CO3 (Version 2.3)

Naoki Nishida and Misaki Kojima

Nagoya University, Nagoya, Japan
{nishida@, k-misaki@trs.css.}i.nagoya-u.ac.jp

CO3, a converter for proving confluence of conditional TRSs,[1] tries to prove confluence of conditional term rewrite systems (CTRSs, for short) by using a transformational approach (cf. [7]). The tool first transforms a given weakly-left-linear (WLL, for short) 3-DCTRS into an unconditional term rewrite system (TRS, for short) by using $\mathbb{U}_{conf}$ [3], a variant of the *unraveling* $\mathbb{U}$ [9], and then verifies confluence of the transformed TRS by using the following theorem: A 3-DCTRS $\mathcal{R}$ is confluent if $\mathcal{R}$ is WLL and $\mathbb{U}_{conf}(\mathcal{R})$ is confluent [2, 3]. The tool is very efficient because of very simple and lightweight functions to verify properties such as confluence and termination of TRSs.

Since version 2.0, a *narrowing-tree*-based approach [8, 4] to prove infeasibility of a condition w.r.t. a CTRS has been implemented [5]. The approach is applicable to *syntactically deterministic* CTRSs that are operationally terminating and *ultra-right-linear* w.r.t. the *optimized* unraveling. To prove infeasibility of a condition c, the tool first prove confluence, and then linearizes c if failed to prove confluence; then, the tool computes and simplifies a narrowing tree for c, and examines the emptiness of the narrowing tree.

The current version accepts both *join* and *semi-equational* CTRSs, and transforms them into equivalent DCTRSs to prove confluence or infeasibility [6].

This version has two improvements compared with the previous one [6]. One is the removal of valid conditions: For a conditional rule $\ell \to r \Leftarrow s_1 \twoheadrightarrow t_1, \ldots, s_k \twoheadrightarrow t_k \in \mathcal{R}$, a *valid* conditions $s_i \twoheadrightarrow t_i$ such that either s_i is a variable appearing once in the rule or $s_i \to t_i$ is in $\mathcal{R}$ as an unconditional rule is removed from the conditional part. The other is the computation of approximated edges for estimated dependency graphs: For dependency pairs $s \to t, u \to v$ of a TRS $\mathcal{R}$, if t is ground and $REN(CAP(t))$ is not unifiable with u, then we check whether there exists a reduct t' of t such that $REN(CAP(t'))$ is unifiable with u (cf. a *narrowing processor* [1]).

References

[1] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Mechanizing and improving dependency pairs. *J. Autom. Reason.*, 37(3):155–203, 2006.

[2] K. Gmeiner, B. Gramlich, and F. Schernhammer. On soundness conditions for unraveling deterministic conditional rewrite systems. In *Proc. RTA 2012*, vol. 15 of *LIPIcs*, pp. 193–208, 2012.

[3] K. Gmeiner, N. Nishida, and B. Gramlich. Proving confluence of conditional term rewriting systems via unravelings. In *Proc. IWC 2013*, pp. 35–39, 2013.

[4] Y. Maeda, N. Nishida, M. Sakai, and T. Kobayashi. Extending narrowing trees to basic narrowing in term rewriting. IEICE Tech. Rep. SS2018-39, Vol. 118, No. 385, pp. 73–78, 2019, in Japanese.

[5] N. Nishida. CO3 (Version 2.1). In *Proc. IWC 2020*, page 67, 2020.

[6] N. Nishida. CO3 (Version 2.2). In *Proc. IWC 2021*, page 61, 2021.

[7] N. Nishida, T. Kuroda, and K. Gmeiner. CO3 (Version 1.3). In *Proc. IWC 2016*, p. 74, 2016.

[8] N. Nishida and Y. Maeda. Narrowing trees for syntactically deterministic conditional term rewriting systems. In *Proc. FSCD 2018*, vol. 108 of *LIPIcs*, pp. 26:1–26:20, 2018.

[9] E. Ohlebusch. Termination of logic programs: Transformational methods revisited. *Appl. Algebra Eng. Commun. Comput.*, 12(1/2):73–116, 2001.

[1] http://www.trs.css.i.nagoya-u.ac.jp/co3/