condition for confluence of first-order left-linear rewrite systems within the proof assistant Isabelle/HOL. This criterion, originally proposed by Okui (1998), is based on *simultaneous* critical pairs, which finitely represent peaks consisting of a multi-step and a normal step. It properly subsumes the formalized result on development-closed critical pairs.

CCS Concepts: • **Theory of computation → Equational logic and rewriting**; **Logic and verification**.

Keywords: formalization, term rewriting, confluence

ACM Reference Format:
Christina Kirk and Aart Middeldorp. 2025. Formalizing Simultaneous Critical Pairs for Confluence of Left-Linear Rewrite Systems. In *Proceedings of the 14th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP '25), January 20–21, 2025, Denver, CO, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3703595.3705881

1 Introduction

Confluence is one of the most-studied properties of rewrite systems. Numerous sufficient conditions for (non-)confluence are known and implemented in software tools that compete in the annual confluence competition. Recently there has been an increased activity to formalize confluence conditions in proof assistants. In the past two years three papers [9, 11, 12] were published, reporting on the Isabelle/HOL formalization of advanced confluence criteria based on restricted joinability conditions of (parallel) critical pairs for left-linear rewrite systems: (almost) development closedness of critical pairs (van Oostrom [26, 27]) and sufficient confluence conditions based on parallel critical pairs (Gramlich [8], Toyama [25], Shintani and Hirokawa [22]). Extending this line of research, the formalization of Okui's confluence criterion [19] is a natural and significant progression.

CPP '25, January 20–21, 2025, Denver, CO, USA

ACM ISBN 979-8-4007-1347-7/25/01
https://doi.org/10.1145/3703595.3705881

The general idea of critical pair criteria is the following. In order to show confluence, for all peaks $s \,{}^*\!\leftarrow t \rightarrow^* u$ a common reduct v of s and u must be found. The problem is that there can be infinitely many peaks. One solution is to consider only local peaks $s \leftarrow t \rightarrow u$ consisting of two single steps, and to prove that s and u can be joined in a special way (e.g., by a parallel or multi-step. Conditions on the rewrite system (e.g., termination or left-linearity) then ensure that the joinability of local peaks extends to arbitrary peaks. Critical pairs provide a finite and computable description of those local peaks that are critical for confluence.

The starting point of the formalized confluence results in Hirokawa et al. [9] are local peaks $s \leftarrow\!\!\shortparallel t \rightarrow u$ consisting of a parallel step and a single step, which are finitely characterized by so-called parallel critical pairs. In this paper we report on a formalization of the confluence result of Okui [19], in which local peaks $s \leftarrow\!\!\circ t \rightarrow u$ consisting of a multi-step and a single step are considered. Simultaneous critical pairs provide a finite characterization of these. The joinability condition is $s \rightarrow^* v \leftarrow\!\!\circ u$ and, together with left-linearity, guarantees confluence. Okui's result properly generalizes the earlier results of van Oostrom [26, 27], but it is incomparable to the confluence results based on parallel critical pairs. One advantage of using multi-steps, which are also called development steps, is that it opens the way for higher-order rewriting, an important topic which is, however, beyond current formalization efforts. Okui's result is implemented in at least two confluence tools (ACP [1] and CSI [15]), and so our contribution will help close the verification gap in the confluence competition.[1]

One of the major challenges we faced when attempting to formalize Okui's result is to formally define simultaneous critical pairs. Okui [19, Section 3] employs residuals and complete developments. Inspired by the successful formalization [11, 12] of development closed critical pairs [26, 27], it is natural to use *proof terms* [24, 28] which effectively represent multi-steps and have a simple inductive definition. This, however, is not enough. In contrast to parallel critical pairs [9], when computing a most general unifier, the individual equations that make up the unification problem cannot be treated independently because of nesting, so the order in which we compose the substitutions to obtain a most general unifier is crucial. Further, since the same left-hand side can appear

[1] https://project-coco.uibk.ac.at/

multiple times in the construction of a simultaneous critical pair, we must take renamings seriously, and the bookkeeping of these adds another layer of complexity to the formalized proof. Like parallel critical pairs, each simultaneous critical pair includes a root step, but unlike in parallel critical pairs, this root step can occur in either the multi-step or the single step. The position of the root step significantly influences the structure of the simultaneous critical pair, necessitating careful case analysis. This explains why a large part of this paper is devoted to the formal, proof term based, definition of simultaneous critical pairs.

The remainder of this paper is structured as follows. We start with some preliminaries on term rewriting in Section 2. Then we recap proof terms in Section 3 and introduce some new definitions and results. In Section 4 we address the kind of unification problem involved in simultaneous critical pairs. In Section 5 we present simultaneous critical pairs, followed by the formalization of Okui's result in Section 6. We conclude with a discussion of related work in Section 7 and some final remarks as well as directions for future work in Section 8.

One of the aims of this paper is to provide a detailed account of the proof steps involved in formalizing Okui's confluence result, in order to facilitate future extensions or formalizations in other proof assistants. Consequently, some sections of this paper are quite technical and may not appeal to all readers. Readers mainly interested in the final result, may choose to skip the technical details in Sections 3, 4, and 6.1.

We conclude this section with a motivating example.

Example 1.1. The rewrite system consisting of the rules

$$\begin{aligned} \mathsf{g}(\mathsf{f}(\mathsf{b},x)) &\to \mathsf{g}(\mathsf{h}(\mathsf{h}(\mathsf{f}(\mathsf{f}(\mathsf{h}(\mathsf{k}(\mathsf{k}(x,x),x)),\\ &\qquad \mathsf{h}(\mathsf{k}(\mathsf{k}(x,x),x))),\mathsf{h}(\mathsf{k}(\mathsf{k}(x,x),x))))))\\ \mathsf{f}(x,\mathsf{b}) &\to \mathsf{h}(\mathsf{f}(\mathsf{f}(x,x),x))\\ \mathsf{k}(x,\mathsf{b}) &\to \mathsf{f}(x,\mathsf{b}) \end{aligned}$$

has one critical peak starting from the term $\mathsf{g}(\mathsf{f}(\mathsf{b},\mathsf{b}))$ and applying either the first or second rule. The results of [9, 26, 27] do not apply, since nested steps from both directions are needed to close the peak. Okui's criterion, however, is readily applicable, as confirmed by ACP and CSI.[2]

2 Preliminaries

Previous experience with Isabelle is not necessary for following our proof, but familiarity with the basics of term rewriting [2, 24] will be helpful. Below we recall some definitions and notations.

2.1 Term Rewriting

A term rewrite system $\mathcal{R}$ is a set of directed equations, so-called rewrite rules, which induces a relation $\to_{\mathcal{R}}$ on terms.

[2] https://ari-cops.uibk.ac.at/CoCo/2024/full-run/TRS/?q=464.ari

Let $\mathcal{F}$ be a signature (consisting of function symbols f, g, … with associated arities) and $\mathcal{V}$ an infinite set of variables $\{x, y, \dots\}$ disjoint from $\mathcal{F}$. By $\mathcal{T}(\mathcal{F},\mathcal{V})$ we denote the set of terms $s, t, \dots$ over $\mathcal{F}$ and $\mathcal{V}$. A context is a term containing exactly one occurrence of the special constant symbol $\Box$, which is called hole. If C is a context then $C[t]$ denotes the result of replacing the hole by t. Positions are strings of positive natural numbers used to address subterm occurrences. The set of positions of a term t is defined as $\mathcal{P}\mathrm{os}(t) = \{\epsilon\}$ if t is a variable and as $\mathcal{P}\mathrm{os}(t) = \{\epsilon\} \cup \{iq \mid 1 \leqslant i \leqslant n \text{ and } q \in \mathcal{P}\mathrm{os}(t_i)\}$ if $t = f(t_1, \dots, t_n)$. The subterm of t at position $p \in \mathcal{P}\mathrm{os}(t)$ is defined as $t|_p = t$ if $p = \epsilon$ and as $t|_p = t_i|_q$ if $p = iq$ and $t = f(t_1, \dots, t_n)$. We write $s[t]_p$ for the result of replacing the subterm at position p of s with t. The symbol in t at position $p \in \mathcal{P}\mathrm{os}(t)$ is denoted by $t(p)$. 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, positions q and p are parallel, written as $q \parallel p$, if neither $q \leqslant p$ nor $p < q$. We denote the subset of $\mathcal{P}\mathrm{os}(t)$ of non-variable positions (i.e., the positions $p \in \mathcal{P}\mathrm{os}(t)$ such that $t|_p \notin \mathcal{V}$) by $\mathcal{P}\mathrm{os}_{\mathcal{F}}(t)$. We write $\mathcal{V}\mathrm{ar}(t)$ for the set of variables occurring in the term t. A term is linear if every variable occurs at most once in it. Given a linear term t, we write $\mathrm{var}(t)$ for the list $(x_1, \dots, x_n)$ of variables appearing in t in some fixed order. Moreover, $\mathrm{vpos}(t)$ denotes the corresponding list $(p_1, \dots, p_n)$ of positions in t where these variables occur. A substitution is a map σ from $\mathcal{V}$ to $\mathcal{T}(\mathcal{F},\mathcal{V})$ such that its domain $\mathrm{dom}(\sigma) = \{x \in \mathcal{V} \mid \sigma(x) \neq x\}$ is finite. We write $t\sigma$ for the result of applying σ to the term t.

A rewrite rule is a pair of terms (ℓ, r), written $\ell \to r$, such that $\mathcal{V}\mathrm{ar}(r) \subseteq \mathcal{V}\mathrm{ar}(\ell)$ and ℓ is not a variable. A rewrite rule $\ell \to r$ is left-linear if ℓ is linear, it is called erasing if $\mathcal{V}\mathrm{ar}(r) \subsetneq \mathcal{V}\mathrm{ar}(\ell)$. A variant of a rewrite rule is obtained by renaming its variables. A term rewrite system (TRS) is a set of rewrite rules over a signature. In the sequel, signatures are left implicit. A TRS is *left-linear* if all its rules are left-linear. A TRS $\mathcal{R}$ induces the relation $\to_{\mathcal{R}}$ defined on terms as follows: $s \to_{\mathcal{R}} t$ if there exists a position $p \in \mathcal{P}\mathrm{os}(s)$, a rewrite rule $\ell \to r \in \mathcal{R}$ and a substitution σ such that $s|_p = \ell\sigma$ and $t = s[r\sigma]_p$. The subterm $s|_p = \ell\sigma$ is called a *redex*, the pair (p, ℓ) is called a *redex pattern*. Replacing $\ell\sigma$ by $r\sigma$ in s is called *contracting* the redex $\ell\sigma$. Two redex patterns (p_1, ℓ_1) and (p_2, ℓ_2) overlap if $p_1 \leqslant p_2$ and $p_2\backslash p_1 \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(l_2)$, or $p_2 \leqslant p_1$ and $p_1\backslash p_2 \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(l_1)$. Multiple redex-patterns of a term s which do not overlap, can be contracted simultaneously in a *multi-step*. The multi-step relation $\multimap_{\mathcal{R}}$ is inductively defined on terms as follows: $x \multimap_{\mathcal{R}} x$ for all variables x, $f(s_1, \dots, s_n) \multimap_{\mathcal{R}} f(t_1, \dots, t_n)$ if $s_i \multimap_{\mathcal{R}} t_i$ for all $1 \leqslant i \leqslant n$, and $\ell\sigma \multimap_{\mathcal{R}} r\tau$ if $\ell \to r \in \mathcal{R}$ and $\sigma(x) \multimap_{\mathcal{R}} \tau(x)$ for all $x \in \mathcal{V}\mathrm{ar}(\ell)$. From the definition it easily follows that $\to_{\mathcal{R}} \subseteq \multimap_{\mathcal{R}}$ for any TRS $\mathcal{R}$. Whenever the underlying TRS $\mathcal{R}$ is clear from the context, we omit the index in $\multimap_{\mathcal{R}}$ and simply write $\multimap$.

Besides termination, which forbids infinite computations, *confluence* has been conceived as one of the central properties of rewriting. A TRS $\mathcal{R}$ is confluent if for all terms s, t and u such that $s \to_{\mathcal{R}}^* t$ and $s \to_{\mathcal{R}}^* u$ (here $\to_{\mathcal{R}}^*$ denotes the transitive reflexive closure of $\to_{\mathcal{R}}$) there exists a term v such that $t \to_{\mathcal{R}}^* v$ and $u \to_{\mathcal{R}}^* v$. A relation $\to$ is *strongly confluent* if $\leftarrow \cdot \to \subseteq \to^* \cdot \leftarrow$. It is well-known that strong confluence implies confluence and that we can conclude confluence of $\to$ from confluence of a relation $\to_1$ for which $\to_1^* = \to^*$.

