PGF transformer semantics

Cheng Peng Dantong Liu

May 29, 2024

You may have doubts

The theme: PGF transform semantics based equivalence checking of pGCL programs

Equivalence checking?

This work is about linearity no normalization is ever involved.

I heard the problem is undecidable!

We will focus on a restricted fragment of pGCL called ReDiP.

Contents in this section

Backgrounds

Motivation

Contents in this section

The research questions

Overview of our solution

Contents in this section

Syntax of pGCL

Semantics of pGCL

A fragement of pGCL: ReDiP

Contents in this section

Review on PGFs

PGF transformer semantics

Fixed Point Induction

Linearity & Closed-form preservation

Summary

Contents in this section

Benchmark setup

Results

Case Study

Contents in this section

Conclusion

Limitations and future works

Q & A

Questions are welcomed.

Contents in this section

references I

[1] Mingshuai Chen et al. Does a Program Yield the Right Distribution? Verifying Probabilistic Programs via Generating Functions. 2022. arXiv: 2205.01449 [cs.LO].

[2] He Jifeng, K. Seidel, and A. McIver. "Probabilistic models for the guarded command language". In: Science of Computer Programming 28.2 (1997). Formal Specifications: Foundations, Methods, Tools and Applications, pp. 171–192. ISSN: 0167-6423. DOI: https://doi.org/10.1016/S0167-6423(96)00019-6. URL: https://www.sciencedirect.com/science/article/pii/S0167642396000196.

[3] Lutz Klinkenberg et al. "Exact Bayesian Inference for Loopy Probabilistic Programs using Generating Functions". In: Proc. ACM Program. Lang. 8.OOPSLA1 (2024).

[4] Lutz Klinkenberg et al. "Exact Probabilistic Inference Using Generating Functions". In: LAFI. [Extended Abstract]. 2023.

[5] Aaron Meurer et al. "SymPy: symbolic computing in Python". In: PeerJ Computer Science 3 (Jan. 2017), e103. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.103. URL: https://doi.org/10.7717/peerj-cs.103.

[6] Chen Mingshuai et al. "Does a program yield the right distribution? Verifying probabilistic programs via generating functions". In: International Conference on Computer Aided Verification. Springer. 2022, pp. 79–101.

[7] Philipp Schröer, Lutz Klinkenberg, and Leo Mommers. Probably. https://github.com/Philipp15b/probably. 2021.

[8] Herbert S Wilf. generatingfunctionology. CRC press, 2005.