Verifying and Generalizing Simultaneous Critical Pairs

René Thiemann

University of Innsbruck, Austria
rene.thiemann@uibk.ac.at

Abstract

Okui proved that simultaneous critical pairs (SCPs) can be used as a sufficient criterion to ensure confluence of term rewrite systems. His definitions, lemmas and proofs where reformulated by Kirk and Middeldorp. They heavily utilize proof terms and finally arrive at a formalized proof of Okui's result in Isabelle/HOL. However, Kirk and Middeldorp's formalization lacks an executable algorithm to compute the set of proof term based SCPs in a verified way.

In this work, we provide such an algorithm, and we further modify the formalization in such a way, that SCPs can also be used to show commutation, generalizing the confluence result. Our results have fully been integrated in the certifier CeTA, that now can deal with both commutation- and confluence-proofs by SCPs.

1 Introduction

Confluence is an important and well-studied property of term rewrite systems (TRSs). Since confluence is an undecidable property, several sufficient criteria have been developed, often using (variants of) critical pairs. For instance, there is van Oostrom's criterion of development closedness [9] using critical pairs, there is Toyama's criterion [8] based on parallel critical pairs, and there is Okui's criterion [6] based on simultaneous critical pairs (SCPs).

In previous works the mentioned confluence criteria have been formally verified in Isabelle/HOL [5]: Kirk and Middeldorp handled development closedness and Okui's criterion [4, 3], and Hirokawa et al. verified Toyama's criterion [2]. Here, during the formalization some of the criteria have been generalized from confluence to commutation. In this work, we extend this line of research in two directions.

- We generalize Okui's criterion from confluence to commutation, based on the formalization of Kirk and Middeldorp.
- We develop a verified implementation of SCPs, so that Okui's criterion can be checked by our certifier CeTA [7].

Regarding the second contribution, note that Kirk and Middeldorp provide a definition of SCPs that uses unbounded existential quantifiers, which makes an implementation challenging.

2 Preliminaries

We assume familiarity with term rewriting [1]. Given a binary relation $\to$, we write $\to^*$ for its reflexive closure, $\to^=$ for its reflexive closure, and $\leftarrow$ for its inverse. Given two binary relations $\to_1$ and $\to_2$, $\to_1 \circ \to_2$ denotes the relation composition of $\to_1$ and $\to_2$. Relations $\to_1$ and $\to_2$ commute if ${}^*_1\!\leftarrow \circ \to_2^* \subseteq \to_2^* \circ {}^*_1\!\leftarrow$, and they strongly commute if ${}_1\!\leftarrow \circ \to_2 \subseteq \to_2^= \circ {}^*_1\!\leftarrow$. Strong commutation implies commutation. Confluence of $\to$ is commutation where $\to_1 = \to_2 = \to$, and similarly strong confluence is strong commutation of two identical relations.

First order terms $s, t, \ell, r, \ldots \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ are build over some infinite set of variables $x, y, z, \ldots \in \mathcal{V}$ and some set of function symbols $a, b, f, g, \ldots \in \mathcal{F}$, where each $f \in \mathcal{F}$ has a fixed arity. We assume that $\mathcal{F}$ and $\mathcal{V}$ are arbitrary but fixed within this paper. A substitution σ is a function of type $\mathcal{V} \to \mathcal{T}(\mathcal{F}, \mathcal{V})$, and we write $t\sigma$ for the term that is obtained by replacing each variable x in term t by $\sigma(x)$. A context C is a term with exactly one hole, and $C[t]$ is the term where this hole is replaced by t. Let $\mathcal{V}ars(t)$ denote the set of variables of term t. Given a term t and a position p within this term, $t|_p$ denotes the subterm of t at position p and $t[s]_p$ replaces the subterm at position p by s. For every term t, we use the notation $FPos(t)$ for the set of positions p of t that satisfy $t|_p \notin \mathcal{V}$.