Lemma 2.1. *Let $\to$, $\to_1$ and $\to_2$ be binary relations.*

1. *If $\to$ is strongly confluent then $\to$ is confluent.*
2. *If $\to_1 \subseteq \to_2 \subseteq \to_1^*$ and $\to_2$ is confluent then $\to_1$ is confluent.*

When applying this lemma to prove Okui's confluence criterion, we will first instantiate $\to$ with $\nrightarrow$ and establish strong confluence of $\nrightarrow$. By the first item we then obtain confluence of $\nrightarrow$. Finally, we use the second item with the property $\to \subseteq \nrightarrow \subseteq \to^*$ to obtain confluence of $\to$.

2.2 Isabelle/HOL

Our formalization is developed in the proof assistant Isabelle/HOL [18] and is part of the Isabelle Formalization of Rewriting (IsaFoR),[3] a comprehensive library of formalized results for rewriting. Having this robust foundation of definitions and lemmas related to terms, term rewrite systems, and proof terms was crucial for the success of the formalization discussed in this paper. Working with such an established library poses its own challenges however. Designing new definitions in such a way that they fit nicely into the existing framework is crucial for being able to reuse facts. Additionally, it is not always easy to find relevant lemmas for specific goals. Learning to look for the right results with Isabelle's search functionality, especially using type patterns, proved to be an essential skill. Another indispensable tool was Sledgehammer [4, 5], which comes as part of the Isabelle installation. Sledgehammer enables automated proof generation by applying theorem provers and satisfiability-modulo-theories (SMT) solvers to the current goal using hundreds of facts from the current theory context. So instead of hand-selecting and applying appropriate facts, we often leveraged Sledgehammer's selection heuristic and the abilities of external provers like E, SPASS, Vampire, and Zipperposition, as well as the SMT solvers CVC4, veriT, and Z3.

To formalize Okui's simultaneous critical pairs, we heavily utilized Isabelle's contexts and locales. Contexts allow global assumptions and variable declarations within a fixed scope. Locales are named, and therefore reusable, contexts. Moreover, by using *locale expressions* one can compose locale specifications, providing even better modularity [3]. These concepts enabled us to group intermediate results with shared preconditions into compact, self-contained modules, improving code organization and reusability. Further details on this structure are provided in Section 3.4 and Section 6.2.

HTML versions of the theories relevant to this paper are provided via the following URL:

http://informatik-protem.uibk.ac.at/Okui/

The HTML versions have Isabelle syntax highlighting and usages of lemmas are hyperlinked, so one can easily navigate to the results that a certain lemma depends on. We annotated important results in this paper by the Isabelle logo  which directly links to the HTML presentation of the corresponding result.

3 Proof Terms

Proof terms represent computations in term rewriting. They were introduced by van Oostrom and de Vrijer for first-order left-linear rewrite systems to study equivalence of reductions in [28] and [24, Chapter 8]. Proof terms were used in the formalization of development closed critical pairs [11, 12]. The material in this section extends the description in [11] with operations and results required for the formalization of simultaneous critical pairs.

Proof terms are built from function symbols, variables, and rule symbols. Rule symbols represent rewrite rules and have a fixed arity which is the number of different variables in the represented rule. In this way we can represent any multi-step as a proof term. The special case of a proof term with only one rule symbol corresponds to a single step and a proof term without any rule symbols denotes an empty step.

We use Greek letters $(\alpha, \beta, \gamma, \dots)$ for rule symbols and uppercase letters $(A, B, C, \dots)$ for proof terms. If α is a rule symbol then $\mathsf{lhs}(\alpha)$ ($\mathsf{rhs}(\alpha)$) denotes the left-hand (right-hand) side of the rewrite rule denoted by α. Furthermore $\mathsf{var}(\alpha) = \mathsf{var}(\mathsf{lhs}(\alpha))$ and similarly $\mathsf{vpos}(\alpha) = \mathsf{vpos}(\mathsf{lhs}(\alpha))$. The length of this list is the arity of α. Given a rule symbol α with $\mathsf{var}(\alpha) = (x_1, \dots, x_n)$ and terms $t_1, \dots, t_n$, we write $\langle t_1, \dots, t_n \rangle_\alpha$ for the substitution $\{x_i \mapsto t_i \mid 1 \leqslant i \leqslant n\}$. Given a substitution σ we write $\alpha(\sigma)$ to denote $\alpha(\sigma(x_1), \dots, \sigma(x_n))$. Given a proof term A, its source $\mathsf{src}(A)$ and target $\mathsf{tgt}(A)$ are computed by the following clauses:

$$\begin{aligned}
\mathsf{src}(x) &= \mathsf{tgt}(x) = x \\
\mathsf{src}(f(A_1, \dots, A_n)) &= f(\mathsf{src}(A_1), \dots, \mathsf{src}(A_n)) \\
\mathsf{src}(\alpha(A_1, \dots, A_n)) &= \mathsf{lhs}(\alpha)\langle \mathsf{src}(A_1), \dots, \mathsf{src}(A_n) \rangle_\alpha \\
\mathsf{tgt}(f(A_1, \dots, A_n)) &= f(\mathsf{tgt}(A_1), \dots, \mathsf{tgt}(A_n)) \\
\mathsf{tgt}(\alpha(A_1, \dots, A_n)) &= \mathsf{rhs}(\alpha)\langle \mathsf{tgt}(A_1), \dots, \mathsf{tgt}(A_n) \rangle_\alpha
\end{aligned}$$

The proof term A is a witness of the multi-step $\mathsf{src}(A) \nrightarrow \mathsf{tgt}(A)$. For every multi-step there exists a proof term witnessing it. Proof terms A and B are said to be *co-initial* if they have the same source. The following example illustrates these concepts.

[3] http://cl-informatik.uibk.ac.at/isafor

Example 3.1. Consider the left-linear TRS $\mathcal{R}_1$ consisting of the rules:

$$\alpha\colon \mathsf{f}(\mathsf{g}(x), y) \to \mathsf{f}(y, y) \quad \beta\colon \mathsf{g}(\mathsf{a}) \to \mathsf{g}(\mathsf{b}) \quad \gamma\colon \mathsf{a} \to \mathsf{c}$$

and the proof terms

$$A = \mathsf{f}(\mathsf{g}(\mathsf{a}), \beta) \qquad B = \mathsf{f}(\mathsf{g}(\gamma), \mathsf{g}(\mathsf{a})) \qquad C = \alpha(\mathsf{a}, \beta)$$

A, B and C are co-initial. A and B represent single steps

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

and C represents a multi-step

$$\mathsf{f}(\mathsf{g}(\mathsf{a}), \mathsf{g}(\mathsf{a})) \twoheadrightarrow \mathsf{f}(\mathsf{g}(\mathsf{b}), \mathsf{g}(\mathsf{b}))$$

We state some simple results about src and tgt.

Lemma 3.2.

$$\mathcal{V}\mathrm{ar}(\mathrm{src}(A)) = \mathcal{V}\mathrm{ar}(A) \supseteq \mathcal{V}\mathrm{ar}(\mathrm{tgt}(A))$$

Moreover, for any proof term A, A is a linear term if and only if its source $\mathrm{src}(A)$ is a linear term.

Note that $\mathcal{V}\mathrm{ar}(\mathrm{tgt}(A)) = \mathcal{V}\mathrm{ar}(A)$ does not hold in general, since erasing rules can make variables disappear.

In the setting of left-linear TRSs we can extend the definition of src to contexts of proof terms by adding the clause $\mathrm{src}(\Box) = \Box$. Doing the same for tgt or for arbitrary TRSs however could lead to more than one hole appearing in the computation. The following result is an easy consequence of the idempotence of src and tgt.

Lemma 3.3. *For any substitution σ over proof terms, proof term context C, proof term A and simple terms s and t we have*

1. $\mathrm{src}(A\sigma) = \mathrm{src}(A) \cdot (\mathrm{src} \circ \sigma)$
   $\mathrm{tgt}(A\sigma) = \mathrm{tgt}(A) \cdot (\mathrm{tgt} \circ \sigma)$
   $\mathrm{src}(s[A]_p) = s[\mathrm{src}(A)]_p$
   $\mathrm{tgt}(s[A]_p) = s[\mathrm{tgt}(A)]_p$
   $\mathrm{src}(C[A]) = \mathrm{src}(C[\mathrm{src}(A)]) = \mathrm{src}(C)[\mathrm{src}(A)])$
   $\mathrm{tgt}(C[A]) = \mathrm{tgt}(C[\mathrm{tgt}(A)])$
2. *if $s \to^* t$ then* $\mathrm{tgt}(C[s\sigma]) \to^* \mathrm{tgt}(C[t\sigma])$ □

3.1 Operations on Proof Terms

For co-initial proof terms A and B we define the binary *join* ($\sqcup$) and residual ($/$) operations. The residual $A \,/\, B$ is used to compute which redexes in A remain after contracting the redexes of B and $A \sqcup B$ is used to obtain a single proof term containing all redexes of A and B. Note that these are partial operations. The result is defined whenever the redexes of the two proof terms do not interfere with each other. We use the binary *orthogonality* predicate ($\perp$) to model this condition.[4] For working with simultaneous critical pairs we want to combine an arbitrary number of proof terms into a single proof term, so we extend the usual definition by introducing an n-ary join operation ($\bigsqcup$) for $n > 0$.

Definition 3.4. Let A and B be proof terms. The *orthogonality* predicate $A \perp B$, the *residual* operation $A \,/\, B$, the binary *join* operation $A \sqcup B$ and the n-ary join operation

$$\bigsqcup_{i=1}^{n} A_i \quad \text{for } n \geqslant 1$$

are inductively defined by the clauses in Table 1.

Example 3.5. Consider again the TRS $\mathcal{R}_1$ and proof terms A, B, C from Example 3.1. A, B and C are pairwise orthogonal and

$$A \sqcup B \sqcup C = A \sqcup \alpha(\gamma, \beta) = \alpha(\gamma, \beta)$$

Moreover,

$$\mathsf{f}(\beta, \mathsf{g}(\mathsf{a})) \not\perp \alpha(\mathsf{a}, \mathsf{g}(\mathsf{a}))$$
$$A \,/\, B = \mathsf{f}(\mathsf{g}(\mathsf{c}), \beta) \quad \text{and} \quad B \,/\, C = \mathsf{f}(\mathsf{g}(\mathsf{b}), \mathsf{g}(\mathsf{b}))$$

There exist many interesting algebraic properties of the residual and join operations. For example $\sqcup$ is commutative and associative and src is its identity, in the sense that $\mathrm{src}(A) \sqcup A = A \sqcup \mathrm{src}(A) = A$.[5] We present a small selection of results which will be used in the proofs later on.

Lemma 3.6. *For any proof terms A and B, substitution σ and context C:*

1. *If $A \,/\, B$ and $B \,/\, A$ are defined then* $\mathrm{src}(B \,/\, A) = \mathrm{tgt}(A)$ *and* $\mathrm{tgt}(A \,/\, B) = \mathrm{tgt}(B \,/\, A)$.
2. $\mathrm{src}(A) = \mathrm{src}(B) \wedge A\sigma \perp B\sigma \implies A \perp B$
   $C[A] \perp C[B] \implies A \perp B$

The first item of Lemma 3.6 has already been formalized in Isabelle as part of [11], the second item is new and will be used to show that the result of certain computations with $\sqcup$ is well-defined. As mentioned above, $/$, $\sqcup$ and $\bigsqcup$ are partial operations. Orthogonality is a necessary and sufficient condition for definedness, but only the latter is needed in subsequent proofs and has been formalized in Isabelle.

Lemma 3.7. *If A and B are orthogonal then $A \,/\, B$ and $A \sqcup B$ are defined.* □

Lemma 3.8. *If $n > 0$ and $A_i \sqcup A_j$ is defined for all $1 \leqslant i, j \leqslant n$ then $\bigsqcup_{i=1}^{n} A_i$ is defined and*

$$\mathrm{src}\Big(\bigsqcup_{i=1}^{n} A_i\Big) = \mathrm{src}(A_1) = \cdots = \mathrm{src}(A_n)$$

We briefly describe the well-definedness proof since it requires more than just a straightforward induction.

Proof (sketch) for Lemma 3.8. We use induction on n.

[4] In contrast to previous works on proof terms [10, 11], we consider two co-initial proof terms with the same rule symbols to be orthogonal.

[5] So proof terms with join form a so-called *partial Abelian monoid*.

