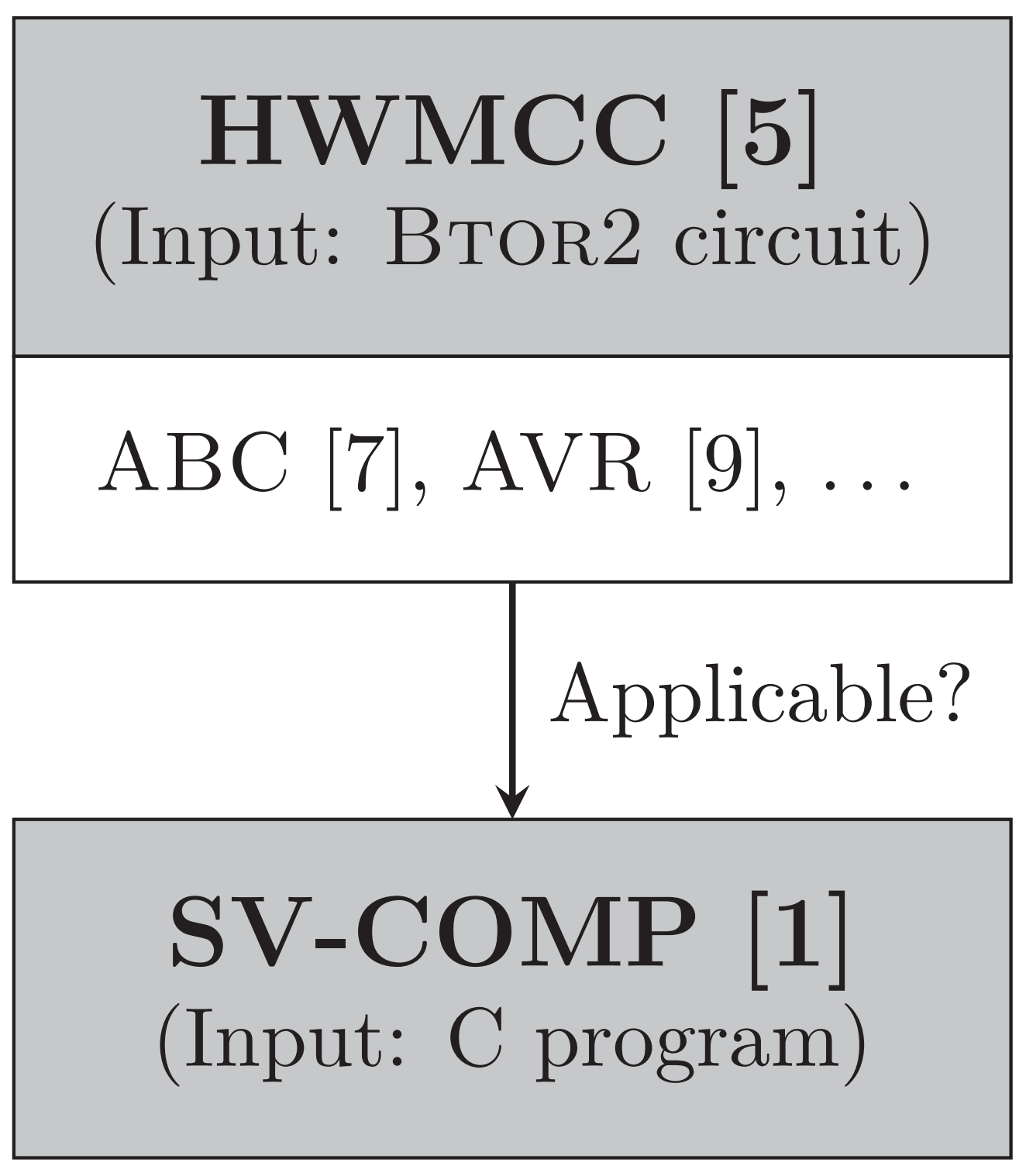
CPV: A Circuit-Based Program Verifier

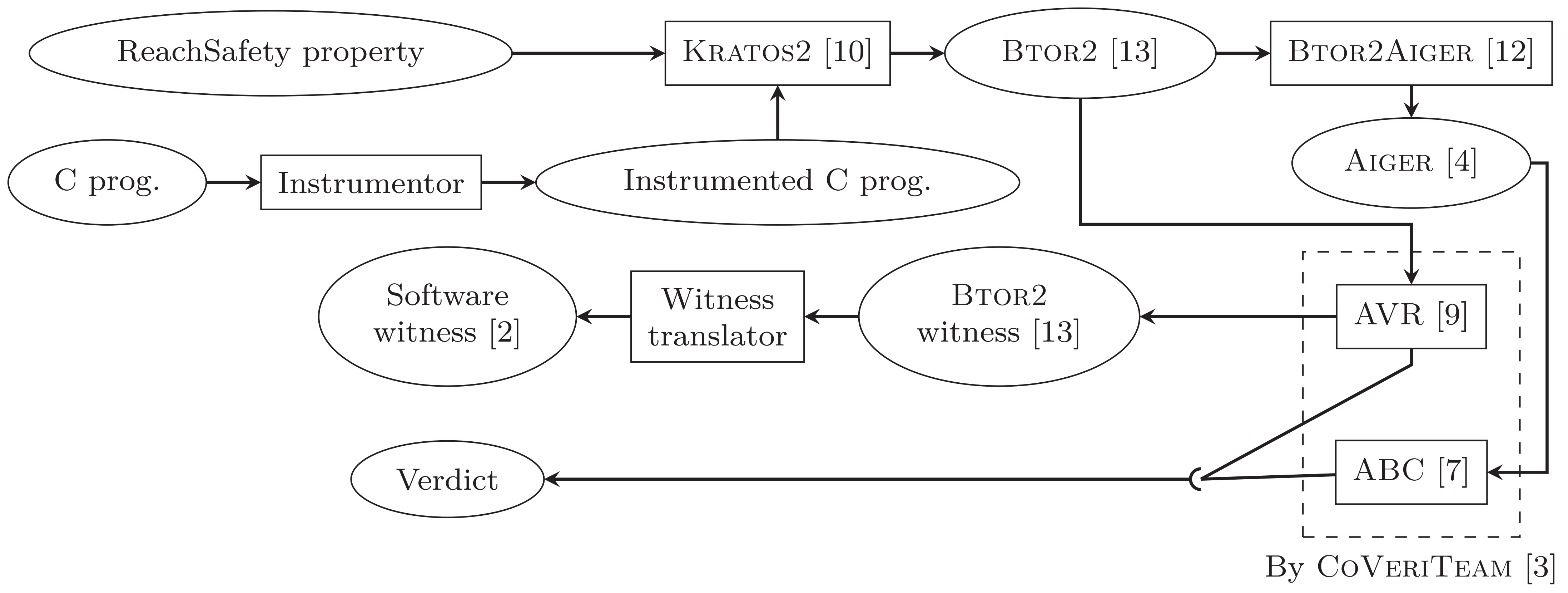
Po-Chun Chien and Nian-Ze Lee