A term rewrite system (TRS) $\mathcal{R}$ is a set of rules $\ell \to r$ where $\ell, r \in \mathcal{T}(\mathcal{F}, \mathcal{V})$, $\mathcal{V}ars(\ell) \supseteq \mathcal{V}ars(r)$, and $\ell \notin \mathcal{V}$. A TRS $\mathcal{R}$ is left-linear if for every rule $\ell \to r \in \mathcal{R}$ the term ℓ is linear, i.e., no variable occurs more than once in ℓ. Given a TRS $\mathcal{R}$, its corresponding rewrite relation is defined as $s \to_{\mathcal{R}} t$ iff $s = C[\ell\sigma]$ and $t = C[r\sigma]$ for some rule $\ell \to r \in \mathcal{R}$, some context C and substitution σ. A TRS $\mathcal{R}$ is confluent if $\to_{\mathcal{R}}$ is confluent, and TRSs $\mathcal{R}$ and $\mathcal{S}$ commute if $\to_{\mathcal{R}}$ and $\to_{\mathcal{S}}$ commute. The multistep relation $\multimap_{\mathcal{R}}$ of a TRS is defined as the least relation such that $x \multimap_{\mathcal{R}} x$, $f(s_1, \ldots, s_n) \multimap_{\mathcal{R}} f(t_1, \ldots, t_n)$ whenever $s_i \multimap_{\mathcal{R}} t_i$ for all i, and $\ell\sigma \multimap_{\mathcal{R}} r\delta$ whenever $\ell \to r \in \mathcal{R}$ and $\sigma(x) \multimap_{\mathcal{R}} \delta(x)$ for all $x \in \mathcal{V}ars(\ell)$. It is known that $\to_{\mathcal{R}} \subseteq \multimap_{\mathcal{R}} \subseteq \to_{\mathcal{R}}^*$ for every TRS $\mathcal{R}$. Therefore, strong confluence of $\multimap_{\mathcal{R}}$ implies confluence of $\mathcal{R}$. Similarly, strong commutation of $\to_{\mathcal{S}}$ and $\multimap_{\mathcal{R}}$, which is equivalent to strong commutation of $\multimap_{\mathcal{S}}$ and $\multimap_{\mathcal{R}}$, implies commutation of $\mathcal{R}$ and $\mathcal{S}$.

Okui's criterion demands that a TRS $\mathcal{R}$ is left-linear and all SCPs $s \mathrel{\multimap\!\!\circ}_{\mathcal{R}} \circ \to_{\mathcal{R}} t$ are joinable in a certain way, i.e., $s \to_{\mathcal{R}}^* \circ \mathrel{\circ\!\!\multimap}_{\mathcal{R}} t$. He proved that for such TRSs, $\multimap_{\mathcal{R}}$ is strongly confluent and thus, $\mathcal{R}$ is confluent. Kirk and Middeldorp formalized this result and gave a definition of SCPs that is based on proof terms, cf. Definition 2.

A proof term of a TRS $\mathcal{R}$ is a term $A, B, \ldots \in \mathcal{T}(\mathcal{F} \uplus \mathcal{R}, \mathcal{V})$, i.e., where function symbols can either be normal function symbols in $\mathcal{F}$ or rule symbols $\alpha, \beta, \ldots$ where each rule symbol represents a rewrite rule $\ell \to r \in \mathcal{R}$. Here, the arity of a rule symbol α is exactly the number of variables in the corresponding rule. There is a one-to-one correspondence between multisteps and proof terms, where *src* and *tgt* are operations that project rule symbols to their left-hand sides and right-hand sides, respectively. We say that a proof term is empty, if it does not contain any rule symbols, i.e., the corresponding multistep does not perform a single rewrite step. Merging two compatible proof terms is performed with a $\sqcup$ operation, that represents the union of the the individual rewrite steps. The operation $RP(A)$ provides a distinct list of redex patterns of a proof term A, where pattern (α, p) is part of the list, if rule α is applied at position p in the step $src(A) \multimap tgt(A)$ that is encoded by A. The list is sorted by positions in ascending order. The function $OV(A, B)$ defines a new proof term that is similar to A, however, only those redex patterns of A are kept in $OV(A, B)$ that overlap with redexes of B.