Table 1. Orthogonality, residual and join of proof terms.

$$
\begin{aligned}
x \perp x \quad x \,/\, x = x \quad x \sqcup x &= x \\
f(A_1,\dots,A_n) \perp f(B_1,\dots,B_n) &\iff A_i \perp B_i \quad \text{for all } 1 \leqslant i \leqslant n \\
\alpha(A_1,\dots,A_n) \perp \mathsf{lhs}(\alpha)\langle B_1,\dots,B_n\rangle_\alpha &\iff A_i \perp B_i \quad \text{for all } 1 \leqslant i \leqslant n \\
\mathsf{lhs}(\alpha)\langle A_1,\dots,A_n\rangle_\alpha \perp \alpha(B_1,\dots,B_n) &\iff A_i \perp B_i \quad \text{for all } 1 \leqslant i \leqslant n \\
\alpha(A_1,\dots,A_n) \perp \alpha(B_1,\dots,B_n) &\iff A_i \perp B_i \quad \text{for all } 1 \leqslant i \leqslant n \\
f(A_1,\dots,A_n) \,/\, f(B_1,\dots,B_n) &= f(A_1 \,/\, B_1,\dots,A_n \,/\, B_n) \\
\alpha(A_1,\dots,A_n) \,/\, \alpha(B_1,\dots,B_n) &= \mathsf{rhs}(\alpha)\langle A_1 \,/\, B_1,\dots,A_n \,/\, B_n\rangle_\alpha \\
\alpha(A_1,\dots,A_n) \,/\, \mathsf{lhs}(\alpha)\langle B_1,\dots,B_n\rangle_\alpha &= \alpha(A_1 \,/\, B_1,\dots,A_n \,/\, B_n) \\
\mathsf{lhs}(\alpha)\langle A_1,\dots,A_n\rangle_\alpha \,/\, \alpha(B_1,\dots,B_n) &= \mathsf{rhs}(\alpha)\langle A_1 \,/\, B_1,\dots,A_n \,/\, B_n\rangle_\alpha \\
f(A_1,\dots,A_n) \sqcup f(B_1,\dots,B_n) &= f(A_1 \sqcup B_1,\dots,A_n \sqcup B_n) \\
\alpha(A_1,\dots,A_n) \sqcup \alpha(B_1,\dots,B_n) &= \alpha(A_1 \sqcup B_1,\dots,A_n \sqcup B_n) \\
\alpha(A_1,\dots,A_n) \sqcup \mathsf{lhs}(\alpha)\langle B_1,\dots,B_n\rangle_\alpha &= \alpha(A_1 \sqcup B_1,\dots,A_n \sqcup B_n) \\
\mathsf{lhs}(\alpha)\langle A_1,\dots,A_n\rangle_\alpha \sqcup \alpha(B_1,\dots,B_n) &= \alpha(A_1 \sqcup B_1,\dots,A_n \sqcup B_n) \\
\bigsqcup(A_1,\dots,A_n) &= \begin{cases} A_1 & \text{if } n = 1 \\ A_1 \sqcup \bigsqcup(A_2,\dots,A_n) & \text{if } n > 1 \end{cases}
\end{aligned}
$$

- For $n = 1$ the statement holds by definition:

$$\bigsqcup_{i=1}^{n} A_i = A_1$$

- Assume $n > 1$. We can apply the induction hypothesis to $A_2,\dots,A_n$, yielding a well-defined proof term A' which is the result of computing $\bigsqcup_{i=2}^{n} A_i$. It remains to show that $A_1 \sqcup A'$ is defined. Without more information about A' this is not possible. So we introduce the following fact:

$$\begin{aligned}&\text{If } A \sqcup B,\ B \sqcup C, \text{ and } A \sqcup C \text{ are defined,}\\ &\qquad\text{then also } A \sqcup (B \sqcup C) \text{ is defined.}\end{aligned} \tag{1}$$

  The proof of (1) is by induction on the structure of A and exhaustive case analysis. Proceeding with the main proof, first observe that if $n = 2$ we are immediately done since $A_1 \sqcup A_2$ is defined by assumption. Otherwise, we additionally establish definedness of

$$A_2 \sqcup \bigsqcup_{i=3}^{n} A_i \qquad \text{and} \qquad A_1 \sqcup \bigsqcup_{i=3}^{n} A_i$$

  by applying the induction hypothesis two times. Then definedness of

$$\bigsqcup_{i=1}^{n} A_i = A_1 \sqcup (A_2 \sqcup \bigsqcup_{i=3}^{n} A_i)$$

  follows from (1). □

3.2 Redex Patterns and Overlapping Proof Terms

We introduce definitions which are used to reason about the redex patterns of a proof term and overlap between the redex patterns of two proof terms.

Definition 3.9. Given a proof term A we compute the set of redex patterns inductively:

$$
\begin{aligned}
\mathsf{RP}(x) &= \varnothing \\
\mathsf{RP}(f(A_1,\dots,A_n)) &= \bigcup_{i=1}^{n} \{\, (\alpha, ip) \mid (\alpha, p) \in \mathsf{RP}(A_i) \,\} \\
\mathsf{RP}(\alpha(A_1,\dots,A_n)) &= \{\, (\alpha, \epsilon) \,\} \cup \\
&\qquad \bigcup_{i=1}^{n} \{\, (\beta, p_i p) \mid (\beta, p) \in \mathsf{RP}(A_i) \,\}
\end{aligned}
$$

where $\mathsf{vpos}(\alpha) = (p_1,\dots,p_n)$ in the last item.

It is easy to see that for a term s without rule symbols we have $\mathsf{RP}(s) = \varnothing$ and if $p \in \mathcal{P}\mathrm{os}(s)$ then $\mathsf{RP}(s[A]_p) = \{\, (\alpha, pq) \mid (\alpha, q) \in \mathsf{RP}(A) \,\}$. Moreover, the set of redex patterns for proof terms obtained by the join operation can be computed by the following lemma.

Lemma 3.10. $\mathsf{RP}(\bigsqcup_{i=1}^{n} A_i) = \bigcup_{i=1}^{n} \mathsf{RP}(A_i)$ □

Definition 3.11. Let $\mathsf{src}(A) = s$ and $(\alpha, p) \in \mathsf{RP}(A)$ with $\mathsf{vpos}(\alpha) = (p_1,\dots,p_n)$. The single rewrite step $\Delta(s, \alpha, p)$ corresponding to this redex pattern is defined as

$$\Delta(s, \alpha, p) := s[\alpha(s|_{pp_1},\dots,s|_{pp_n})]_p$$

Two single steps extracted from the same proof term are either equal or orthogonal.

Lemma 3.12. *If* $(\alpha, p), (\beta, q) \in \mathsf{RP}(A)$ *and*

$$\Delta_1 = \Delta(\mathsf{src}(A), \alpha, p) \qquad \Delta_2 = \Delta(\mathsf{src}(A), \beta, q)$$

then $\Delta_1 \perp \Delta_2$.

Note that this means that $\Delta_1 \sqcup \Delta_2$ is always defined when Δ_1 and Δ_2 have been extracted from the same proof term.

The main proof of Okui's confluence criterion is based on a case analysis on the amount of overlap between two co-initial proof terms. We recap the special labeling of the source of a proof term introduced in [10], which gives rise to an inductive definition of the amount of overlap between two co-initial proof terms.

Definition 3.13. We write $\mathsf{lhs}^\sharp(\alpha)$ for the result of labeling every function symbol in $\mathsf{lhs}(\alpha)$ with α as well as the distance to the root of α:

$$\mathsf{lhs}^\sharp(\alpha) = \varphi(\mathsf{lhs}(\alpha), \alpha, 0)$$

with $\varphi(t, \alpha, i) = t$ if $t \in \mathcal{V}$ and

$$\varphi(t, \alpha, i) = f_{\alpha^i}(\varphi(t_1, \alpha, i+1), \ldots, \varphi(t_n, \alpha, i+1))$$

if $t = f(t_1, \ldots, t_n)$. The mapping $\mathsf{src}^\sharp$ computes the labeled source of a proof term: $\mathsf{src}^\sharp(A) =$

$$\begin{cases} A & \text{if } A \in \mathcal{V} \\ f(\mathsf{src}^\sharp(A_1), \ldots, \mathsf{src}^\sharp(A_n)) & \text{if } A = f(A_1, \ldots, A_n) \\ \mathsf{lhs}^\sharp(\alpha)\langle \mathsf{src}^\sharp(A_1), \ldots, \mathsf{src}^\sharp(A_n)\rangle_\alpha & \text{if } A = \alpha(A_1, \ldots, A_n) \end{cases}$$

The function ℓ extracts labels from function symbols: $\ell(f_{\alpha^n}) = \alpha^n$. The set of labeled positions for a proof term A is defined as

$$\mathcal{P}\mathsf{os}_L(A) = \{p \in \mathcal{P}\mathsf{os}(\mathsf{src}^\sharp(A)) \mid \ell(\mathsf{src}^\sharp(A)(p)) \text{ is defined}\}$$

There exists a straightforward connection between $\mathsf{src}^\sharp(A)$ and $\mathsf{RP}(A)$.

Lemma 3.14. $(\alpha, p) \in \mathsf{RP}(A) \iff \ell(\mathsf{src}^\sharp(A)(p)) = \alpha^0$ □

In the lemma below $\mathsf{src}(A)[\,]_p$ turns the term $\mathsf{src}(A)$ into a context by replacing the subterm at position p into a hole.

Lemma 3.15. *If* $\mathsf{src}^\sharp(A)|_p$ *is unlabeled or* $\ell(\mathsf{src}^\sharp(A)(p)) = \alpha^0$ *then there exists a position* $q \in \mathcal{P}\mathsf{os}(A)$ *such that*

$$\mathsf{src}(A)[\,]_p = \mathsf{src}(A[\,]_q) \tag{2}$$

$$\mathsf{src}(A)|_p = \mathsf{src}(A|_q) \tag{3}$$

$$\mathsf{src}^\sharp(A) = \mathsf{src}^\sharp(A)[\mathsf{src}^\sharp(A|_q)]_p \tag{4}$$

Definition 3.16. For co-initial proof terms A and B we use the number of positions that are labeled in both $\mathsf{src}^\sharp(A)$ and $\mathsf{src}^\sharp(B)$ as a measure for the amount of overlap between A and B:

$$\blacktriangle(A, B) = |\mathcal{P}\mathsf{os}_L(A) \cap \mathcal{P}\mathsf{os}_L(B)|$$

Lemma 3.17. *For co-initial proof terms* A *and* B

$$\blacktriangle(A, B) = 0 \implies A \perp B$$

Note that the reverse direction of Lemma 3.17 does not hold since, e.g. $A \perp A$ but $\blacktriangle(A, A) > 0$ if A contains rule symbols.

3.3 Matching for Proof Terms

First consider the problem of matching linear terms in general. It is quite obvious that a linear term t matches a term s if all function symbols of t coincide with the function symbols of s. Then a matching substitution can be easily computed, as described in the following lemma.

Lemma 3.18. *Consider terms* s *and* t *and let*

$$\sigma = \{x \mapsto s|_p \mid p \in \mathcal{P}\mathsf{os}_\mathcal{V}(t) \wedge t|_p = x\}$$

If t *is a linear term and for all* $p \in \mathcal{P}\mathsf{os}_\mathcal{F}(t)$ *we have* $p \in \mathcal{P}\mathsf{os}_\mathcal{F}(s)$ *with* $t(p) = s(p)$, *then* $t\sigma = s$. □

Since proof terms are also just first-order terms, this result directly holds for proof terms. So if all function symbols and rule symbols of a linear proof term A coincide with the function symbols and rule symbols of a proof term B, then A matches B. However, when we want to apply this result in the main proof later on, it is easier to argue about the function symbols in the sources of A and B as well as about the elements in $\mathsf{RP}(A)$ and $\mathsf{RP}(B)$. We introduce the following lemma to facilitate that argument.

Lemma 3.19. *Assume that* A *is a linear proof term,* $\mathsf{src}(A)$ *matches* $\mathsf{src}(B)$, $\mathsf{RP}(A) \subseteq \mathsf{RP}(B)$, *and if* $(\alpha, p) \in \mathsf{RP}(B)$ *and* $p \in \mathcal{P}\mathsf{os}(A) \setminus \mathcal{P}\mathsf{os}_\mathcal{V}(A)$ *then* $(\alpha, p) \in \mathsf{RP}(A)$. *Then* A *matches* B *as witnessed by the substitution*

$$\sigma = \{x \mapsto B|_p \mid p \in \mathcal{P}\mathsf{os}_\mathcal{V}(A) \wedge A|_p = x\}$$

The assumptions in Lemma 3.19 are just an alternative way of stating the conditions of Lemma 3.18 specifically for proof terms. They ensure that each rule symbol and each function symbol of A is present at the exact same position in B.

3.4 Formalization Details

All results described in this chapter have been formalized in Isabelle/HOL as part of the IsaFoR library. Proof terms themselves and many useful properties are included in the library since the formalization efforts described in [11]. We only had to extend the existing theories with some additional results, like the first part of Lemma 3.6. Moreover, the definition of $\mathsf{RP}(A)$ is new. Its formalized version as well as relevant lemmas are collected in the file `Redex_Patterns.thy`. This is a completely new part of the IsaFoR session on proof terms. In total, we extended the proof term session with about 4000 lines of new Isabelle code.

Note that, by default, IsaFoR does not enforce the variable conditions on rewrite rules. This means that variable left-hand sides and variables that appear only on the right-hand side are allowed. While many results in term rewriting can

be formulated in this more general setting, several key results concerning proof terms require the strict conditions, as well as left-linearity of the underlying TRS. The following examples illustrate this.

