universität
innsbruck

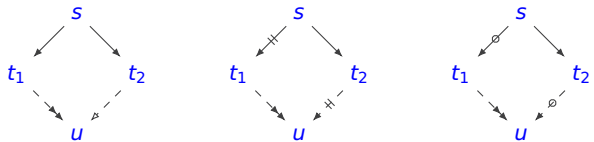
Verifying and Generalizing Simultaneous Critical Pairs

René Thiemann

International Workshop on Confluence, Lisbon, July 24, 2026

Background and Motivation: Proving Confluence

- strong confluence is a sufficient criterion for proving confluence



- new?: strong confluence is semi-decidable for left-linear TRSs
  - $\longrightarrow$ is strongly closed iff critical pairs are strongly closed
    new?: extension to left-linear TRSs
    (add critical pairs $\ell \longrightarrow\!\!\!\!\!\rightarrow \cdot \longleftarrow r\{x/x'\}$ for duplicating variables x and fresh x')
  - $\longrightarrow\!\!\!\!\!\!\shortmid\!\!\!\!\!\rightarrow$ is strongly closed iff Toyama's PCP-criterion is applicable
    new?: Toyama's PCP-criterion is complete
  - $\longrightarrow\!\!\!\!\!\!\circ\!\!\!\!\!\rightarrow$ is strongly closed iff Okui's criterion is applicable
- criteria are incomparable (62.ari + 73 $\longrightarrow$, 72 + 23 $\longrightarrow\!\!\!\!\!\!\shortmid\!\!\!\!\!\rightarrow$, 37 + 37 $\longrightarrow\!\!\!\!\!\!\circ\!\!\!\!\!\rightarrow$)
- new: strong commutation is semi-decidable for left-linear TRSs
  new: Okui's criterion extends to commutation

Okui's Theorem and Certification

- Okui's theorem on simultaneous critical pairs (SCPs): for left-linear TRS

   is strongly confluent iff all SCPs are joinable by
- Isabelle formalization of Okui's theorem by Kirk and Middeldorp (CPP 2025)
  - deviation to Okui's proof: proof terms to represent multisteps and SCPs
  - the formal definition of the set of SCPs is not executable
- consequence: CeTA, a verified certifier for confluence proofs, did not support Okui's criterion
- results
  - develop verified algorithm to compute SCPs,
    so that CeTA can certify applications of Okui's theorem
  - generalize result to commutation
  - CeTA-prover automates semi-decision procedures of strong CR / COM
- before doing so we introduce multistep rewriting and proof terms in this talk

Multistep Rewriting

- intuition: a multistep combines finitely many normal rewriting steps, provided that there is no overlap between these steps
- example for SRS 833.ari

$$\begin{aligned} \mathsf{baabb} &\to \mathsf{baaaabb} \\ \mathsf{babb} &\to \mathsf{bb} \\ \mathsf{babaaaab} &\to \mathsf{baaaabaaaababaaabaaaab} \\ \mathsf{baabaaaab} &\to \mathsf{babaabaaabaaaab} \\ \mathsf{baaabaaaab} &\to \mathsf{b} \end{aligned}$$

$$\overline{\mathsf{baaaabbaaabaaaa}}\overline{\mathsf{baaaabb}} \multimap \overline{\mathsf{baab}}\underline{\mathsf{baaabaaaab}}\overline{\mathsf{aabb}} \longrightarrow \mathsf{baab}\underline{\mathsf{b}}\mathsf{aabb}$$

- example for TRS $\{\alpha : \mathsf{a} \to \mathsf{b}, \quad \beta : \mathsf{f}(\mathsf{h}(x_1), x_2) \to \mathsf{g}(x_2, \mathsf{c}, x_1)\}$

$$\mathsf{f}(\underline{\mathsf{f}(\mathsf{h}(\underline{\mathsf{a}}), \mathsf{a})}, z) \multimap \mathsf{f}(\underline{\mathsf{g}(\mathsf{a}, \mathsf{c}, \underline{\mathsf{b}})}, z)$$