Example 1. *If $\mathcal{R}$ consists of rules $\alpha : a \to b$ and $\beta : f(h(x), y) \to g(y, c, x)$, then $f(\underline{f(h(\underline{a}), a)}, z) \multimap_{\mathcal{R}} f(\underline{g(a, c, \underline{b})}, z)$ and this step is encoded in the proof term $A := f(\beta(\alpha, a), z)$, i.e., $src(A) = f(f(h(a), a), z)$ and $tgt(A) = f(g(a, c, b), z)$. Proof terms A and $B := f(f(h(\alpha), \alpha), z)$ are compatible and $A \sqcup B = f(\beta(\alpha, \alpha), z)$ with $tgt(A \sqcup B) = f(g(b, c, b), z)$. The lists of redex patterns in this example are $RP(A) = [(\beta, 1), (\alpha, 1.1.1)]$ and $RP(B) = [(\alpha, 1.1.1), (\alpha, 1.2)]$. Since $(\beta, 1) \in RP(A)$ does not overlap with redexes in B we obtain $RP[OV(A, B)] = [(\alpha, 1.1.1)]$ and $OV(A, B) = f(f(h(\alpha), a), z)$.*

The upcoming definition of SCPs is taken literally from Kirk and Middeldorp. Note that our main interest in this paper is to provide a verified implementation of SCPs. For the question why SCPs are defined in the way they are, we refer to the motivation given in the original papers [3, 6].

Definition 2 (Simultaneous Critical Pairs using Proof Terms [3, Definition 5.2]). *Let $\mathcal{R}$ be some TRS. We define $SCP(\mathcal{R})$ as the set of all pairs (A, B) of proof terms A and B of $\mathcal{R}$ that satisfy the following conditions for suitable values of $\alpha_1, p_1, \beta, p, \tau, \ldots$.*

1. $RP(A) = [(\alpha_1, p_1), \ldots, (\alpha_n, p_n)]$
2. $RP(B) = [(\beta, q)]$
3. $q = \varepsilon$ *or* $p_1 = \varepsilon$
4. $OV(A, B) = A$
5. *there are renaming substitutions* $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}$ *and* σ_β *such that all variables of rules* $\alpha_1, \ldots, \alpha_n, \beta$ *are renamed apart*
6. $\ell = lhs(\alpha_1)\sigma_{\alpha_1}[lhs(\beta)\sigma_\beta]_q$
7. τ *is a most general unifier of* $\{lhs(\alpha_1)\sigma_{\alpha_1} \approx \ell|_{p_1}, \ldots, lhs(\alpha_n)\sigma_{\alpha_n} \approx \ell|_{p_n}\}$
8. $A = \bigsqcup_{i=1}^{n} A_i$ *where* $A_i = \ell\tau[\alpha_i(\sigma_{\alpha_i}\tau)]_{p_i}$ *for all* $1 \le i \le n$
9. $B = \ell\tau[\beta(\sigma_\beta\tau)]_q$

Here, $lhs(\alpha)$ returns the left-hand side of a rule symbol α. The conditions in particular enforce $n \ge 1$, so proof term A must be non-empty.

Theorem 3 (Okui's Criterion using Proof Terms [3, Listing 3]). *If $\mathcal{R}$ is left-linear and $tgt(A) \to_{\mathcal{R}}^* \circ \mathrel{\leftarrow\!\!\circ\!\!-}_{\mathcal{R}} tgt(B)$ for all $(A, B) \in SCP(\mathcal{R})$, then $\mathrel{-\!\!\circ\!\!\to}_{\mathcal{R}}$ is strongly confluent and thus, $\mathcal{R}$ is confluent.*

3 From Confluence to Commutation

Our first aim is to try to generalize Definition 2 and Theorem 3 to commutation. Basically, an SCP (A, B) encodes the critical peak $tgt(A) \;{}_{\mathcal{R}}\!\mathrel{\leftarrow\!\!\circ\!\!-} src(A) = src(B) \to_{\mathcal{R}} tgt(B)$. In order to generalize this setting to commutation, we just have replace one of the TRSs in the peak by another TRS $\mathcal{S}$:

$$tgt(A) \;{}_{\mathcal{R}}\!\mathrel{\leftarrow\!\!\circ\!\!-} src(A) = src(B) \to_{\mathcal{S}} tgt(B).$$

To accommodate for this change we just have to generalize the definition of $SCP(\mathcal{R})$ to be based on two TRSs.