- The condition $\mathcal{V}\mathrm{ar}(\mathrm{rhs}(\alpha)) \subseteq \mathcal{V}\mathrm{ar}(\mathrm{lhs}(\alpha))$ is crucial whenever residuals are involved, since right-hand sides need to be instantiated during the residual computation.
- The condition $\mathrm{lhs}(\alpha) \notin \mathcal{V}$ plays a role when considering orthogonality of proof terms. For instance, if $\mathrm{lhs}(\alpha) = x$ were allowed, then, according to the definition, $\alpha(\beta) \perp \beta$ for any rule symbol β, but $\alpha(\beta) \sqcup \beta$ is undefined.
- Left-linearity is essential when dealing with $\mathrm{src}^\sharp$. Consider for example the non-left-linear TRS consisting of the rules: $\alpha\colon \mathsf{f}(x,x) \to \mathsf{g}(x,x)$, $\beta\colon \mathsf{a} \to \mathsf{b}$ and $\gamma\colon \mathsf{a} \to \mathsf{c}$. Then $\blacktriangle(\mathsf{f}(\beta,\gamma), \alpha(\mathsf{a})) = 0$ but $\mathsf{f}(\beta,\gamma) \not\perp \alpha(\mathsf{a})$.

We aimed to adhere to the philosophy of keeping results in IsaFoR as general as possible, i.e., require only the conditions that are really necessary. Initially, we stated these conditions explicitly as assumptions in the corresponding lemmas. However, this made instantiations of these lemmas more tedious, and in particular hindered Sledgehammer's ability to automatically solve subgoals. To address this, we introduced locales for each individual condition, as well as combined locales for cases where multiple conditions are needed. While this approach does improve reusability and modularity, it is still not ideal. As can be seen in Listing 1, the locale `left_lin_wf_trs` does not directly contain `left_lin_no_var_lhs`. So when proving results within `left_lin_wf_trs`, the conditions from `left_lin_no_var_lhs` must still be instantiated explicitly. Furthermore, some proofs could have been simplified by requiring stricter conditions upfront. So in hindsight, some complexity could have been avoided by placing the entire theory on proof terms within a single locale that enforces all three necessary conditions. As is often the case, the current structure reflects historical development.

4 Unification

We reuse the following definition from [11].

Definition 4.1. For linear terms s and t without common variables, let $\mathrm{vpos}(s) = (p_1, \ldots, p_n)$, $\mathrm{var}(s) = (x_1, \ldots, x_n)$, $\mathrm{vpos}(t) = (q_1, \ldots, q_m)$, and $\mathrm{var}(t) = (y_1, \ldots, y_m)$. The substitution $\tau(s,t)$ is defined as follows:

$$\begin{aligned}\tau(s,t) = \{x_i \mapsto t|_{p_i} \mid 1 \leqslant i \leqslant n \text{ and } p_i \in \mathcal{P}\mathrm{os}(t)\} \\ \cup \{y_j \mapsto s|_{q_j} \mid 1 \leqslant j \leqslant m \text{ and } q_j \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(s)\}\end{aligned}$$

In [11] it was shown that $\tau(s,t)$ is an mgu of s and t if s and t are linear unifiable terms without common variables. When working with mgus for simultaneous critical pairs, computing a single substitution τ like above is not enough.

```
locale left_lin =
 fixes R :: "('f, 'v) trs"
 assumes "left_linear_trs R"

locale no_var_lhs =
 fixes R :: "('f, 'v) trs"
 assumes "Ball R (λ(l, r). is_Fun l)"

locale var_rhs_subset_lhs =
 fixes R :: "('f, 'v) trs"
 assumes "Ball R (λ(l, r). vars_term r ⊆ vars_term l)"

locale wf_trs = no_var_lhs + var_rhs_subset_lhs
locale left_lin_no_var_lhs = left_lin + no_var_lhs
locale left_lin_wf_trs = left_lin + wf_trs
```

Listing 1. Locales for variable conditions on rewrite rules.

We need to unify the n left-hand sides appearing in the multi-step on the left with the single left-hand side of the step on the right. Moreover, the n individual unification problems are not independent (as is the case for unification in parallel critical pairs), since the rewrite rules on the left can be nested. In our case we need an mgu for a set E of equations of the following shape:

$$E = \{s_1 \approx t|_{p_1}, \ldots, s_n \approx t|_{p_n}\}$$

where

$$\begin{aligned} s_1, \ldots, s_n, t &\quad \text{are linear terms} \\ p_1, \ldots, p_n &\quad \text{are positions in } t \\ \mathcal{V}\mathrm{ar}(s_i) \cap \mathcal{V}\mathrm{ar}(s_j) = \varnothing &\quad \text{for all } 1 \leqslant i < j \leqslant n \\ \mathcal{V}\mathrm{ar}(s_i) \cap \mathcal{V}\mathrm{ar}(t) = \varnothing &\quad \text{for all } 1 \leqslant i \leqslant n \end{aligned} \tag{5}$$

We need to work with a concrete instance of an mgu for the set E later on (Section 6) and will use the composed substitution $\tau(s_1, t|_{p_1}) \cdots \tau(s_n, t|_{p_n})$ for this purpose. The following lemma introduces some general properties of this composed substitution.

Lemma 4.2. *Assume we have a set of equations E for which the conditions (5) hold. If $\tau = \tau_1 \cdots \tau_n$ where $\tau_i = \tau(s_i, t|_{p_i})$ for all $1 \leqslant i \leqslant n$ then $\mathrm{dom}(\tau) \subseteq \mathcal{V}\mathrm{ar}(E)$ and, for $x \in \mathrm{dom}(\tau)$,*

1. *if $x = s_i|_p$ for some $1 \leqslant i \leqslant n$ and $p \in \mathcal{P}\mathrm{os}_{\mathcal{V}}(s_i)$ then $\tau(x) = t|_{p_i q}\tau_{i+1} \cdots \tau_n$ for some $q \in \mathcal{P}\mathrm{os}(t|_{p_i})$,*
2. *if $x = t|_q$ for some $q \in \mathcal{P}\mathrm{os}_{\mathcal{V}}(t)$ then $\tau(x) = s_i|_p$ for some $1 \leqslant i \leqslant n$ and $p \in \mathcal{P}\mathrm{os}(s_i)$ with $p_i p = q$.*

Note that computing the result of $\tau(x)$ for $x \in \mathcal{V}\mathrm{ar}(s_i)$ is the more complex of the two cases, since it is possible that some variables in $t|_{p_i q}$ are affected by some of the $\tau_{i+1}, \ldots, \tau_n$ (see Example 4.3). In the case $x \in \mathcal{V}\mathrm{ar}(t)$, the term $\tau(x) = s_i|_p$ has distinct variables from all terms $s_{i+1}, \ldots, s_n$ and t, and therefore $s_i|_p\tau_{i+1} \cdots \tau_n = s_i|_p$.

Example 4.3. Consider the terms

$$s_1 = \mathsf{f}(x_1, x_2) \qquad s_2 = \mathsf{g}(\mathsf{a}) \qquad t = \mathsf{f}(\mathsf{g}(y_1), y_2)$$

and positions $p_1 = \epsilon$, $p_2 = 1$. We obtain

$$\tau_1 = \tau(s_1, t|_{p_1}) = \{x_1 \mapsto \mathsf{g}(y_1),\ x_2 \mapsto y_2\}$$
$$\tau_2 = \tau(s_2, t|_{p_2}) = \{y_1 \mapsto \mathsf{a}\}$$

Hence $x_1\tau_1\tau_2 = \mathsf{g}(y_1)\tau_2 = \mathsf{g}(\mathsf{a})$, so the value of x_1 under the substitution τ is affected by both τ_1 and τ_2. Moreover, the order in which the individual substitutions are composed is critical for unification: $s_1\tau_1\tau_2 = \mathsf{f}(\mathsf{g}(\mathsf{a}), y_2) = t\tau_1\tau_2$ and $s_2\tau_1\tau_2 = \mathsf{g}(\mathsf{a}) = t|_1\tau_1\tau_2$ but $s_1\tau_2\tau_1 = \mathsf{f}(\mathsf{g}(y_1), y_2) \neq \mathsf{f}(\mathsf{g}(\mathsf{a}), y_2) = t\tau_2\tau_1$

Now consider a general unification problem $E = \{s_1 \approx t_1, \ldots, s_n \approx t_n\}$ and assume that τ_i is a unifier for $s_i \approx t_i$ for all $i \in \{1, \ldots, n\}$. As shown in Example 4.3, the composition $\tau = \tau_1 \cdots \tau_n$ is not necessarily a unifier of E. By inspecting the unification algorithm in [2, Chapter 4] we see that a composition of this shape is an mgu of E if each new variable binding, introduced in an *Eliminate* step, has no effect on the remaining equations. This condition is satisfied whenever $s_1, \ldots, s_n, t_1, \ldots, t_n$ are linear terms with pairwise disjoint variables. This was one of the key properties used to show that $\tau(s, t)$ is an mgu of two unifiable, linear and variable disjoint terms s, t in [11, Lemma 6.2]. Since we do not have the luxury of pairwise disjoint variables, we now use the more general requirements $\mathcal{V}\mathsf{ar}(s_i) \cap \mathcal{V}\mathsf{ar}(t_i) = \varnothing$ and $s_j\tau_i = s_j$ and $t_j\tau_i = t_j$ for all $1 \leqslant i < j \leqslant n$.

Lemma 4.4. *Let $E = \{s_1 \approx t_1, \ldots, s_n \approx t_n\}$ where*

1. *E is unifiable*
2. *$s_1, \ldots, s_n, t_1, \ldots, t_n$ are linear terms*
3. *$\mathcal{V}\mathsf{ar}(s_i) \cap \mathcal{V}\mathsf{ar}(t_i) = \varnothing$ for all $1 \leqslant i \leqslant n$*
4. *$s_j\tau(s_i, t_i) = s_j$ and $t_j\tau(s_i, t_i) = t_j$ for all $1 \leqslant i < j \leqslant n$*

Then the unification algorithm yields $\tau(s_1, t_1) \cdots \tau(s_n, t_n)$ as an mgu for E.

Proof (sketch). The proof is an extension of the proof of [11, Lemma 6.2]. It uses the same structure, i.e., induction over the definition of the unification algorithm in [2, Chapter 4]. Compared to the proof of [11, Lemma 6.2], the order in which we look at equations now matters. The main difficulty, however, lies in showing that the fourth condition remains true after a decomposition step. □

5 Simultaneous Critical Pairs

We propose a definition of simultaneous critical pairs (SCPs for short) which is very close to Felgenhauer's definition in [6]. It is more suitable to formalizing than Okui's definition in [19]. Considering that the formalization relies heavily on proof terms, we define an SCP as a pair of proof terms, not just terms. This has the advantage that the SCP itself contains all the necessary information about the redexes that are involved in it. The idea is that an SCP (A, B) consists of a proof term A representing a non-empty multi-step and a proof term B representing a single step. One of the essential properties of an SCP (A, B) is that each redex pattern that occurs in A needs to have overlap with the single redex pattern occurring in B. To model this fact we introduce a function OV which first uses RP to obtain all single steps $A_1, \ldots, A_n$ contained in A and then filters out all A_i which do not have overlap with B. Finally, the single steps are combined into a multi-step again using n-fold join ($\bigsqcup$).

Definition 5.1.

$$\mathsf{OV}(A, B) = \bigsqcup \{\Delta(\mathsf{src}(A), \alpha, p) \mid (\alpha, p) \in \mathsf{RP}(A) \text{ and } \blacktriangle(\Delta(\mathsf{src}(A), \alpha, p), B) \neq 0\}$$

From Lemma 3.12 and Lemma 3.7(2) we can immediately conclude that the n-fold join in this definition is always well-defined. From Lemma 3.8(3) we infer that $\mathsf{src}(\mathsf{OV}(A, B)) = \mathsf{src}(A)$.

Now we have all the ingredients for SCPs. Below, we present the definition that has been formalized in Isabelle.