Proof Terms

- a proof term is a succinct representation of a multistep
- for each rule $\alpha : \ell \to r$, α is now considered as a new symbol
- the arity of symbol α is exactly the number of variables of rule α
- proof terms are just terms in this enlarged signature
- example for TRS $\{\alpha : \mathsf{a} \to \mathsf{b}, \quad \beta : \mathsf{f}(\mathsf{h}(x_1), x_2) \to \mathsf{g}(x_2, \mathsf{c}, x_1)\}$

$$\mathsf{f}(\underline{\mathsf{f}(\mathsf{h}(\underline{\mathsf{a}}), \mathsf{a})}, z) \twoheadrightarrow \mathsf{f}(\underline{\mathsf{g}(\mathsf{a}, \mathsf{c}, \underline{\mathsf{b}})}, z)$$

- step is encoded by proof term $A := \mathsf{f}(\beta(\alpha, \mathsf{a}), z)$, i.e., $src(A) \twoheadrightarrow tgt(A)$ where

$$\begin{aligned} src(x) &= x \\ src(f(t_1, \ldots, t_n)) &= f(src(t_1), \ldots, src(t_n)) \\ src(\alpha(t_1, \ldots, t_n)) &= lhs(\alpha)\{x_1/src(t_1), \ldots, x_n/src(t_n)\} \end{aligned}$$

  tgt is defined similarly where $lhs(\alpha)$ is replaced by $rhs(\alpha)$

Auxiliary Definitions used in Formal Definition of SCPs

- two proof terms A and B are compatible if all of the single redexes can be merged into a joined proof terms, which is then written as $A \sqcup B$
- $RP(A)$ is the list of redex patterns for a proof term A;
  (α, p) occurs in $RP(A)$ iff α is applied at position p in step $src(A) \multimap tgt(A)$;
  the list is sorted by position
- example TRS $\{\alpha : \mathsf{a} \to \mathsf{b}, \quad \beta : \mathsf{f}(\mathsf{h}(x_1), x_2) \to \mathsf{g}(x_2, \mathsf{c}, x_1)\}$
  - $A := \mathsf{f}(\beta(\alpha, \mathsf{a}), z)$ and $B := \mathsf{f}(\mathsf{f}(\mathsf{h}(\alpha), \alpha), z)$ are compatible
  - $A \sqcup B = \mathsf{f}(\beta(\alpha, \alpha), z)$
  - $src(A) = src(B) = src(A \sqcup B) = \mathsf{f}(\mathsf{f}(\mathsf{h}(\mathsf{a}), \mathsf{a}), z)$
  - $RP(A) = [(\beta, 1), (\alpha, 1.1.1)]$
- a proof term A is empty if $length(RP(A)) = 0$
- a proof term A is a single multistep if $length(RP(A)) = 1$
- non-empty A can be decomposed into single multisteps $A = A_1 \sqcup \ldots \sqcup A_n$

Definition (SCPs using Proof Terms [Kirk and Middeldorp])

Let $\mathcal{R}$ be some TRS. (A, B) is a SCP of $\mathcal{R}$ if all conditions are satisfied for suitable $\alpha_1, p_1, \ldots, \beta, q, \tau, \ldots$; it represents SCP $tgt(A) \leftarrow\!\circ\!- src(A) = src(B) \longrightarrow tgt(B)$

1. $RP(A) = [(\alpha_1, p_1), \ldots, (\alpha_n, p_n)]$ with $n > 0$
2. $RP(B) = [(\beta, q)]$
3. $q = \varepsilon$ or $p_1 = \varepsilon$
4. $OV(A, B) = A$ all redexes of A overlap with redex of B
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. $A = A_1 \sqcup \ldots \sqcup A_n$ where $A_i = \ell\tau[\alpha_i(\sigma_{\alpha_i}\tau)]_{p_i}$ for all $1 \leqslant i \leqslant n$
9. $B = \ell\tau[\beta(\sigma_\beta\tau)]_q$

First Problem: Infinitely SCPs

- problem: allowing arbitrary renamings or arbitrary most general unifiers results in infinitely many SCPs
- solution
  - fix a scheme to rename variables of rules (part of existing formalization)
  - fix a unification algorithm (minor modification of existing formalization)

Definition (SCPs using Proof Terms)

Let *ren* be some renaming algorithm and *mgu* be some unification algorithm. ...