Definition 4 (SCPs for commutation of $\mathcal{R}$ and $\mathcal{S}$). *We define $SCP(\mathcal{R}, \mathcal{S})$ in the same way as $SCP(\mathcal{R})$ in Definition 2 with the only exception that instead of requiring that both A and B are proof terms of $\mathcal{R}$, we now demand that A is a proof term of $\mathcal{R}$ and B is a proof term of $\mathcal{S}$.*

Theorem 5 (Okui's Criterion for Commutation). *If $\mathcal{R}$ and $\mathcal{S}$ are left-linear and $tgt(A) \to_{\mathcal{S}}^* \circ \;{}_{\mathcal{R}}\!\mathrel{\leftarrow\!\!\circ\!\!-} tgt(B)$ for all $(A, B) \in SCP(\mathcal{R}, \mathcal{S})$, then $\to_{\mathcal{S}}$ and $\mathrel{-\!\!\circ\!\!\to}_{\mathcal{R}}$ strongly commute and thus, $\mathcal{R}$ and $\mathcal{S}$ commute.*

In order to arrive at this new theorem, we heavily used the existing formalization: It just took two hours to replace each occurrence of$\mathcal{R}$ by either $\mathcal{R}$ or $\mathcal{S}$ in the existing formalization,

i.e., we needed to make these replacement in every intermediate definition, every lemma and in every proof. The property that we made the correct replacements, and that every generalized statement is still provable was automatically checked by Isabelle: we did not encounter a single problem during the generalization process.

First note that Theorem 5 can easily be extended to show that joinability of SCPs actually characterizes strong commutation of $\multimap_{\mathcal{S}}$ and $\multimap_{\mathcal{R}}$. The reason is that one direction is immediate: since every SCP (A, B) encodes a peak $tgt(A)\ {}_{\mathcal{R}}\!\!\leftarrow\!\!\circ\ \circ\ \multimap_{\mathcal{S}} tgt(B)$, all these peaks are joinable in the desired form, provided that $\multimap_{\mathcal{S}}$ and $\multimap_{\mathcal{R}}$ strongly commute.

Corollary 6 (Strong Confluence and Strong Commutation via Simultaneous Critical Pairs)**.** *Let $\mathcal{R}$ and $\mathcal{S}$ be left-linear TRSs.*

- *The relations $\multimap_{\mathcal{S}}$ and $\multimap_{\mathcal{R}}$ strongly commute if and only if $tgt(A) \to_{\mathcal{S}}^* \circ\ {}_{\mathcal{R}}\!\!\leftarrow\!\!\circ\ tgt(B)$ for all $(A, B) \in SCP(\mathcal{R}, \mathcal{S})$.*
- *The relation $\multimap_{\mathcal{R}}$ is strongly confluent if and only if $tgt(A) \to_{\mathcal{R}}^* \circ\ {}_{\mathcal{R}}\!\!\leftarrow\!\!\circ\ tgt(B)$ for all $(A, B) \in SCP(\mathcal{R}, \mathcal{R})$.*

As a consequence of the corollary, we see that joinability of simultaneous critical pairs subsumes every criterion for left-linear TRSs that proves strong confluence (or strong commutation) of the multistep relation(s), e.g., the criterion of development closedness.

Further note that Theorem 5 is not symmetric, i.e., when actually trying to prove commutation of $\mathcal{R}$ and $\mathcal{S}$, one should try to apply Theorem 5 for $\mathcal{R}$ and $\mathcal{S}$ as stated, but also in the setting where the roles of $\mathcal{R}$ and $\mathcal{S}$ are swapped.

Finally note that for Okui's confluence criterion it is not known whether one can restrict Definition 2 in such a way that the single step is always at the root position (as it is done for parallel critical pairs), i.e., replace the disjunction $q = \varepsilon \vee p_1 = \varepsilon$ in Condition 3 just by $q = \varepsilon$. At least for commutation, we show that such a modification is not possible.

Example 7. *Let $\mathcal{R} = \{\alpha : f(a) \to b\}$ and $\mathcal{S} = \{\beta : a \to c\}$. These TRSs do not commute since $b\ {}_{\mathcal{R}}\!\leftarrow f(a) \to_{\mathcal{S}} f(c)$, and b and $f(c)$ are not joinable.*

If however we replace Condition 3 by $q = \varepsilon$, then $SCP(\mathcal{R}, \mathcal{S})$ must be empty. Otherwise, there would be some $(A, B) \in SCP(\mathcal{R}, \mathcal{S})$. But this implies that B must be β, i.e., $src(A) = src(B) = a$, and hence A must be an empty proof term, violating the conditions in Definition 4. But if $SCP(\mathcal{R}, \mathcal{S}) = \emptyset$ with the modification, it shows that Theorem 5 becomes unsound.