Definition 5.2. Let $\mathcal{R}$ be a left-linear term rewrite system. A *simultaneous critical pair* of $\mathcal{R}$ is a pair of proof terms (A, B) such that:

1. $\mathsf{RP}(A) = \{(\alpha_1, p_1), \ldots, (\alpha_n, p_n)\}$
2. $\mathsf{RP}(B) = \{(\beta, q)\}$
3. there are renamings $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}$ and σ_β such that
   $$\mathcal{V}\mathsf{ar}(\mathsf{lhs}(\alpha_i)\sigma_{\alpha_i}) \cap \mathcal{V}\mathsf{ar}(\mathsf{lhs}(\beta)\sigma_\beta) = \varnothing$$
   $$\mathcal{V}\mathsf{ar}(\mathsf{lhs}(\alpha_i)\sigma_{\alpha_i}) \cap \mathcal{V}\mathsf{ar}(\mathsf{lhs}(\alpha_j)\sigma_{\alpha_j}) = \varnothing$$
   for all $1 \leqslant i, j \leqslant n$ and $i \neq j$
4. $\mathsf{OV}(A, B) = A$
5. $q = \epsilon$ or $p_1 = \epsilon$
6. $\ell = \mathsf{lhs}(\alpha_1)\sigma_{\alpha_1}[\mathsf{lhs}(\beta)\sigma_\beta]_q$
7. the set of equations
   $$\{\mathsf{lhs}(\alpha_1)\sigma_{\alpha_1} \approx \ell|_{p_1}, \ldots, \mathsf{lhs}(\alpha_n)\sigma_{\alpha_n} \approx \ell|_{p_n}\}$$
   is unifiable and τ is a most general unifier
8. $A = \bigsqcup_{i=1}^{n} A_i$ where $A_i = \ell\tau[\alpha_i(\sigma_{\alpha_i}\tau)]_{p_i}$ for $1 \leqslant i \leqslant n$
9. $B = \ell\tau[\beta(\sigma_\beta\tau)]_q$

Note that $n \geqslant 1$ by item 8 since otherwise the join operation would be undefined. Moreover, $\mathsf{src}(A) = \mathsf{src}(B)$ since $\mathsf{src}(A_i) = \ell\tau$ for all $1 \leqslant i \leqslant n$ and $\mathsf{src}(B) = \ell\tau$. For an SCP (A, B) we call the peak $\mathsf{tgt}(A) \leftarrow\!\!\circ\!\!- \mathsf{src}(A) = \mathsf{src}(B) \rightarrow \mathsf{tgt}(B)$ a *simultaneous critical peak*.

Contrary to Okui, we allow variants of the same rule at the root. This does not affect the main theorem, since joinability of critical pairs in such cases is always trivial. However, it simplifies the formalized proofs, as it eliminates the need to address this special case separately. As previously mentioned, our definition is almost identical to Felgenhauer's in [6], the difference being only the use of proof terms to describe rewrite steps, and a small variation in the way we define the unification problem. The challenge of formulating

a concise description of the unification problem lies in the fact, that the root step can occur on either the left or the right side. Felgenhauer addressed this by relying on differences of positions, we chose to introduce the term ℓ (item 6) to capture both possibilities.

Example 5.3. Consider the left-linear TRS $\mathcal{R}_2$ consisting of the single rule [13]

$$\alpha : \mathsf{f}(\mathsf{f}(x)) \to \mathsf{f}(\mathsf{g}(\mathsf{f}(x), \mathsf{f}(x)))$$

There exists the SCP

$$(A, B) = \big(\alpha(\alpha(x_2)), \mathsf{f}(\alpha(\mathsf{f}(x_2)))\big)$$

We illustrate how (A, B) fulfills the 9 conditions in the definition above.

1. $\mathrm{RP}(A) = \{(\alpha_1, \epsilon), (\alpha_2, 11)\}$ where $\alpha_1 = \alpha_2 = \alpha$
2. $\mathrm{RP}(B) = \{(\beta, 1)\}$ where $\beta = \alpha$
3. We define renamings σ_{α_1}, σ_{α_2} and σ_β:

$$\sigma_{\alpha_1} = \{x \mapsto x_1\} \quad \sigma_{\alpha_2} = \{x \mapsto x_2\} \quad \sigma_\beta = \{x \mapsto y\}$$

4. We have

$$\blacktriangle(\Delta(\mathrm{src}(A), \alpha, \epsilon), B) = \blacktriangle(\alpha(\mathsf{f}(\mathsf{f}(x_2))), B) = 1$$
$$\blacktriangle(\Delta(\mathrm{src}(A), \alpha, 11), B) = \blacktriangle(\mathsf{f}(\mathsf{f}(\alpha(x_2))), B) = 1$$
$$\alpha(\mathsf{f}(\mathsf{f}(x_2))) \sqcup \mathsf{f}(\mathsf{f}(\alpha(x_2))) = \alpha(\alpha(x_2)) = A$$

   and hence $\mathrm{OV}(A, B) = A$.
5. $p_1 = \epsilon$
6. $\ell = \mathsf{f}(\mathsf{f}(\mathsf{f}(y)))$
7. We compute a most general unifier τ:

$$\begin{aligned}\tau &= \mathrm{mgu}(\{\mathsf{f}(\mathsf{f}(x_1)) \approx \mathsf{f}(\mathsf{f}(\mathsf{f}(y))), \mathsf{f}(\mathsf{f}(x_2)) \approx \mathsf{f}(y)\}) \\ &= \{x_1 \mapsto \mathsf{f}(y)\}\{y \mapsto \mathsf{f}(x_2)\} \\ &= \{x_1 \mapsto \mathsf{f}(\mathsf{f}(x_2)), y \mapsto \mathsf{f}(x_2)\}\end{aligned}$$

8. $A = A_1 \sqcup A_2$ where

$$A_1 = \alpha(\sigma_{\alpha_1}\tau) = \alpha(\mathsf{f}(\mathsf{f}(x_2)))$$
$$A_2 = \mathsf{f}(\mathsf{f}(\alpha(\sigma_{\alpha_2}\tau))) = \mathsf{f}(\mathsf{f}(\alpha(x_2)))$$

9. $B = \mathsf{f}(\beta(\sigma_\beta\tau))$

The number of SCPs for any TRS with a finite number of rewrite rules is finite (modulo renaming of the variables). Computing SCPs can, however, be very expensive. There are at least twice as many SCPs as there are standard critical pairs, since the root step can occur on both sides.

Example 5.4 ([14])**.** Consider the TRS $\mathcal{R}_3$ consisting of the two rules

$$\alpha : \mathsf{f}(x) \to \mathsf{g}(x, \mathsf{f}(x)) \quad \beta : \mathsf{f}(\mathsf{f}(\mathsf{f}(x))) \to \mathsf{f}(\mathsf{f}(\mathsf{g}(x, \mathsf{f}(x))))$$

The tool CSI[6] computes 26 SCPs. Among them are

$$\big(\alpha(\mathsf{f}(\mathsf{f}(x))), \beta(x)\big), \ \big(\beta(x), \alpha(\mathsf{f}(\mathsf{f}(x)))\big),$$
$$\big(\alpha(\mathsf{f}(\alpha(x))), \beta(x)\big), \ \big(\mathsf{f}(\alpha(\alpha(x))), \beta(x)\big),$$
$$\big(\alpha(\mathsf{f}(\beta(x))), \beta(\mathsf{f}(\mathsf{f}(x)))\big)$$

[6] http://cl-informatik.uibk.ac.at/software/csi/

Adding another nested f to the rule β increases the number of SCPs to 58.

This example illustrates why it is extremely beneficial to be able to automatically verify and certify results that rely on SCPs. Computing simultaneous critical pairs by hand is simply not feasible in most cases.

5.1 Formalization Details

For the renamings $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}, \sigma_\beta$ we adopt the renaming scheme introduced in [9] for parallel critical pairs. I.e., we simply put the definition of an SCP into a locale where two appropriate renaming functions `ren1` : $\mathbb{N} \times V \to V$ and `ren2` : $V \to V$ are assumed to exist. The function `ren1` can be instantiated with an index i to obtain the renaming σ_{α_i}, while `ren2` is used for σ_β. We opted not to use the wrapper function `mgu_vd_list` which is used for parallel critical pairs and combines unification with renaming. Instead, we explicitly apply renaming first and then do unification. This approach allows us to construct the term ℓ using the renamed versions of $\mathrm{lhs}(\alpha_1)$ and $\mathrm{lhs}(\beta)$, ensuring that ℓ is a linear term.

6 Confluence via SCPs

Okui has shown in 1998 that a term rewrite system is confluent if all SCPs are strongly closed with respect to the multi-step relation ($\multimap$) [19].

Definition 6.1. A simultaneous critical peak $t \leftarrow\!\!\circ\ s \to u$ is strongly closed (with respect to $\multimap$) if $t \to^* v \leftarrow\!\!\circ\ u$ for some term v.

Theorem 6.2 (Okui 1998)**.** *A left-linear TRS $\mathcal{R}$ is confluent if every SCP of $\mathcal{R}$ is strongly closed.*

Example 6.3. Consider again Example 5.3. The SCP (A, B) is strongly closed. The proof terms

$$C = \mathsf{f}(\mathsf{g}(\alpha(\mathsf{g}(\mathsf{f}(x), \mathsf{f}(x))), \alpha(\mathsf{g}(\mathsf{f}(x), \mathsf{f}(x)))))$$
$$D = \alpha(\mathsf{g}(\alpha(x), \alpha(x)))$$

witness rewrite sequences $\mathrm{tgt}(A) \to^* v$ and $\mathrm{tgt}(B) \multimap v$ respectively (where $v = \mathrm{tgt}(C) = \mathrm{tgt}(D)$). In fact all three (non-trivial) SCPs of $\mathcal{R}_2$ are strongly closed, showing that $\mathcal{R}_2$ is confluent.

6.1 A Proof of Okui's Confluence Criterion

The following result is the key lemma used for Okui's confluence criterion. Our proof employs the same overall structure and concepts as Okui's original proof in [19]. However, the details differ significantly, due in part to our use of proof terms, and also because we explicitly provide substitutions and intermediate terms, where Okui only roughly sketches their existence.

Lemma 6.4. *Consider a left-linear TRS where each SCP is strongly closed. For any terms s, t and u there exists some term v such that*

$$t \leftarrow\!\!\circ\ s \to u \implies t \to^* v \leftarrow\!\!\circ\ u$$

Proof. Suppose $t \leftarrow\!\!\circ\ s \to u$. Let A be a proof term representing the multi-step $s \multimap t$ and B a proof term representing the single step $s \to u$. We prove the claim by a case distinction on $\blacktriangle(A, B)$. If $\blacktriangle(A, B) = 0$ then $A \,/\, B$ and $B \,/\, A$ are well-defined and represent the multi-steps $t \multimap \mathrm{tgt}(A \,/\, B)$ and $u \multimap \mathrm{tgt}(B \,/\, A)$ respectively. Moreover, $\mathrm{tgt}(A \,/\, B) = \mathrm{tgt}(B \,/\, A)$. If $\blacktriangle(A, B) > 0$ the idea is to use the strongly closedness assumption. Before diving into the details of the proof, we give a high level overview: First, we collect all single steps of A which have overlap with the single step B, and strip away the surrounding context. Then we perform unification between all the overlapping left-hand sides and construct co-initial proof terms A' and B' which form an SCP. The SCP (A', B') must be strongly closed, so we obtain a sequence of steps $\mathrm{tgt}(A') \to^* \cdot\ \leftarrow\!\!\circ\ \mathrm{tgt}(B')$. We embed these steps into the previously stripped away parts of A to obtain the desired $t \to^* \cdot\ \leftarrow\!\!\circ\ u$. The context $A[\ ]_{p'}$ will account for all parts of A which are parallel to, or above the single step of B. The substitution ρ will account for all parts of A which occur below the single step of B.

Below, we divide the necessary details for the proof of the case $\blacktriangle(A, B) > 0$ into 10 more digestible steps. Moreover, Figure 1 gives a visual overview of the proof.

***1. Preliminary work on proof term** A.* We collect all single steps $A_1, \ldots, A_n$ from A which have overlap with B, starting from leftmost outermost redex positions. I.e., $A_1, \ldots, A_n$ are defined such that $\{A_1, \ldots, A_n\}$ equals

$$\{\Delta(s, \alpha, p) \mid (\alpha, p) \in \mathsf{RP}(A) \wedge \blacktriangle(\Delta(s, \alpha, p), B) \neq 0\}$$

with $A_i = \Delta(s, \alpha_i, p_i)$ and $\mathsf{vpos}(\alpha_i) = (p_{i1}, \ldots, p_{in_i})$ for all $1 \leqslant i \leqslant n$, and $p_i <_{\mathsf{lex}} p_j$ for all $1 \leqslant i < j \leqslant n$. Note that $n > 0$ since $\blacktriangle(A, B) \neq 0$.

***2. Preliminary work on proof term** B.* Let $q \in \mathcal{P}\mathrm{os}(s)$ and $\beta \in \mathcal{R}$ such that

$$B = \Delta(s, \beta, q)$$

and let $(q_1, \ldots, q_m) = \mathsf{vpos}(\beta)$.