5 there are $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}$ and σ_β for rules $\alpha_1, \ldots, \alpha_n, \beta$ computed by *ren*

7 $\tau = \mathit{mgu}\{\mathit{lhs}(\alpha_1)\sigma_{\alpha_1} \approx \ell|_{p_1}, \ldots, \mathit{lhs}(\alpha_n)\sigma_{\alpha_n} \approx \ell|_{p_n}\} \neq \bot$

- consequence for certification
  - tools might compute SCPs using different variable names
  - same problem was already occurring for standard critical pairs
  - same solution applies, too: check SCPs in certificates modulo variable names

Second Problem: $OV(A, B) = A$ condition

- fact: condition $OV(A, B) = A$ is crucial to ensure finiteness of set of SCPs
- problem: the definition of $OV(A, B)$ is rather unwieldy
  - decompose into single steps $A = A_1 \sqcup \ldots \sqcup A_n$
  - drop all single steps that do not overlap with B
  - join all remaining single steps
- solution: develop characterization of $OV(A, B) = A$ based on positions only

Lemma (Overlap Condition via Positions)

The following two conditions are equivalent in the definition of SCPs.

4. $OV(A, B) = A$
4. $\forall 1 \leqslant i \leqslant n.\ \{p_i p \mid p \in FPos(lhs(\alpha_i))\} \cap \{qp \mid p \in FPos(lhs(\beta))\} \neq \emptyset$

Here, $FPos(t)$ denotes the set of function-symbol positions of t.

Definition (SCPs using Proof Terms)

1. $RP(A) = [(\alpha_1, p_1), \ldots, (\alpha_n, p_n)]$ with $n > 0$
2. $RP(B) = [(\beta, q)]$
3. $q = \varepsilon$ or $p_1 = \varepsilon$
4. $\forall 1 \leqslant i \leqslant n.\ \{p_i p \mid p \in FPos(lhs(\alpha_i))\} \cap \{qp \mid p \in FPos(lhs(\beta))\} \neq \emptyset$
5. there are $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}$ and σ_β for rules $\alpha_1, \ldots, \alpha_n, \beta$ computed by *ren*
6. $\ell = lhs(\alpha_1)\sigma_{\alpha_1}[lhs(\beta)\sigma_\beta]_q$
7. $\tau = mgu\{lhs(\alpha_1)\sigma_{\alpha_1} \approx \ell|_{p_1}, \ldots, lhs(\alpha_n)\sigma_{\alpha_n} \approx \ell|_{p_n}\} \neq \bot$
8. $A = A_1 \sqcup \ldots \sqcup A_n$ where $A_i = \ell\tau[\alpha_i(\sigma_{\alpha_i}\tau)]_{p_i}$ for all $1 \leqslant i \leqslant n$
9. $B = \ell\tau[\beta(\sigma_\beta\tau)]_q$

- circularity problem: given $RP(A)$ and $RP(B)$ one can compute A and B or vice versa
- solution: algorithm to compute $RP(A)$ and $RP(B)$ without knowing A and B

Definition (SCPs using Proof Terms, only Case $q = \varepsilon$)

1. $RP(A) = [(\alpha_1, p_1), \ldots, (\alpha_n, p_n)]$ with $n > 0$
2. $RP(B) = [(\beta, \varepsilon)]$

4. $\forall 1 \leqslant i \leqslant n.\ p_i \in FPos(lhs(\beta))$
5. there are $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}$ and σ_β for rules $\alpha_1, \ldots, \alpha_n, \beta$ computed by *ren*
6. $\ell = lhs(\beta)\sigma_\beta$
7. $\tau = mgu\{lhs(\alpha_1)\sigma_{\alpha_1} \approx \ell|_{p_1}, \ldots, lhs(\alpha_n)\sigma_{\alpha_n} \approx \ell|_{p_n}\} \neq \bot$

...

Naive algorithm

- pick arbitrary β from $\mathcal{R}$
- pick arbitrary distinct and sorted positions $p_1, \ldots, p_n$ from $FPos(lhs(\beta))$
- pick arbitrary $\alpha_1, \ldots, \alpha_n$ from $\mathcal{R}$
- compute other values $\sigma_{\alpha_1}, \ldots, \sigma_{\alpha_n}, \sigma_\beta, \ell, \tau, A_1, \ldots, A_n, A, B$
- if all conditions are satisfied, generate SCP (A, B)