4 Implementing Simultaneous Critical Pairs

Our main aim was to be able to use the existing formalization for enabling support of Okui's criterion in CeTA. To this end, two tasks have to be solved: first, one needs a verified algorithm that actually computes $SCP(\mathcal{R})$ (or $SCP(\mathcal{R}, \mathcal{S})$ for commutation), and second one needs to check the joining sequences that are provided by automated confluence and commutation tools. Checking left-linearity of $\mathcal{R}$ and $\mathcal{S}$ is trivial.

For the second task, all the machinery is already included in CeTA, i.e., tools just need to output for each simultaneous critical pair (s, t) a joining sequence of terms $s \multimap u_1 \multimap u_2 \multimap \ldots \multimap u_n \leftarrow\!\!\circ\ t$, so that $s \to^* u_n \leftarrow\!\!\circ\ t$ can be concluded. Here, CeTA takes care of mismatches in names of variables in SCPs, e.g., by using different renaming substitutions, etc. Moreover, trivial critical pairs (t, t) can be omitted in certificates, too.

Hence, it remains to solve the first task, an implementation of Definition 2 (or the more general Definition 4) for finite TRSs. For presentation reasons we only speak about Definition 2 in the sequel, though the formalization covers Definition 4.

The first step we performed, is to actually change Definition 2, so that $SCP(\mathcal{R})$ becomes finite for finite TRSs. Definition 2 has the problem that there are infinitely many possible SCPs which just differ by a variable renaming. The problem is that with the existential quantifier one has to consider all renaming substitutions, and also every most general unifier τ. This problem is already partially solved in the Isabelle theories: there, the renaming substitutions are not arbitrary, but they are uniquely computed by some renaming algorithm, and this renaming algorithm is a parameter of the SCP definition. To deal with unifiers, we changed Condition 7 in a way that also the most general unifier τ is not an arbitrary one, but it is the single unifier that one gets from a fixed unification algorithm. Fortunately, this change did not require any big adjustments in the formal proof of Theorem 5.

Note that it is not immediately clear how to resolve the remaining existential quantifiers in Definition 2. For instance, A is defined via A_i where A_i can be computed only if (α_i, p_i) is known (Condition 8), and (α_i, p_i) is obtained from A (Condition 1). In order to break this cyclic dependence, we show that for finite TRSs there are only finitely many possibilities to choose the values of α_i and p_i. To this end, we use an alternative characterization of Condition 4 based on positions: condition $OV(A, B) = A$ is equivalent to

$$\forall 1 \le i \le n.\ \{p_i p \mid p \in FPos(lhs(\alpha_i))\} \cap \{qp \mid p \in FPos(lhs(\beta))\} \neq \emptyset.$$

Using this equivalence it will be possible to enumerate all possible redex patterns in Conditions 1 and 2 without knowing A and B.

Let us consider the easier case $q = \varepsilon$ first. Here, Condition 2 can be simplified to $q = \varepsilon \wedge \beta \in \mathcal{R}$, so an enumeration over all finitely many rules of $\mathcal{R}$ is possible to cover all possibilities to choose q and β. Since $q = \varepsilon$, condition $OV(A, B) = A$ simplifies further to

$$\forall 1 \le i \le n.\ p_i \in FPos(lhs(\beta)).$$

Hence, also for $RP(A)$ there are only finitely many possibilities, since the positions are taken from a finite set and there are only finitely many rule symbols. Instead of blindly enumerating all these possibilities, we use the following algorithm rp that takes a term t as input and returns a set of lists of redex patterns, namely those lists that correspond to proof terms where all redex patterns overlap with function positions of t. Here, $\oplus$ denotes list concatenation and $\mathit{unify\text{-}vd}(s, t)$ denotes that s and t unify after making the variables of these terms disjoint.