***3. Rename variables appearing in** $\alpha_1, \ldots, \alpha_n$ **and** β.* We assume there exist suitable variable renamings $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}$ and σ_β. For brevity, we will denote the renamed version of $\mathsf{lhs}(\alpha_i)$ by $\mathsf{lhs}(\alpha_i)'$, and similarly for $\mathsf{lhs}(\beta)$:

$$\mathsf{lhs}(\alpha_i)' := \mathsf{lhs}(\alpha_i)\sigma_{\alpha_i} \qquad \text{for all } 1 \leqslant i \leqslant n$$
$$\mathsf{lhs}(\beta)' := \mathsf{lhs}(\beta)\sigma_\beta$$

4. Set up the unification problem. Define

$$p := \begin{cases} q & \text{if } q < p_1 \\ p_1 & \text{otherwise} \end{cases} \tag{6}$$

$$\ell := \mathsf{lhs}(\alpha_1)'[\mathsf{lhs}(\beta)']_{q \setminus p} \tag{7}$$

$$\tau := \tau_1 \cdots \tau_n \quad \text{where} \quad \tau_i := \mathsf{tau}(\mathsf{lhs}(\alpha_i)', \ell|_{p_i \setminus p}) \tag{8}$$

Here $\mathsf{tau}(\cdot, \cdot)$ is defined in Definition 4.1. Note that p denotes the root position of our SCP. Hence, the definition of ℓ and the equations for the unification problem use positions relative to p (i.e., $q \setminus p$ and $p_i \setminus p$). To establish well-definedness of the definitions above, we need to prove that

$$p \leqslant q \quad \text{and} \quad p \leqslant p_i \quad \text{for all } 1 \leqslant i \leqslant n \tag{9}$$

We first prove $p \leqslant q$ and $p \leqslant p_1$ by considering the two cases for p:

1. If $q \leqslant p_1$ then by definition $p = q$. Hence $p = q \leqslant q$ and $p = q \leqslant p_1$.
2. Otherwise, we have $p = p_1$. We know that $p_1 \not\parallel q$ since the redexes at p_1 and q overlap, which could not be the case if p_1 and q were parallel positions. Hence, $p = p_1 < q$.

If $1 < i \leqslant n$ then $p_1 <_{\mathsf{lex}} p_i$. If $p_1 < p_i$, we are immediately done. Otherwise, we have $p_1 \parallel p_i$. As before, we use the fact that the redexes at p_1 and p_i overlap with the redex at q and infer $p_1 \not\parallel q$ and $p_i \not\parallel q$. Together with $p_1 \parallel p_i$ it follows that $q < p_1$ and $q < p_i$ which implies $p = q \leqslant p_i$.

Furthermore, since all $\mathsf{lhs}(\alpha)'$ are linear and do not have common variables, we can prove that $\ell\tau$ is also linear.

Ultimately we want to establish τ as an mgu of

$$E = \{\mathsf{lhs}(\alpha_1)' \approx \ell|_{p_1 \setminus p}, \ldots, \mathsf{lhs}(\alpha_n)' \approx \ell|_{p_n \setminus p}\}$$

In order to do that, we first need to verify that the set is indeed unifiable. The underlying idea is, that this is possible since $s = \mathrm{src}(B) = \mathrm{src}(A) = \mathrm{src}(A_1) = \cdots = \mathrm{src}(A_n)$. This means that all terms in the set of equations above match subterms of s. In the next step we introduce an actual witness σ for unifiability, which is basically the combination of all these matching substitutions.

***5. Define substitution** σ **mapping from left-hand sides to** s.* Let $\mathrm{var}(\mathsf{lhs}(\beta)') = (y_1, \ldots, y_m)$ and $\mathrm{var}(\mathsf{lhs}(\alpha_i)') = (x_{i1}, \ldots, x_{in_i})$ for all $1 \leqslant i \leqslant n$. Define

$$\sigma := \{x_{ij} \mapsto s|_{p_i p_{ij}}\} \cup \{y_j \mapsto s|_{q q_j}\}$$

and show

$$\mathsf{lhs}(\alpha_i)'\sigma = s|_{p_i} \qquad \text{for all } 1 \leqslant i \leqslant n \tag{10}$$

$$\mathsf{lhs}(\beta)'\sigma = s|_q \tag{11}$$

Similar properties were also established in [11]. The proofs are relatively straightforward by using $A_i = \Delta(s, \alpha_i, p_i) = s[\alpha_i(s|_{p_i p_{i,1}}, \ldots, s|_{p_i p_{i,n}})]_{p_i}$ and $\mathrm{src}(A_i) = s$ to prove (10). For proving (11) we use $B = \Delta(s, \beta, q) = s[\beta(s|_{q1}, \ldots, s|_{qm})]$ and $\mathrm{src}(B) = s$. From (10) and (11) it follows that σ is a unifier for the set E. Moreover, applying Lemma 4.2 together with (10) to each variable of $\mathsf{lhs}(\beta)'$ yields $\mathsf{lhs}(\beta)'\tau\sigma = s|_q$. Similarly, applying Lemma 4.2 together with (11) to each variable of $\mathsf{lhs}(\alpha_i)'$ yields $\mathsf{lhs}(\alpha_i)'\tau\sigma = s|_{p_i}$. Finally, we obtain

$$\ell\tau\sigma = s|_p \tag{12}$$

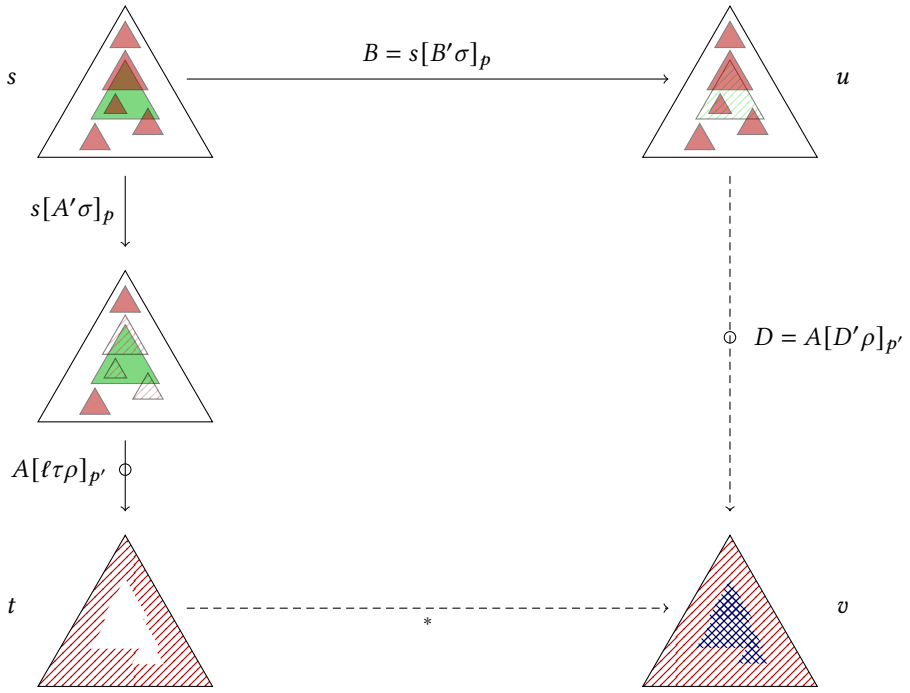

Figure 1. Picture for the case $\blacktriangle(A, B) > 0$ in the proof of Lemma 6.4. Redex patterns of A are marked in red. The single redex pattern of B is marked in green. The overlapping triangles form the SCP (A', B'). Contracted redex patterns are marked by shaded triangles. The terms t and v contain redex patterns contracted by $A[\ ]_{p'}$ and ρ (which were originally part of A). The term v also contains redex patterns contracted by the closing sequence for the SCP (A', B'). These are marked in blue. Note that the blue pattern (and the corresponding white pattern in t) could also be duplicated by $A[\ ]_{p'}$, but we did not model this in the picture.

6. Show that τ is an mgu of E. The goal is to apply Lemma 4.4 for the set of equations E. Note that $\ell|_{p_1\setminus p}$ may contain variables of $\mathsf{lhs}(\alpha_1)'$ if $p = p_1$ and thereby violating condition 3. We choose to circumvent this issue by considering the altered set of equations E' below.

$$E' = \{s_1 \approx \mathsf{lhs}(\beta)'|_{p'_1}, \ldots, s_n \approx \mathsf{lhs}(\beta)'|_{p'_n}\}$$

where $s_1 = \mathsf{lhs}(\alpha_1)'|_{q\setminus p}$, $p'_1 = p_1\setminus p$ and

$$s_i = \mathsf{lhs}(\alpha_i)' \qquad\qquad p'_i = p_i\setminus q$$

for $i \in \{2, \ldots, n\}$. Note that either $p'_1 = p_1\setminus p = \epsilon$ or $q\setminus p = \epsilon$. Hence

$$\ell|_{p_1\setminus p} = \mathsf{lhs}(\alpha_1)'[\mathsf{lhs}(\beta)'|_{p'_1}]_{q\setminus p}$$

Moreover, $\ell|_{p_i\setminus p} = \mathsf{lhs}(\beta)'|_{p'_i}$ for all $i \in \{2, \ldots, n\}$ since $q \leqslant p_i$ for $i \in \{2, \ldots, n\}$ and therefore $p_i\setminus p = (q\setminus p)p'_i$. These equalities guarantee that E' and E have the same set of mgus. Moreover, the four conditions of Lemma 4.4 hold for E':

1. In the previous step we showed that E (and therefore E') is unifiable using σ.
2. Since $\mathcal{R}$ is assumed to be left-linear, we know that $\mathsf{lhs}(\alpha_1), \ldots, \mathsf{lhs}(\alpha_n)$, $\mathsf{lhs}(\beta)$ are linear terms.
3. Since the variables of $\mathsf{lhs}(\alpha_i)'$ and $\mathsf{lhs}(\beta)'$ were deliberately renamed apart in Step 3 we have $\mathcal{V}\mathrm{ar}(s_i) \cap \mathcal{V}\mathrm{ar}(\mathsf{lhs}(\beta)') = \varnothing$.
4. The first part of this condition is easy to establish. Assume $1 \leqslant i < j \leqslant n$ and let $\tau_i = \tau(s_i, \mathsf{lhs}(\beta)'|_{p'_i})$. Then due to the renamings we have $s_j\tau_i = s_j$. Showing that $\mathsf{lhs}(\beta)'|_{p'_j}$ is not affected by τ_i is more tricky. The proof works by contradiction, assuming that there exists a variable $y \in \mathrm{dom}(\tau_i)$ and position r such that $p'_j \leqslant r$ and $\mathsf{lhs}(\beta)|_r = y$. From the definition of $\tau(\cdot,\cdot)$ we know that τ_i must contain a binding $y \mapsto s_i|_{r'}$ for some $r' \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(s_i)$ with $r = p'_i r'$. Let us consider the case $i > 1$. Since the positions $p_1, \ldots, p_n$ are a subset of the redex positions of A, ordered such that leftmost outermost redexes appear first, we know that all positions in $\mathsf{lhs}(\alpha_j)$ must either be parallel to, or below any function position of $\mathsf{lhs}(\alpha_i) = s_i$. So $p_i r' < p_j$ and therefore $p'_i r' < p'_j$. Now a contradiction can be derived from the following chain of inequalities:

$$r = p'_i r' < p'_j \leqslant r$$

The case $i = 1$ essentially works the same but requires some additional fiddling with positions.

So Lemma 4.4 is applicable to E' and yields τ as mgu for E' and E.

Since τ is an mgu of (subterms of) the left-hand sides $\mathsf{lhs}(\alpha_i)'$ with subterms of $\mathsf{lhs}(\beta)'$, each function symbol that appears in the term $\ell\tau$ either belongs to one of the $\mathsf{lhs}(\alpha_i)$ or to $\mathsf{lhs}(\beta)$. This allows us to use the following result in Step 8:

$$r \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell\tau) \implies pr \in \mathcal{P}\mathrm{os}_L(A_1) \cup \cdots \cup \mathcal{P}\mathrm{os}_L(A_n) \cup \mathcal{P}\mathrm{os}_L(B) \tag{13}$$

7. Obtain the two proof terms making up the SCP. Define

$$A'_i := \ell\tau[\alpha_i(\sigma_{\alpha_i}\tau)]_{p_i\backslash p} \qquad A' := \bigsqcup_{i=1}^{n} A'_i$$
$$B' := \ell\tau[\beta(\sigma_\beta\tau)]_{q\backslash p}$$

From the properties of τ and σ it follows that

$$\mathrm{src}(A'_i) = \ell\tau \text{ and } A_i = s[A'_i\sigma]_p \quad \text{for all } 1 \leqslant i \leqslant n \tag{14}$$
$$\mathrm{src}(B') = \ell\tau \text{ and } B = s[B'\sigma]_p \tag{15}$$

We want to show that A' is a well-defined proof term using Lemma 3.8. Therefore, we need to show that $A'_i \sqcup A'_j$ is defined for all $A'_i, A'_j \in \{A'_1, \ldots, A'_n\}$. From Lemma 3.12 we already know that $A_i \sqcup A_j$ is defined. Then the desired property follows directly from (14) and Lemma 3.6. Hence, Lemma 3.8 yields

$$A' \text{ is well-defined} \quad \text{and} \quad \mathrm{src}(A') = \ell\tau \tag{16}$$

Moreover, from the definition of A' and Lemma 3.10 it follows that

$$\mathsf{RP}(A') = \bigsqcup_{i=1}^{n} \mathsf{RP}(A'_i) = \{(\alpha_1, p_1\backslash p), \ldots, (\alpha_n, p_n\backslash p)\} \tag{17}$$

8. Transform A' into A using a substitution ρ. From the definition of p we know that either $p = p_1$ or $p = q$. In the first case we have $\ell(\mathrm{src}^\sharp(A)|_p) = \alpha^0$. In the second case $q < p_1$ and $\mathrm{src}^\sharp(A)|_p$ cannot be labeled, since p_1 is by definition the smallest position where a redex in A overlaps with B. Thus, Lemma 3.15 can be applied to obtain a position p' such that

$$\mathrm{src}(A[\,]_{p'}) = s[\,]_p \quad \text{and} \quad \mathrm{src}(A|_{p'}) = s|_p \tag{18}$$
$$\mathrm{src}^\sharp(A) = \mathrm{src}^\sharp(A)[\mathrm{src}^\sharp(A|_{p'})]_p \tag{19}$$

Define

$$\rho := \{x \mapsto A|_{p'r} \mid x \in \mathcal{V}\mathrm{ar}(A') \wedge A'|_r = x\}$$

Note that A' is a linear term according to Lemma 3.2, since $\ell\tau$ is a linear term (see Step 4) and $\mathrm{src}(A') = \ell\tau$. Hence, the substitution ρ is well-defined. We show that the following key property holds for ρ:

$$A'\rho = A|_{p'} \tag{20}$$

We present a rather detailed account of the formalized proof of (20), since it shows why some of the intermediate results we have seen so far are indeed necessary. This proof was also one of the bigger challenges for the formalization, and this illustrates why.