{po-chun.chien, nian-ze.lee}@sosy.ifi.lmu.de

Motivation



Software Architecture

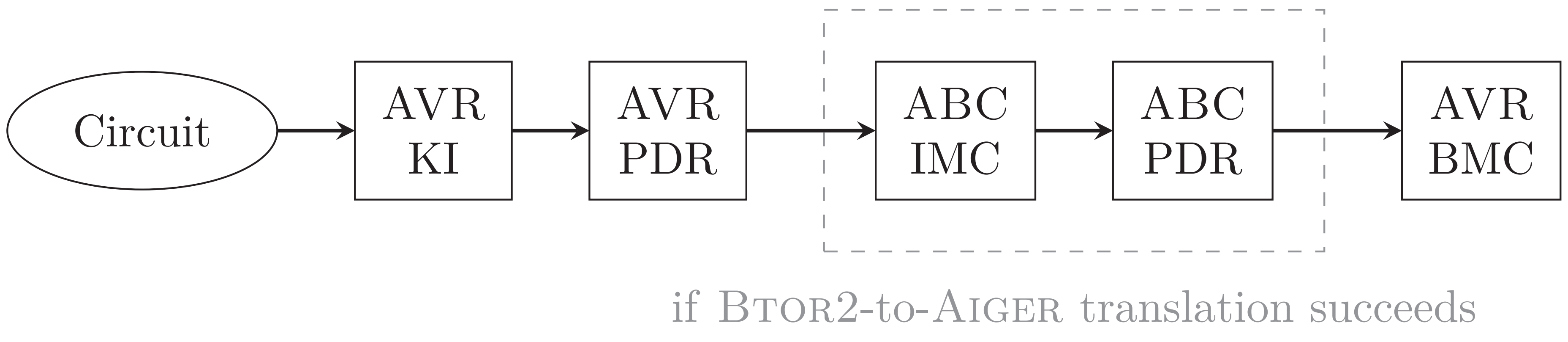


Try CPV!