- $rp(x) = \{[]\}$
- $rp(f(t_1, \ldots, t_n)) = \Big(\bigcup_{ps_1 \in rp(t_1), \ldots, ps_n \in rp(t_n)} \bigoplus_{1 \le i \le n} [(\beta, i.p) \mid (\beta, p) \in ps_i]\Big)$
  $\cup \bigcup_{\alpha:\ell \to r \in \mathcal{R}, \mathit{unify\text{-}vd}(\ell, f(t_1, \ldots, t_n))} \mathit{rp\text{-}root}(f(t_1, \ldots, t_n), \alpha, \ell)$
- $\mathit{rp\text{-}root}(t, \alpha, \ell) = \bigcup_{ps_1 \in \mathit{rp\text{-}sub}(t,p_1), \ldots, ps_n \in \mathit{rp\text{-}sub}(t,p_n)} [(\alpha, \varepsilon)] \oplus \bigoplus_{1 \le i \le n} [(\alpha, p_i.p) \mid (\alpha, p) \in ps_i]$
  where $p_1, \ldots, p_n$ are the variable positions of ℓ
- $\mathit{rp\text{-}sub}(t, p) = rp(t|_p)$, if $p \in FPos(t)$, and $\mathit{rp\text{-}sub}(t, p) = \{[]\}$, otherwise

In the equation for $rp(f(t_1, \ldots, t_n))$ on the left of the $\cup$-operation we compute the redex patterns for those multisteps that do not perform a root step, and on the right of the $\cup$-operation are those redex patterns that include a root step. Here, the unification condition restricts the set of potential rules. For the root step itself, in $\mathit{rp\text{-}root}$ we insert the redex pattern (α, ε) for the root step at the front of the list, and then gather further redex patterns that are below variable positions of the left-hand side ℓ. Note that the case analysis in the definition of $\mathit{rp\text{-}sub}(t, p)$ is essential, since p is a variable position of ℓ, and this is not necessarily a position of t.

The algorithm rp and its auxiliary algorithms terminate, and it is characterized as follows.

Theorem 8 (Soundness of rp)**.** *Let $\mathcal{R}$ be left-linear. For every list ps and term t the following two statements are equivalent.*

- $ps \in rp(t)$.
- *There is some proof term A of $\mathcal{R}$ such that*

$$RP(A) = ps \wedge (\forall(\alpha, p) \in ps.\ p \in FPos(t) \wedge \mathit{unify\text{-}vd}(t|_p, lhs(\alpha))).$$

With the help of Theorem 8 we proved that SCPs with $q = \varepsilon$ can be computed as follows:

- Pick every rule symbol β of $\mathcal{R}$ and fix $q = \varepsilon$.
- Pick every non-empty list $[(\alpha_1, p_1), \ldots, (\alpha_n, p_n)]$ of $rp(lhs(\beta))$.
- Compute the remaining objects of Definition 2 following Condition 5 onwards.

For the case $q \neq \varepsilon$ we perform the following steps to compute the SCPs, again based on rp.

- Pick every rule $\alpha_1 : \ell_1 \to r_1 \in \mathcal{R}$ and fix $p_1 = \varepsilon$.
- Pick every $q \in FPos(\ell_1)$ satisfying $q \neq \varepsilon$.
- Pick every rule $\beta \in \mathcal{R}$ such that $\mathit{unify\text{-}vd}(lhs(\beta), \ell_1|_q)$.
- For each q_i from variable positions $q_2, \ldots, q_k$ of α_1, pick every qs_i from $rp(lhs(\beta)|_p)$ and define $ps_i = [(\gamma, q_i p') \mid (\gamma, p') \in qs_i]$ if $q_i = qp$ and $p \in FPos(lhs(\beta))$ for some p, and define $ps_i = [\,]$, otherwise.
- Define $(\alpha_2, p_2), \ldots, (\alpha_n, p_n)$ such that $[(\alpha_2, p_2), \ldots, (\alpha_n, p_n)] = \bigoplus_{i=2}^{k} ps_i$.
- Compute the remaining objects of Definition 2 following Condition 5 onwards.

The function *sim_cp_impl* of the IsaFoR library implements all these steps (using lists instead of sets) in the general case of Definition 4, and we verify that it exactly computes $SCP(\mathcal{R}, \mathcal{S})$. Moreover, the implementation is optimal in the sense that whenever the list representations of $\mathcal{R}$ and $\mathcal{S}$ are distinct, then so is $sim\_cp\_impl(\mathcal{R}, \mathcal{S})$. In particular, no critical pair is computed twice, since the implementation does not perform any explicit elimination of duplicates. We further tuned the implementation in a way that certain elements are precomputed, e.g., the variable positions of each left-hand side are only computed once.