We already established linearity of A' above. We additionally prove the remaining assumptions needed for Lemma 3.19.

1. Using (16) and (12) we obtain
$$\mathrm{src}(A')\sigma = \ell\tau\sigma = s|_p = \mathrm{src}(A|_{p'})$$
2. Assume $(\alpha, r) \in \mathsf{RP}(A')$. From the definition of A' we know that there exists an i such that $1 \leqslant i \leqslant n$ and $(\alpha, r) \in \mathsf{RP}(A'_i)$. From the definition of A'_i we know that $\mathsf{RP}(A'_i) = \{(\alpha_i, p_i\backslash p)\}$ and hence, $(\alpha, r) = (\alpha_i, p_i\backslash p)$. Moreover, recall that $(\alpha_i, p_i) \in \mathsf{RP}(A)$. From (19), Lemma 3.14, and (9) from Step 4 it follows that $\ell(\mathrm{src}^\sharp(A|_{p'})(p_i\backslash p)) = \ell(\mathrm{src}^\sharp(A)(p_i)) = \alpha_i^0$. By another application of Lemma 3.14 we conclude that $(\alpha_i, p_i\backslash p) = (\alpha, r) \in \mathsf{RP}(A|_{p'})$.
3. Assume $(\alpha, r) \in \mathsf{RP}(A|_{p'})$ and $r \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\mathrm{src}(A'))$. Like in the previous item, we apply (19) and Lemma 3.14 to obtain $(\alpha, pr) \in \mathsf{RP}(A)$. From $r \in \mathcal{P}\mathrm{os}_{\mathcal{F}}(\mathrm{src}(A')) = \mathcal{P}\mathrm{os}_{\mathcal{F}}(\ell\tau)$ and (13), we know that either $pr \in \mathcal{P}\mathrm{os}_L(B)$ or $pr \in \mathcal{P}\mathrm{os}_L(A_i)$ for some $i \in \{1, \ldots, n\}$. Hence, $\blacktriangle(\Delta(s, pr, \alpha), B) \neq 0$ and therefore $\Delta(s, pr, \alpha) = A_i$ for some $1 \leqslant i \leqslant n$. Then $A'_i = \ell\tau[\alpha_i(\sigma_{\alpha_i}\tau)]_{p_i\backslash p}$ by definition. Finally, we conclude $(\alpha, r) \in \mathsf{RP}(A'_i)$ and thus $(\alpha, r) \in \mathsf{RP}(A') = \bigcup_{i=1}^{n} \mathsf{RP}(A'_i)$.

Note that (20) immediately implies

$$A = A[A'\rho]_{p'} \tag{21}$$

Moreover, we obtain

$$\forall x \in \mathcal{V}\mathrm{ar}(\ell\tau) \colon \mathrm{src}(\rho(x)) = \sigma(x) \tag{22}$$

since $\mathcal{V}\mathrm{ar}(A') = \mathcal{V}\mathrm{ar}(\mathrm{src}(A')) = \mathcal{V}\mathrm{ar}(\ell\tau)$ (using Lemma 3.2 and (16)) and

$$\mathrm{src}(A'\rho) \overset{(20)}{=} \mathrm{src}(A|_{p'}) \overset{(18)}{=} s|_p \overset{(12)}{=} \ell\tau\sigma \overset{(16)}{=} \mathrm{src}(A')\sigma$$

9. Prove that (A', B') is an SCP. We go through the nine defining properties of an SCP:

1. $\mathsf{RP}(A') = \{(\alpha_1, p_1\backslash p), \ldots, (\alpha_n, p_n\backslash p)\}$ and $n > 0$ with $p_i\backslash p <_{\mathsf{lex}} p_j\backslash p$ for all $1 \leqslant i < j \leqslant n$ follows from (17) and the order on $p_1, \ldots, p_n$ defined in Step 1.
2. $\mathsf{RP}(B') = \{(\beta, q\backslash p)\}$ follows from the definition of B'.
3. The renamings $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}$ and σ_β defined in Step 3 fulfill the conditions.
4. Let $i \in \{1, \ldots, n\}$. Then we have $A_i = s[A'_i\sigma]_p$ from (14) and $B = s[B'\sigma]_p$ from (15). Moreover, from Step 1 we know that $\blacktriangle(A_i, B) \neq 0$. It follows that $\blacktriangle(A'_i, B') \neq 0$. From (17) we know that $\{\Delta(\mathrm{src}(A'), \alpha, p) \mid (\alpha, p) \in \mathsf{RP}(A')\} = \{A'_1, \ldots, A'_n\}$ and thus
$$\mathsf{OV}(A', B') = \bigsqcup_{i=1}^{n} A'_i = A'$$
5. $q\backslash p = \epsilon$ or $p_1\backslash p = \epsilon$ follows from the definition of p in Step 4.

6. $\ell = \mathsf{lhs}(\alpha_1)'[\mathsf{lhs}(\beta)']_{q\setminus p}$ by definition.
7. The set of equations
$$\{\mathsf{lhs}(\alpha_1)' \approx \ell|_{p_1\setminus p}, \ldots, \mathsf{lhs}(\alpha_n)' \approx \ell|_{p_n\setminus p}\}$$
is unifiable with mgu τ, as was shown in Step 6.
8. $A' = \bigsqcup_{i=1}^{n} A'_i$ follows directly from (16) in the previous step.
9. $B' = \ell\tau[\beta(\sigma_\beta\tau)]_{q\setminus p}$ by definition.

10. Apply the strongly closedness assumption. By assumption every SCP is strongly closed. Therefore, we obtain a term v' such that
$$\mathsf{tgt}(A') \to^* v' \mathrel{\leftarrow\!\!\!\circ} \mathsf{tgt}(B') \tag{23}$$
Let D' be the multi-step representing $\mathsf{tgt}(B') \mathrel{-\!\!\circ\!\!\to} v'$ and define $v := \mathsf{tgt}(A[v'\rho]_{p'})$ and $D := A[D'\rho]_{p'}$. We show
$$t \to^* v \tag{24}$$
$$u \mathrel{-\!\!\circ\!\!\to} v \quad \text{(witnessed by } D\text{)} \tag{25}$$
Proof of (24):
$$\begin{aligned} t = \mathsf{tgt}(A) &= \mathsf{tgt}(A[A'\rho]_{p'}) && (21)\\ &= \mathsf{tgt}(A[\mathsf{tgt}(A')\rho]_{p'}) && (\text{Lemma 3.3})\\ &\to^* \mathsf{tgt}(A[v'\rho]_{p'}) = v && (\text{Lemma 3.3 and 23}) \end{aligned}$$
Proof of (25):
$$\begin{aligned} \mathsf{src}(D) &= \mathsf{src}(A)[\mathsf{src}(D')(\mathsf{src}\circ\rho)]_p && (\text{Lemma 3.3})\\ &= s[\mathsf{tgt}(B')(\mathsf{src}\circ\rho)]_p && (\text{definition of } D')\\ &= s[\mathsf{tgt}(B')\sigma]_p && (22)\\ &= \mathsf{tgt}(B) = u && (\text{Lemma 3.3 and 15})\\ \mathsf{tgt}(D) &= \mathsf{tgt}(A[\mathsf{tgt}(D')\rho]_{p'}) && (\text{Lemma 3.3})\\ &= \mathsf{tgt}(A[v'\rho]_{p'}) = v && (\text{definition of } D') \end{aligned}$$
Note that for the application of (22) we need $\mathcal{V}\mathrm{ar}(\mathsf{tgt}(B')) \subseteq \mathcal{V}\mathrm{ar}(\ell\tau)$. This follows from (15) and Lemma 3.2. □

Using Lemma 6.4, it is now easy to prove Theorem 6.2.

Proof of Theorem 6.2. By Lemma 2.1 it suffices to show that $\mathrel{-\!\!\circ\!\!\to}$ is strongly confluent. Assume $t \mathrel{\leftarrow\!\!\circ} s \mathrel{-\!\!\circ\!\!\to} u$. Hence, $s \to^n u$ for some $n \geqslant 0$. We show the existence of a term v such that $t \mathrel{-\!\!\circ\!\!\to}^* v \mathrel{\leftarrow\!\!\circ} u$ by induction on n. For $n = 0$ the proof is trivial by taking $u = v$. For $n > 0$ we obtain an intermediate term u' such that $s \to u' \to^{n-1} u$ and using Lemma 6.4 we obtain a term v' such that $t \mathrel{-\!\!\circ\!\!\to}^* v' \mathrel{\leftarrow\!\!\circ} u'$. Since we have $v' \mathrel{\leftarrow\!\!\circ} u' \to^{n-1} u$, we can apply the induction hypothesis to obtain the desired term v' such that $v' \mathrel{-\!\!\circ\!\!\to}^* v \mathrel{\leftarrow\!\!\circ} u$. See also Figure 2.

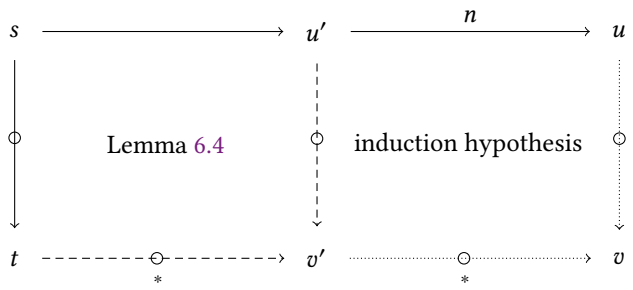

Figure 2. Proof of Theorem 6.2.

```
locale tau =
 fixes ss :: "('f, 'v) term list"
   and t :: "('f, 'v) term"
   and ps :: "pos list"
 assumes "length ss = length ps"
   and "linear_term t"
   and "∀s ∈ set ss. linear_term s"
   and "∀p ∈ set ps. p ∈ poss t"
   and "∀s ∈ set ss. vars_term s ∩ vars_term t = {}"
   and "∀i j. i < j ∧ j < length ss ⟶
        vars_term (ss!i) ∩ vars_term (ss!j) = {}"
```

Listing 2. The locale `tau` used for the proof of Theorem 6.2.

6.2 Formalization Details

As mentioned in Section 2.2, we relied on Isabelle's locales to modularize the main proof with its many intermediate facts. For instance, the definition and properties of the substitution τ of step 4 do not directly depend on the proof terms A and B. Instead, it only matters whether the terms with which $\mathsf{tau}(\cdot,\cdot)$ is instantiated satisfy specific properties. So we encapsulated the properties of τ into a locale called `tau` (see Listing 2). This locale is then instantiated appropriately during the main proof, as described in step 6 of the proof outline above. Moreover, in steps 1 and 2 of the outlined proof, important properties of A and B are established. The locale `overlapping_part` describes a state were these properties are assumed to hold. Within this locale, we define several key elements, including:

- the renamed left-hand sides from step 3,
- the position p, term ℓ and substitution τ of step 4,
- the substitution σ of step 5,
- the proof terms $A'_1, \ldots, A'_n$ and B' of step 7.

The proof term A' of step 7 gets its own subcontext within `overlapping_part`. In this subcontext, we assume that A' is a well-defined proof term and is the result of $\bigsqcup_{i=1}^{n} A'_i$. The existence of the position p' in step 8 is handled in an additional nested context within `overlapping_part`. Figure 3 illustrates the different nested contexts. The main theorem `okui_imp_CR` is stated outside of any locale or context.

