

Symbolic Evaluation Graphs and Term Rewriting — A General Methodology for Analyzing Logic Programs^{*}

Jürgen Giesl¹, Thomas Ströder¹, Peter Schneider-Kamp², Fabian Emmes¹, and Carsten Fuhs³

¹ LuFG Informatik 2, RWTH Aachen University, Germany

² Dept. of Mathematics and Computer Science, University of Southern Denmark

³ Dept. of Computer Science, University College London, UK

There exist many powerful techniques to analyze *termination* and *complexity* of *term rewrite systems* (TRSs). Our goal is to use these techniques for the analysis of other programming languages as well. For instance, approaches to prove termination of definite logic programs by a transformation to TRSs have been studied for decades. However, a challenge is to handle languages with more complex evaluation strategies (such as *Prolog*, where predicates like the *cut* influence the control flow).

We present a general methodology for the analysis of such programs. Here, the logic program is first transformed into a *symbolic evaluation graph* which represents all possible evaluations in a finite way. Afterwards, different analyses can be performed on these graphs. In particular, one can generate TRSs from such graphs and apply existing tools for termination or complexity analysis of TRSs to infer information on the termination or complexity of the original logic program.

More information can be found in [1].

References

1. J. Giesl, T. Ströder, P. Schneider-Kamp, F. Emmes, and C. Fuhs. Symbolic evaluation graphs and term rewriting — a general methodology for analyzing logic programs. In *Proc. PPDP '12*, pages 1–12. ACM Press, 2012.

^{*} Supported by the DFG under grants GI 274/5-3 and GI 274/6-1, the DFG Research Training Group 1298 (*AlgoSyn*), and the Danish Council for Independent Research, Natural Sciences.