Certifying Confluence Proofs using CeTA 2.16*

Julian Nagele René Thiemann Harald Zankl
Institute of Computer Science, University of Innsbruck, Austria

Automatic provers have become popular in several areas like first-order theorem proving, SMT, etc. Since these provers are complex pieces of software, they might contain errors which might lead to wrong answers, i.e., incorrect proofs. Therefore, certification of the generated proofs is of major importance, where soundness of the certifier itself should be proven in some trusted proof assistance like Isabelle/HOL [3]. The tool CeTA is such a certifier [5]. Its soundness is proven in the corresponding IsaFoR-library (Isabelle Formalization of Rewriting) and CeTA can be used to check termination and non-termination proofs of term rewrite systems (TRSs). Starting from version 2.0, it is also possible to certify confluence and non-confluence proofs where the following techniques are currently supported in CeTA (version 2.16).

- Since CeTA's main domain are termination proofs, as a first method to decide confluence we integrated Newman's lemma in combination with the critical pair theorem.
- For possibly non-terminating TRSs, we can ensure that weakly orthogonal and strongly closed TRSs are confluent. Recently we also added support for the rule labeling technique of van Oostrom [4] based on Zankl's formalization of decreasing diagrams [6]. Here one simply has to provide the labeling function and the joining sequences showing all critical peaks decreasing.
- To disprove confluence one can provide two derivations $s \to^* t_1$ and $s \to^* t_2$ and a reason why t_1 and t_2 cannot be joined. Here CeTA supports: t_1 and t_2 are distinct normal forms, testing that $tcap(t_1\sigma)$ and $tcap(t_2\sigma)$ are not unifiable [7] and usable rules, discrimination pairs, argument filters and interpretations [1]. Finally one can provide two tree automata $\mathcal{A}_i$ with $t_i \in \mathcal{L}(\mathcal{A}_i)$, $\mathcal{A}_i$ closed under $\mathcal{R}$ for $i = 1, 2$, and $\mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2) = \varnothing$ (see [2]).

For further details we refer to the certification problem format (CPF) and to the sources of IsaFoR and CeTA (`http://cl-informatik.uibk.ac.at/software/ceta/`). It remains as ongoing and future work to integrate existing and future confluence and non-confluence criteria.

References

[1] T. Aoto. Disproving confluence of term rewriting systems by interpretation and ordering. In *FroCoS*, volume 8152 of *LNCS*, pages 311–326, 2013.

[2] B. Felgenhauer and R. Thiemann. Reachability analysis with state-compatible automata. In *LATA*, volume 8370 of *LNCS*, pages 347–359, 2014.

[3] T. Nipkow, L.C. Paulson, and M. Wenzel. *Isabelle/HOL – A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*. Springer, 2002.

[4] V. van Oostrom. Confluence by decreasing diagrams – converted. In *RTA*, volume 5117 of *LNCS*, pages 306–320, 2008.

[5] R. Thiemann and C. Sternagel. Certification of termination proofs using CeTA. In *TPHOLs*, volume 5674 of *LNCS*, pages 452–468. Springer, 2009.

[6] H. Zankl. Confluence by decreasing diagrams – formalized. In *RTA*, volume 21 of *LIPIcs*, pages 352–367, 2013.

[7] H. Zankl, B. Felgenhauer, and A. Middeldorp. CSI – A confluence tool. In *CADE*, volume 6803 of *LNAI*, pages 499–505, 2011.

*Supported by Austrian Science Fund (FWF), projects P22467 and P22767.