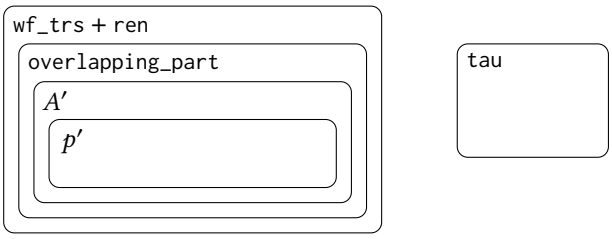

Figure 3. Sketch of nested locales and contexts used in the proof of Theorem 6.2.

```
theorem okui_imp_CR:
  assumes "left_lin_wf_trs R"
    and "⋀A B. (A, B) ∈ ren.sim_cp ren R ⟹ ∃v.
 (target A, v) ∈ (rstep R)* ∧ (target B, v) ∈ mstep R"
  shows "CR (rstep R)"
```

Listing 3. Statement of the main theorem in Isabelle.

As can bee seen in Listing 3, it has only two assumptions:

1. The TRS $\mathcal{R}$ in question adheres to the variable conditions and is left-linear.
2. All simultaneous critical pairs of $\mathcal{R}$ (with respect to a certain renaming function) are strongly closed.

Under these assumptions, the theorem concludes that $\mathcal{R}$ is confluent.

6.3 Okui Subsumes Development-Closedness

Since we did not yet formalize an algorithm for computing all SCPs of a given TRS, we cannot yet execute code to verify confluence proofs based on SCPs directly. Verification of such an algorithm will indeed be another hard task. In the meantime, one can of course question whether the assumptions of the main theorem can be fulfilled in practice. In order to address such concerns we formalized Okui's remark, stating that his criterion subsumes development closedness [19]. I.e., we showed that any development closed TRS indeed has strongly closed SCPs. Since the development closedness criterion formalized in IsaFoR has successfully been applied by CeTA to many TRSs of the ARI-COPS database,[7] each of these TRSs is an example of a TRS fulfilling the conditions formalized in Okui's criterion.

7 Related Work

Several confluence criteria have been formalized in different proof assistants. The most comprehensive collection is contained in the IsaFoR library for Isabelle/HOL. This library includes notable criteria, such as Knuth and Bendix's joinability of critical pairs for terminating TRSs [23], strongly closed critical pairs for linear TRSs [16], and left-linearity with development closed critical pairs [11, 12] (which subsumes the parallel closedness criterion previously formalized in [16]). Most recently, confluence via parallel critical pairs has also been formalized in IsaFoR [9]. Additionally, the landmark result by Knuth and Bendix has been formalized in ACL2 [21] and in PVS [7]. Orthogonality as a sufficient criterion for confluence has also been formalized in PVS [20]. To our knowledge, ours is the first formalization of a confluence result based on simultaneous critical pairs.

Okui noted in [19] that his confluence result subsumes van Oostrom's development closedness, so our result subsumes the confluence results in [11, 12]. However, these results have been lifted to commutation of two TRSs. Although it should be straightforward to extend Okui's simultaneous critical pair criterion to commutation as well, this has not yet been formalized. Notably, confluence via simultaneous critical pairs is incomparable to the parallel critical pair criteria in [9].

8 Conclusion and Future Work

Building on recent successful formalizations of critical pair criteria for confluence, we were able to obtain a formalized proof of Okui's simultaneous critical pair criterion for left-linear TRSs. To achieve this result we have reformulated the definition of simultaneous critical pairs using proof terms. While the structure of Okui's proof in [19] could be essentially followed, the formalization process presented several challenges.

This result is now part of the IsaFoR library, with over 6000 lines of Isabelle code added, including several new results on proof terms as well as unification of linear terms.

Currently, an algorithm for computing SCPs has not yet been verified in IsaFoR. This is a non-trivial task, but as soon as such a formalized procedure is in place, it will be possible to certify confluence proofs based on SCPs with the certifier CeTA [17]. The plan is to adopt a similar certificate format as for development closedness, requiring only the maximum number of steps for the closing sequences to be specified. Preliminary experiments with ACP and CSI on the ARI-COPS database confirm that this will further reduce the gap between the number of YES/NO answers by confluence tools and the number of certified YES/NO answers.

As mentioned in the previous section, it is anticipated that Theorem 6.2 can be extended to commutation of two TRSs. A formalization of this extension remains as future work.

Acknowledgments

This research was funded in part by the Austrian Science Fund (FWF) project I5943 and by a grant of the University of Innsbruck awarded to Christina Kirk. The pertinent comments by the anonymous reviewers improved the presentation.

[7] https://ari-cops.uibk.ac.at/

References

[1] Takahito Aoto, Junichi Yoshida, and Yoshihito Toyama. 2009. Proving Confluence of Term Rewriting Systems Automatically. In *Proc. 20th International Conference on Rewriting Techniques and Applications (LNCS, Vol. 5595)*, Ralf Treinen (Ed.). 93–102. https://doi.org/10.1007/978-3-642-02348-4_7

[2] Franz Baader and Tobias Nipkow. 1998. *Term Rewriting and All That.* Cambridge University Press. https://doi.org/10.1017/CBO9781139172752

[3] Clemens Ballarin. 2003. Locales and Locale Expressions in Isabelle/Isar. In *Proc. 2003 International Workshop on Types for Proofs and Programs (LNCS, Vol. 3085)*, Stefano Berardi, Mario Coppo, and Ferruccio Damiani (Eds.). 34–50. https://doi.org/10.1007/978-3-540-24849-1_3

[4] Jasmin Christian Blanchette, Sascha Böhme, and Lawrence C. Paulson. 2013. Extending Sledgehammer with SMT solvers. *Journal of Automated Reasoning* 51, 1 (2013), 109–128. https://doi.org/10.1007/s10817-013-9278-5

[5] Sascha Böhme and Tobias Nipkow. 2010. Sledgehammer: Judgement Day. In *Proc. 5th International Joint Conference on Automated Reasoning (LNAI, Vol. 6173)*, Jürgen Giesl and Reiner Hähnle (Eds.). 107–121. https://doi.org/10.1007/978-3-642-14203-1_9

[6] Bertram Felgenhauer. 2015. Labeling Multi-Steps for Confluence of Left-Linear Term Rewrite Systems. In *Proc. 4th International Workshop on Confluence*, Ashish Tiwari and Takahito Aoto (Eds.). 33–37. Available from http://cl-informatik.uibk.ac.at/iwc/iwc2015.pdf.

[7] André Luiz Galdino and Mauricio Ayala-Rincón. 2010. A Formalization of the Knuth-Bendix(-Huet) Critical Pair Theorem. *Journal of Automated Reasoning* 45, 3 (2010), 301–325. https://doi.org/10.1007/s10817-010-9165-2

[8] Bernhard Gramlich. 1996. Confluence without Termination via Parallel Critical Pairs. In *Proc. 21st International Colloquium on Trees in Algebra and Programming (LNCS, Vol. 1059)*, Hélène Kirchner (Ed.). 211–225. https://doi.org/10.1007/3-540-61064-2_39

[9] Nao Hirokawa, Dohan Kim, Kiraku Shintani, and René Thiemann. 2024. Certification of Confluence- and Commutation-Proofs via Parallel Critical Pairs. In *Proc. 13th International Conference on Certified Programs and Proofs*, Amin Timany, Dmitriy Traytel, Brigitte Pientka, and Sandrine Blazy (Eds.). 147–161. https://doi.org/10.1145/3636501.3636949

[10] Christina Kohl and Aart Middeldorp. 2018. ProTeM: A Proof Term Manipulator (System Description). In *Proc. 3rd International Conference on Formal Structures for Computation and Deduction (LIPIcs, Vol. 108)*, Hélène Kirchner (Ed.). 31:1–31:8. https://doi.org/10.4230/LIPIcs.FSCD.2018.31

[11] Christina Kohl and Aart Middeldorp. 2023. A Formalization of the Development Closedness Criterion for Left-Linear Term Rewrite Systems. In *Proc. 12th International Conference on Certified Programs and Proofs*, Robbert Krebbers, Dmitriy Traytel, Brigitte Pientka, and Steve Zdancewic (Eds.). 197–210. https://doi.org/10.1145/3573105.3575667

[12] Christina Kohl and Aart Middeldorp. 2023. Formalizing Almost Development Closed Critical Pairs (Short Paper). In *Proc. 14th International Joint Conference on Automated Reasoning (LIPIcs, Vol. 268)*, Adam Naumowicz and René Thiemann (Eds.). 38:1–38:8. https://doi.org/10.4230/LIPIcs.ITP.2023.38

[13] Julian Nagele. 2017. *Mechanizing Confluence.* Ph. D. Dissertation. University of Innsbruck.

[14] Julian Nagele, Bertram Felgenhauer, and Aart Middeldorp. 2015. Improving Automatic Confluence Analysis of Rewrite Systems by Redundant Rules. In *Proc. 26th International Conference on Rewriting Techniques and Applications (LIPIcs, Vol. 36)*, Maribel Fernández (Ed.). 257–268. https://doi.org/10.4230/LIPIcs.RTA.2015.257

[15] Julian Nagele, Bertram Felgenhauer, and Aart Middeldorp. 2017. CSI: New Evidence — A Progress Report. In *Proc. 26th International Conference on Automated Deduction (LNAI, Vol. 10395)*, Leonardo de Moura (Ed.). 385–397. https://doi.org/10.1007/978-3-319-63046-5_24

[16] Julian Nagele and Aart Middeldorp. 2016. Certification of Classical Confluence Results for Left-Linear Term Rewrite Systems. In *Proc. 7th International Joint Conference on Automated Reasoning (LNCS, Vol. 9807)*, Jasmin Christian Blanchette and Stephan Merz (Eds.). 290–306. https://doi.org/10.1007/978-3-319-43144-4_18

[17] Julian Nagele and René Thiemann. 2014. Certification of Confluence Proofs using CeTA. In *Proc. 3rd International Workshop on Confluence*, Takahito Aoto and Delia Kesner (Eds.). 19–23. Available from http://cl-informatik.uibk.ac.at/iwc/iwc2014.pdf.

[18] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. 2002. *Isabelle/HOL – A Proof Assistant for Higher-Order Logic.* LNCS, Vol. 2283. Springer. https://doi.org/10.1007/3-540-45949-9

[19] Satoshi Okui. 1998. Simultaneous Critical Pairs and Church–Rosser Property. In *Proc. 9th International Conference on Rewriting Techniques and Applications (LNCS, Vol. 1379)*, Tobias Nipkow (Ed.). 2–16. https://doi.org/10.1007/BFb0052357

[20] Ana Cristina Rocha-Oliveira, André Luiz Galdino, and Mauricio Ayala-Rincón. 2017. Confluence of Orthogonal Term Rewriting Systems in the Prototype Verification System. *Journal of Automated Reasoning* 58, 2 (2017), 231–251. https://doi.org/10.1007/s10817-016-9376-2

[21] José-Luis Ruiz-Reina, José-Antonio Alonso, María-José Hidalgo, and Francisco-Jesús Martín-Mateos. 2002. Formal Proofs About Rewriting Using ACL2. *Annals of Mathematics and Artificial Intelligence* 36, 3 (2002), 239–262. https://doi.org/10.1023/A:1016003314081

[22] Kiraku Shintani and Nao Hirokawa. 2024. Compositional Confluence Criteria. *Logical Methods in Computer Science* 20, 1 (2024). https://doi.org/10.46298/lmcs-20(1:6)2024

[23] Christian Sternagel and René Thiemann. 2013. Formalizing Knuth–Bendix Orders and Knuth–Bendix Completion. In *Proc. 24th International Conference on Rewriting Techniques and Applications (LIPIcs, Vol. 21)*, Femke van Raamsdonk (Ed.). 287–302. https://doi.org/10.4230/LIPIcs.RTA.2013.287

[24] TeReSe (Ed.). 2003. *Term Rewriting Systems.* Cambridge Tracts in Theoretical Computer Science, Vol. 55. Cambridge University Press.

[25] Yoshihito Toyama. 1981. *On the Church–Rosser Property of Term Rewriting Systems.* Technical Report 17672. NTT ECL Technical Report.

[26] Vincent van Oostrom. 1995. Development Closed Critical Pairs. In *Proc. 2nd International Workshop on Higher-Order Algebra, Logic, and Term Rewriting (LNCS, Vol. 1074)*, Gilles Dowek, Jan Heering, Karl Meinke, and Bernhard Möller (Eds.). 185–200. https://doi.org/10.1007/3-540-61254-8_26

[27] Vincent van Oostrom. 1997. Developing Developments. *Theoretical Computer Science* 175, 1 (1997), 159–181. https://doi.org/10.1016/S0304-3975(96)00173-9

[28] Vincent van Oostrom and Roel de Vrijer. 2002. Four Equivalent Equivalences of Reductions. In *Proc. 2nd International Workshop on Reduction Strategies in Rewriting and Programming (Electronic Notes in Theoretical Computer Science, Vol. 70(6))*, Bernhard Gramlich and Salvador Lucas (Eds.). 21–61. https://doi.org/10.1016/S1571-0661(04)80599-1

Received 2024-09-06; accepted 2024-11-19