Hakusan 0.12: A Confluence Tool

Fuyuki Kawano, Hiroka Hondo, Nao Hirokawa, and Kiraku Shintani

JAIST, Japan
{f-kawano,s2410171,hirokawa}@jaist.ac.jp, s.kiraku@gmail.com

Hakusan (https://www.jaist.ac.jp/project/saigawa/) is a confluence tool for left-linear term rewrite systems (TRSs). It analyzes confluence by successive application of *rule removal* criteria [5, 6, 10] based on rule labeling [9, 13], critical pair systems [4], and the generalization of Knuth and Bendix' criterion by Klein and Hirokawa [7]. Hakusan can produce proof certificates verifiable by CeTA [11], see [3].

Compared to the last version of Hakusan [6], non-confluence analysis has been improved by adopting the approach used in CSI [8]. Given a TRS $\mathcal{R}$ and convertible terms $t_1 \leftrightarrow_{\mathcal{R}}^* t_2$, this approach constructs two tree automata $\mathcal{A}_1$ and $\mathcal{A}_2$ such that $L(\mathcal{A}_i)$ is closed under $\mathcal{R}$-rewriting and $t_i \in L(\mathcal{A}_i)$ for each i. If $L(\mathcal{A}_1) \cap L(\mathcal{A}_2) = \varnothing$ then t_1 and t_2 are not joinable. Thus, non-confluence of $\mathcal{R}$ is concluded. Construction of $\mathcal{A}_i$ is heuristically done by tree automata completion [2] and closedness of its language under rewriting is tested by checking state compatibility and state coherence [1]. Tree automata completion uses the following criterion for testing closedness under rewriting.

Proposition 1. *Let $\mathcal{A} = (\mathcal{F}, Q, Q_f, \Delta)$ be a deterministic tree automaton and $\mathcal{R}$ a TRS. The language $L(\mathcal{A})$ is closed under $\mathcal{R}$-rewriting if $\mathcal{A}$ is compatible with all rules in $\mathcal{R}$. Here* compatibility *with $\ell \to r$ means that $r\sigma \to_\Delta^* q$ whenever $\sigma : \mathcal{V}\mathsf{ar}(\ell) \to Q$ and $\ell\sigma \to_\Delta^* q \in Q$.*

Example 1. *Consider the TRS $\mathcal{R}$ over the signature $\mathcal{F} = \{\mathsf{h}, \mathsf{f}, \mathsf{g}, \mathsf{a}, \mathsf{b}\}$:*

$$1:\ \mathsf{h}(\mathsf{g},\mathsf{a},\mathsf{a}) \to \mathsf{h}(\mathsf{f},\mathsf{a},\mathsf{a}) \qquad 2:\ \mathsf{h}(x,\mathsf{b},y) \to \mathsf{h}(x,y,y) \qquad 3:\ \mathsf{f} \to \mathsf{g} \qquad 4:\ \mathsf{a} \to \mathsf{b}$$

By the aforementioned approach we can prove that the convertible terms $t_1 = \mathsf{h}(\mathsf{g},\mathsf{b},\mathsf{b})$ and $t_2 = \mathsf{h}(\mathsf{g},\mathsf{a},\mathsf{a})$ are not joinable. Here we illustrate how compatibility for Proposition 1 is tested. Consider the deterministic tree automaton $\mathcal{A}_1 = (\mathcal{F}, Q, \{2\}, \Delta)$ with $Q = \{0,1,2\}$ and $\Delta = \{\mathsf{g} \to 0, \mathsf{b} \to 1, \mathsf{h}(0,1,1) \to 2\}$. This automaton satisfies $L(\mathcal{A}_1) = \{t_1\}$ and the compatibility with all rules in $\mathcal{R}$. For example, the compatibility with rule 2 is verified by checking that all triples $(q_1,q_2,q_3) \in Q^3$ satisfy the implication $\mathsf{h}(q_1,\mathsf{b},q_2) \to_\Delta^* q_3 \implies \mathsf{h}(q_1,q_2,q_2) \to_\Delta^* q_3$.

Compatibility check is a major bottleneck in tree automata completion. To ease the check, our tool exploits *persistency* of reachability.

