Certification of Confluence Proofs using CeTA*

René Thiemann, Institute of Computer Science, University of Innsbruck, Austria

Automatic provers have become popular in several areas like first-order theorem proving, SMT, ... 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 might be proven in some trusted proof assistance like Coq [1] or Isabelle/HOL [5]. The tool CeTA is such a certifier [6]. 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 2.5).

- 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 [4]. Here, CeTA just rewrites both sides of a critical pair to arbitrary normal forms and then compares whether these normal forms coincide.

- For possibly non-terminating TRSs, we integrated the criterion of weak orthogonality to ensure confluence. The reason for not (yet) considering left-linear parallel-closed TRSs is the complexity of the soundness proof. For parallel-closed TRSs a complex measure on parallel forks is utilized [3], whereas for weak orthogonality structural induction suffices.

- To disprove confluence one can provide two forking derivations $s \to^* t_1$ and $s \to^* t_2$ in combination with a reason why t_1 and t_2 cannot be joined. Here, CeTA accepts the reason that t_1 and t_2 are distinct normal forms or alternatively, a test is performed using *tcap* [2, 7]: if $tcap(t_1\sigma)$ and $tcap(t_2\sigma)$ are not unifiable then a join is impossible (where σ is a substitution which replaces each variable x by some fresh constant c_x.)

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 future and ongoing work to integrate existing and future confluence and non-confluence criteria.

References

[1] Y. Bertot and P. Castéran. *Interactive Theorem Proving and Program Development; Coq'Art: The Calculus of Inductive Constructions.* TCS Texts. Springer, 2004.

[2] J. Giesl, R. Thiemann, and P. Schneider-Kamp. Proving and disproving termination of higher-order functions. In *FroCoS*, volume 3717 of *LNAI*, pages 216–231. Springer, 2005.

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

[4] D.E. Knuth and P. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.

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

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

[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): P22767