Improved Verified Algorithm to Compute SCPs

- define auxiliary algorithm $rp(t)$
- $rp(t)$ computes exactly those lists of redex pairs ps, for which
  - there is some proof term A of $\mathcal{R}$ such that $RP(A) = ps$, and
  - $\forall (\alpha, p) \in ps.\ p \in FPos(t) \wedge unify\text{-}vd(t|_p, lhs(\alpha))$
- using rp, we show that SCPs with $q = \varepsilon$ can be computed as follows:
  - pick an arbitrary rule β of $\mathcal{R}$ and fix $q = \varepsilon$
  - pick an arbitrary non-empty list $[(\alpha_1, p_1), \ldots, (\alpha_n, p_n)]$ of $rp(lhs(\beta))$
  - compute the remaining values by following Condition **5** onwards
- algorithm is implemented as function from list of rules ($\mathcal{R}$) to list of pairs of proof terms (SCPs) – also including case $q \neq \varepsilon$
- nice properties of algorithm
  - exact: it computes exactly the set of SCP — verified
  - optimal: it does not produce duplicates — verified
  - fast: 84 CSI proofs with SCPs are checked in less than 0.1 seconds each

Definition of Auxiliary Algorithm

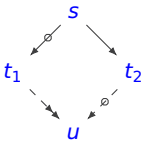
- $rp(x) = \{[]\}$
- $rp(f(t_1, \ldots, t_n)) = \Big(\bigcup_{ps_1 \in rp(t_1), \ldots, ps_n \in rp(t_n)} \bigoplus_{1 \leqslant i \leqslant n} [(\alpha, i.p) \mid (\alpha, p) \in ps_i]\Big)$
  $\cup \bigcup_{\alpha : \ell \to r \in \mathcal{R}, \mathit{unify\text{-}vd}(\ell, f(t_1, \ldots, t_n))} \mathit{rp\text{-}root}(f(t_1, \ldots, t_n), \alpha, \ell)$
- $\mathit{rp\text{-}root}(t, \alpha, \ell) = \bigcup_{ps_1 \in \mathit{rp\text{-}sub}(t, p_1), \ldots, ps_n \in \mathit{rp\text{-}sub}(t, p_n)}$
  $[(\alpha, \varepsilon)] \oplus \bigoplus_{1 \leqslant i \leqslant n} [(\alpha, p_i.p) \mid (\alpha, p) \in ps_i]$

  where $p_1, \ldots, p_n$ are the variable positions of ℓ
- $\mathit{rp\text{-}sub}(t, p) = rp(t|_p)$, if $p \in \mathit{FPos}(t)$, and $\mathit{rp\text{-}sub}(t, p) = \{[]\}$, otherwise

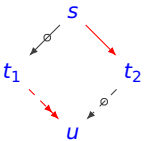
- note: the actual implementation uses lists instead of sets and $\oplus$ instead of $\cup$

Simultaneous Critical Pairs for Commutation

- one can generalize (strong) confluence to (strong) commutation



strong confluence of →



strong commutation of → and →

- Okui's theorem generalized to commutation
  - if two TRSs $\mathcal{R}_1$ and $\mathcal{R}_2$ are left-linear and
  - for every SCP $t_1 \leftarrow \cdot \rightarrow t_2$ there is u such that $t_1 \rightarrow u \leftarrow t_2$
  - then → and → strongly commute, and thus, the TRSs $\mathcal{R}_1$ and $\mathcal{R}_2$ commute
- methodology

  just replace TRS $\mathcal{R}$ by either $\mathcal{R}_1$ or $\mathcal{R}_2$ in existing definitions and lemmas

Experimental Results

- 3 new TRSs solved by CSI/CeTA (419, 463, 464) no previous certified proof
- observation: joinability search in CSI is weak: 
- development of CeTA-prover
  - based on verified CeTA-code
  - combined with unverified and unbounded BFS-search of 
  - semi-decision procedure for strong confluence and strong commutation
- results on COM
  - full run 2024: other tools at most 17 YES, union: 22
  - CeTA-prover on same dataset: 24
- results of CeTA-prover on CR on ARI-DB (success in 3 seconds or timeout)
  - only : 66
  - only : 83
  - only : 88
  - union: 92

Summary

- verified algorithm to compute SCPs in Isabelle/HOL
  - exact: computes precisely the set of SCPs in list representation
  - optimal: no duplicates
  - fast: less than 0.1 seconds to certify CSI certificates that use SCPs
- generalized existing formalization to Okui's theorem for commutation
- effort
  - implementation: 3811 lines of Isabelle ∼ two full weeks
  - commutation version: trivial changes in existing formalization one morning
- new tool: CeTA-prover automates strong confluence and strong commutation

Thank you! Questions?