Artifact DOI: `10.5281/zenodo.10063681`

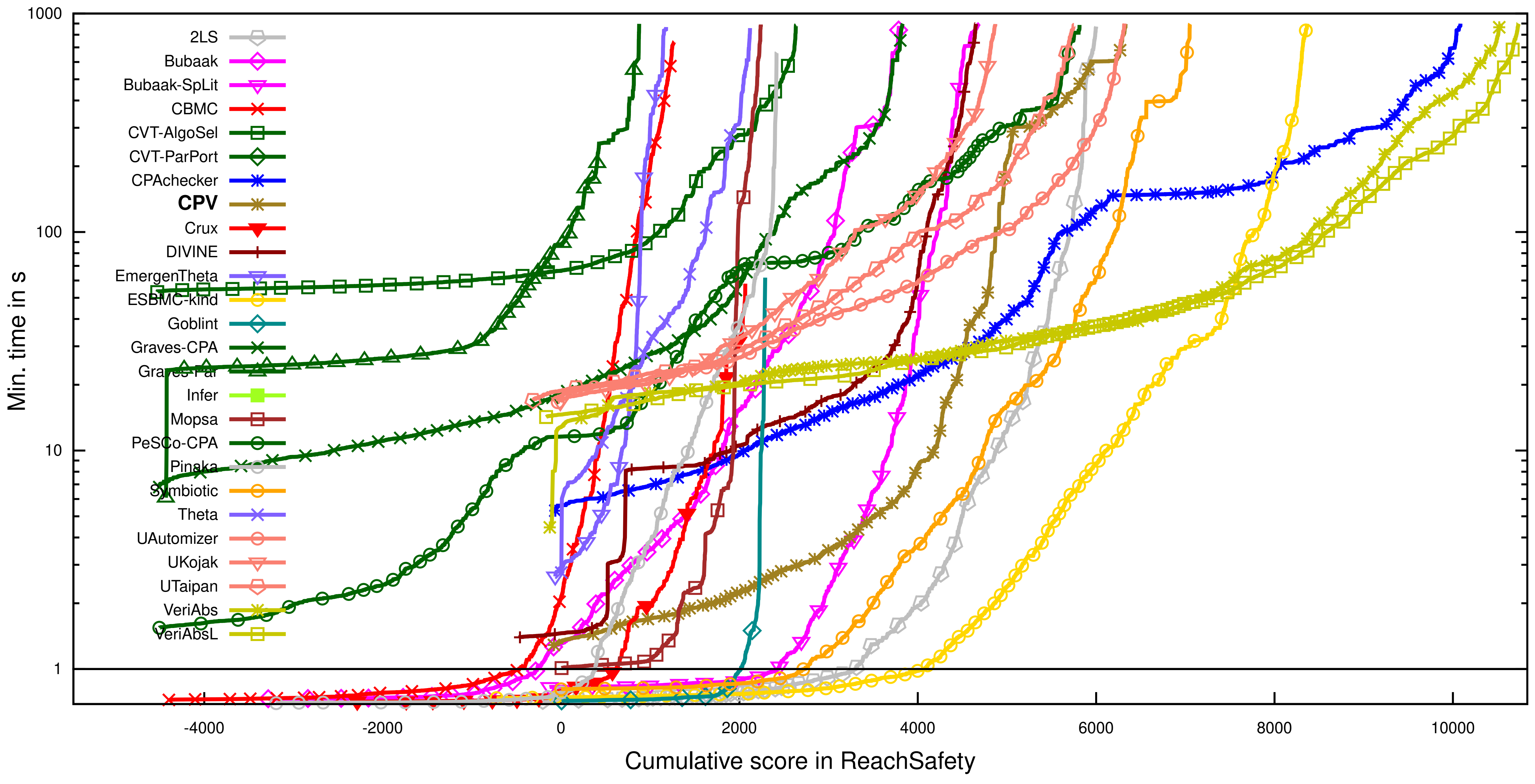
Strategy for SV-COMP 2024

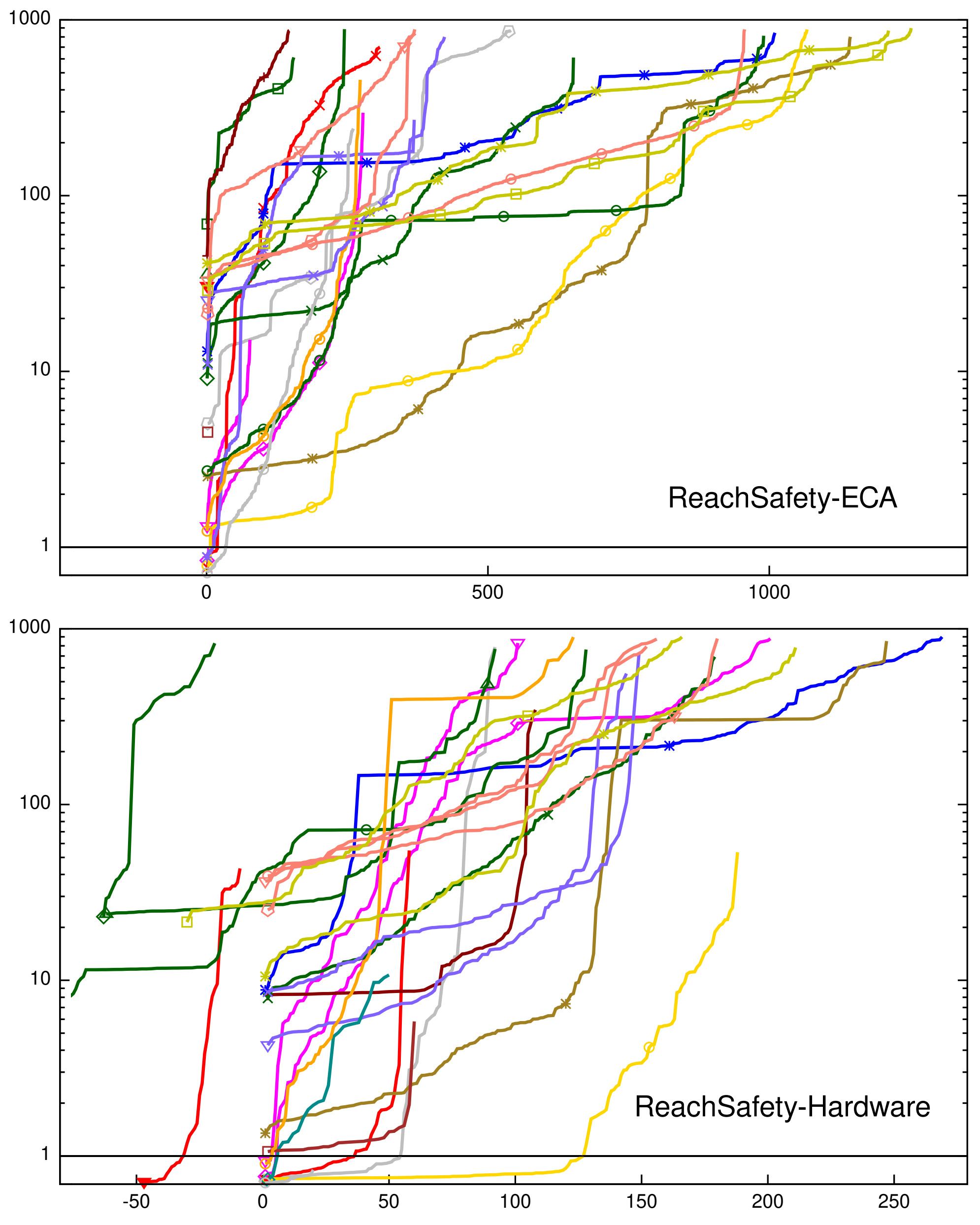
CPV runs a sequential portfolio consisting of property-directed reachability (PDR) [8], interpolation-based model checking (IMC) [11], k-induction (KI) [14], and bounded model checking (BMC) [6].



