AProVE: Becoming Open Source and Recent Improvements

Florian Frohn
RWTH Aachen University, Germany

Carsten Fuhs
Birkbeck, University of London, UK

Jürgen Giesl
RWTH Aachen University, Germany

Jan-Christoph Kassing
RWTH Aachen University, Germany

Nils Lommen
RWTH Aachen University, Germany

Abstract

AProVE (Automated Program Verification Environment) is a tool for fully automatic program verification. More precisely, AProVE is able to analyze termination, complexity, and safety of different programming languages, e.g., Java, C, Haskell, and Prolog. To ensure the correctness of its analysis, AProVE can generate certificates that can be checked by external certification tools. For further details on AProVE's general approach, see [1].

Since the last workshop on termination, several improvements were developed within AProVE. First of all, we want to announce that AProVE will be released as an open source tool by the end of this year. Opening the tool to the community will allow researchers and developers to explore its techniques and experiment with new strategies (e.g., for termination proofs). We hope this will foster the development of novel methods and lead to improved strategies for all supported techniques. In celebration of this milestone, we present a new logo, see Fig. 1.

Figure 1 New AProVE Logo

Furthermore, AProVE will participate in this year's confluence competition (CoCo) [9] for the first time. Several of AProVE's termination techniques rely on confluence properties, so participating in CoCo provides an opportunity to evaluate how well AProVE's techniques to prove confluence compare with specialized tools in that area.

Finally, AProVE will keep on participating in the annual termination competition [2]. Compared to previous years, we have integrated several improvements in multiple categories. In particular, we improved the termination and complexity analysis of probabilistic term rewriting [3, 4, 5], the termination analysis of relative term rewriting [6], the derivational complexity analysis of standard term rewriting [7], and more. Moreover, the analysis of C-programs has been improved by integrating new versions of the tools KoAT and LoAT [8].

2012 ACM Subject Classification Theory of computation → Logic and verification

Keywords and phrases Termination, Complexity, Safety, Automation

References

1 Jürgen Giesl, Cornelius Aschermann, Marc Brockschmidt, Fabian Emmes, Florian Frohn, Carsten Fuhs, Jera Hensel, Carsten Otto, Martin Plücker, Peter Schneider-Kamp, Thomas Ströder, Stephanie Swiderski, and René Thiemann. Analyzing program termination and complexity automatically with AProVE. *J. Autom. Reason.*, 58(1):3–31, 2017. doi:10.1007/s10817-016-9388-y.

2 Jürgen Giesl, Albert Rubio, Christian Sternagel, Johannes Waldmann, and Akihisa Yamada. The termination and complexity competition. In *Proc. TACAS '19*, LNCS 11429, pages 156–166, 2019. Website of *TermComp*: `https://termination-portal.org/wiki/Termination_Competition`. `doi:10.1007/978-3-030-17502-3_10`.

3 Jan-Christoph Kassing, Stefan Dollase, and Jürgen Giesl. A complete dependency pair framework for almost-sure innermost termination of probabilistic term rewriting. In *Proc. FLOPS '24*, LNCS 14659, pages 62–80, 2024. `doi:10.1007/978-981-97-2300-3_4`.

4 Jan-Christoph Kassing, Florian Frohn, and Jürgen Giesl. From innermost to full almost-sure termination of probabilistic term rewriting. In *Proc. FoSSaCS '24*, LNCS 14575, pages 206–228, 2024. `doi:10.1007/978-3-031-57231-9_10`.

5 Jan-Christoph Kassing and Jürgen Giesl. Annotated dependency pairs for full almost-sure termination of probabilistic term rewriting. In *Principles of Verification: Cycling the Probabilistic Landscape*, LNCS 15260, pages 339–366, 2024. `doi:10.1007/978-3-031-75783-9_14`.

6 Jan-Christoph Kassing, Grigory Vartanyan, and Jürgen Giesl. A dependency pair framework for relative termination of term rewriting. In *Proc. IJCAR '24*, LNCS 14740, pages 360–380, 2024. `doi:10.1007/978-3-031-63501-4_19`.

7 Jan-Christoph Kassing and Jürgen Giesl. From innermost to full probabilistic term rewriting: Almost-sure termination, complexity, and modularity, 2025. URL: `https://arxiv.org/abs/2409.17714`, `arXiv:2409.17714`.

8 Nils Lommen and Jürgen Giesl. AProVE (KoAT+ LoAT). In *Proc. TACAS '25*, LNCS 15698, pages 205–211, 2025. `doi:10.1007/978-3-031-90660-2_13`.

9 Aart Middeldorp, Julian Nagele, and Kiraku Shintani. Confluence competition 2019. In *Proc. TACAS '19*, LNCS 11429, pages 25–40, 2019. `doi:10.1007/978-3-030-17502-3_2`.