Proposition 2. *Let $\mathcal{R}$ be a many-sorted TRS and Θ a sort elimination-operator [12]. If s is well-sorted then $s \to_{\mathcal{R}}^* t$ and $\Theta(s) \to_{\Theta(\mathcal{R})}^* \Theta(t)$ are equivalent.*

Thus, we can analyze reachability after performing type introduction. Type disciple enables us to discard ill-sorted state substitutions for compatibility check.

Example 2 (continued from Example 1)**.** *The TRS $\mathcal{R}$ and the terms t_1 and t_2 can be seen as a TRS and terms over the many-sorted signature:*

$$\mathsf{h} : A \times B \times B \to C \qquad \mathsf{f} : A \qquad \mathsf{g} : A \qquad \mathsf{a} : B \qquad \mathsf{b} : B$$

The sorted signature naturally assigns sorts to the states of $\mathcal{A}_1$ as follows:

$$0 : A \qquad 1 : B \qquad 2 : C$$

As $\mathsf{h}(q_1,\mathsf{b},q_2) \to_\Delta^* q_3$ induces $q_1 : A$, $q_2 : B$, and $q_3 : C$, the compatibility of $\mathcal{A}$ with rule 2 follows by testing the previous implication only for $(q_1,q_2,q_3) \in \{0\} \times \{1\} \times \{2\}$.

References

[1] B. Felgenhauer and R. Thiemann. Reachability, confluence, and termination analysis with state-compatible automata. *Information and Computation*, 253:467–483, 2017.

[2] T. Genet. Decidable approximations of sets of descendants and sets of normal forms. In *Proceedings of 9th International Conference on Rewriting Techniques and Applications*, volume 1379 of *LNCS*, pages 151–165, 1998. doi:doi.org/10.1007/BFb0052368.

[3] N. Hirokawa, D. Kim, K. Shintani, and R. Thiemann. Certification of confluence- and commutation-proofs via parallel critical pairs. In *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pages 147–161, 2024. doi:10.1145/3636501.3636949.

[4] N. Hirokawa and A. Middeldorp. Decreasing diagrams and relative termination. *Journal of Automated Reasoning*, 47:481–501, 2011. doi:10.1007/s10817-011-9238-x.

[5] N. Hirokawa and K. Shintani. Rule removal for confluence. In *Proceedings of 13th International Workshop on Confluence*, pages 50–54, 2024.

[6] F. Kawano, N. Hirokawa, and K. Shintani. Hakusan 0.11: A confluence tool. In *Proceedings of 13th International Workshop on Confluence*, pages 67–68, 2024.

[7] D. Klein and N. Hirokawa. Confluence of non-left-linear TRSs via relative termination. In *Proceedings of 18th International Conference on Logic Programming and Automated Reasoning*, volume 7180 of *LNCS*, pages 258–273, 2012. doi:10.1007/978-3-642-28717-6_21.

[8] J. Nagele, B. Felgenhauer, and A. Middeldorp. Csi: New evidence — a progress report. In *Proceedings of 26th International Conference on Automated Deduction*, pages 385–397, 2017. doi:10.1007/978-3-319-63046-5_24.

[9] V. van Oostrom. Confluence by decreasing diagrams, converted. In *Proceedings of 19th International Conference on Rewriting Techniques and Applications*, volume 5117 of *LNCS*, pages 306–320, 2008. doi:10.1007/978-3-540-70590-1_21.

[10] K. Shintani and N. Hirokawa. Compositional confluence criteria. *Logical Methods in Computer Science*, 20:6:1–6:28, 2024. doi:10.46298/lmcs-20(1:6)2024.

[11] R. Thiemann and C. Sternagel. Certification of termination proofs using CeTA. In *Proceedings of the 22nd International Conference on Theorem Proving in Higher Order Logics*, volume 5674 of *LNCS*, pages 452–468, 2009.

[12] J. van de Pol. Modularity in many-sorted term rewriting systems. Technical report, Utrecht University, 1992. Master's thesis.

[13] H. Zankl, B. Felgenhauer, and A. Middeldorp. Labelings for decreasing diagrams. *Journal of Automated Reasoning*, 54(2):101–133, 2015. doi:10.1007/s10817-014-9316-y.