We also integrated certificate generation for Okui's criterion in CSI [10], and because of our efforts, CSI is now able to produce certified proofs for three TRSs from the ARI-COPS database (# 419, # 463, and # 464), where prior to this work no certified proof was known. The generated proofs are verified by CeTA in a fraction of a second.

Both the formalization and the certified confluence proofs for the three TRSs are available on the following website. The website further includes links to the formalized definitions and theorems of this paper.

https://isafor-ceta.uibk.ac.at/experiments/iwc2026/

Acknowledgments

We thank the anonymous reviewers for their detailed and constructive feedback. This research was funded in part by the Austrian Science Fund (FWF) [Grant 10.55776/I5943].

References

[1] Franz Baader and Tobias Nipkow. *Term rewriting and all that.* Cambridge University Press, 1998.

[2] Nao Hirokawa, Dohan Kim, Kiraku Shintani, and René Thiemann. Certification of confluence- and commutation-proofs via parallel critical pairs. In Amin Timany, Dmitriy Traytel, Brigitte Pientka, and Sandrine Blazy, editors, *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2024, London, UK, January 15-16, 2024*, pages 147–161. ACM, 2024. doi:10.1145/3636501.3636949.

[3] Christina Kirk and Aart Middeldorp. Formalizing simultaneous critical pairs for confluence of left-linear rewrite systems. In Kathrin Stark, Amin Timany, Sandrine Blazy, and Nicolas Tabareau, editors, *Proceedings of the 14th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2025, Denver, CO, USA, January 20-21, 2025*, pages 156–170. ACM, 2025. doi:10.1145/3703595.3705881.

[4] Christina Kohl and Aart Middeldorp. A formalization of the development closedness criterion for left-linear term rewrite systems. In Robbert Krebbers, Dmitriy Traytel, Brigitte Pientka, and Steve Zdancewic, editors, *Proceedings of the 12th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2023, Boston, MA, USA, January 16-17, 2023*, pages 197–210. ACM, 2023. doi:10.1145/3573105.3575667.

[5] Tobias Nipkow, Lawrence C. Paulson, and Markus 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.

[6] Satoshi Okui. Simultaneous critical pairs and church-rosser property. In Tobias Nipkow, editor, *Rewriting Techniques and Applications, 9th International Conference, RTA-98, Tsukuba, Japan, March 30 - April 1, 1998, Proceedings*, Lecture Notes in Computer Science, pages 2–16. Springer, 1998. URL: https://doi.org/10.1007/BFb0052357, doi:10.1007/BFB0052357.

[7] René Thiemann and Christian Sternagel. Certification of termination proofs using CeTA. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *Theorem Proving in Higher Order Logics, 22nd International Conference, TPHOLs 2009, Munich, Germany, August 17-20, 2009. Proceedings*, Lecture Notes in Computer Science, pages 452–468. Springer, 2009. doi:10.1007/978-3-642-03359-9_31.

[8] Yoshihito Toyama. On the Church-Rosser property of term rewriting systems. *NTT ECL Technical Report (Japanese)*, 17672, 1981.

[9] Vincent van Oostrom. Development closed critical pairs. In Gilles Dowek, Jan Heering, Karl Meinke, and Bernhard Möller, editors, *Higher-Order Algebra, Logic, and Term Rewriting, Second International Workshop, HOA '95, Paderborn, Germany, September 21-22, 1995, Selected Papers*, Lecture Notes in Computer Science, pages 185–200. Springer, 1995. doi:10.1007/3-540-61254-8_26.

[10] Harald Zankl, Bertram Felgenhauer, and Aart Middeldorp. CSI - A confluence tool. In Nikolaj S. Bjørner and Viorica Sofronie-Stokkermans, editors, *Automated Deduction - CADE-23 - 23rd International Conference on Automated Deduction, Wroclaw, Poland, July 31 - August 5, 2011. Proceedings*, Lecture Notes in Computer Science, pages 499–505. Springer, 2011. doi:10.1007/978-3-642-22438-6_38.