Evaluation Results at SV-COMP 2024

6th, 3rd, and 2nd place in *ReachSafety*, *ReachSafety-ECA*, *ReachSafety-Hardware*, respectively





Summary

- It is feasible to utilize sequential circuits as intermediate representations for software verification
- CPV can employ different hardware verifiers as the backend
- CPV competed well against other mature verifiers in SV-COMP
- Future work:
  - Support more verification properties (e.g., no-overflow and termination)
  - Export correctness witnesses
  - Incorporate more backend verifiers
  - Apply circuit optimization to improve the performance of verification

References

[1] Beyer, D.: State of the art in software verification and witness validation: SV-COMP 2024. In: Proc. TACAS (2024)

[2] Beyer, D., Dangl, M., Dietsch, D., Heizmann, M., Lemberger, T., Tautschnig, M.: Verification witnesses. ACM Trans. Softw. Eng. Methodol. **31**(4), 57:1–57:69 (2022)

[3] Beyer, D., Kanav, S.: CoVeriTeam: On-demand composition of cooperative verification systems. In: Proc. TACAS. pp. 561–579. LNCS 13243 (2022)

[4] Biere, A.: The AIGER And-Inverter Graph (AIG) format version 20071012. Tech. Rep. 07/1, Institute for Formal Models and Verification, Johannes Kepler University (2007)

[5] Biere, A., Froleyks, N., Preiner, M.: 11th Hardware Model Checking Competition (HWMCC 2020). `http://fmv.jku.at/hwmcc20/`, accessed: 2023-01-29

[6] Biere, A., Cimatti, A., Clarke, E.M., Strichman, O., Zhu, Y.: Bounded model checking. Advances in Computers **58**, 117–148 (2003)

[7] Brayton, R., Mishchenko, A.: ABC: An academic industrial-strength verification tool. In: Proc. CAV. pp. 24–40. LNCS 6174 (2010)

[8] Eén, N., Mishchenko, A., Brayton, R.K.: Efficient implementation of property directed reachability. In: Proc. FMCAD. pp. 125–134 (2011)

[9] Goel, A., Sakallah, K.: AVR: Abstractly verifying reachability. In: Proc. TACAS. pp. 413–422. LNCS 12078 (2020)

[10] Griggio, A., Jonáš, M.: Kratos2: An SMT-based model checker for imperative programs. In: Proc. CAV. pp. 423–436 (2023)

[11] McMillan, K.L.: Interpolation and SAT-based model checking. In: Proc. CAV. pp. 1–13. LNCS 2725 (2003)

[12] Niemetz, A., Preiner, M., Wolf, C., Biere, A.: Source-code repository of Btor2, BtorMC, and Boolector 3.0. `https://github.com/Boolector/btor2tools`, accessed: 2023-01-29

[13] Niemetz, A., Preiner, M., Wolf, C., Biere, A.: Btor2, BtorMC, and Boolector 3.0. In: Proc. CAV. pp. 587–595. LNCS 10981 (2018)

[14] Sheeran, M., Singh, S., Stålmarck, G.: Checking safety properties using induction and a SAT-solver. In: Proc. FMCAD, pp. 127–144. LNCS 1954 (2000)