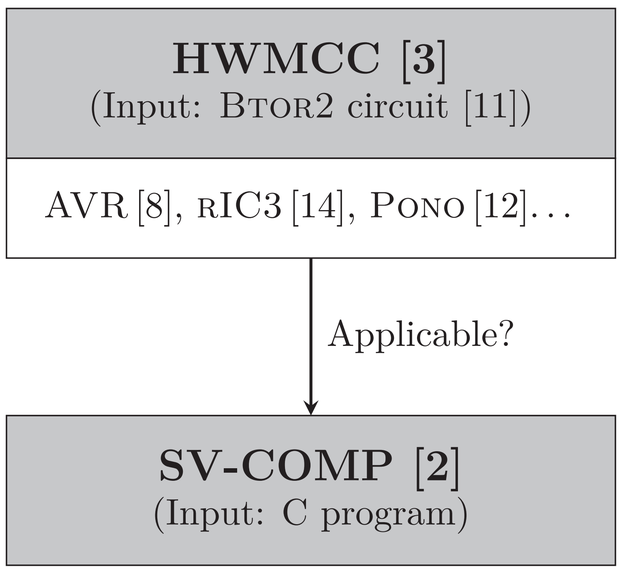
CPV: A Circuit-Based Program Verifier

Po-Chun Chien[1] **and Nian-Ze Lee**[2,1]

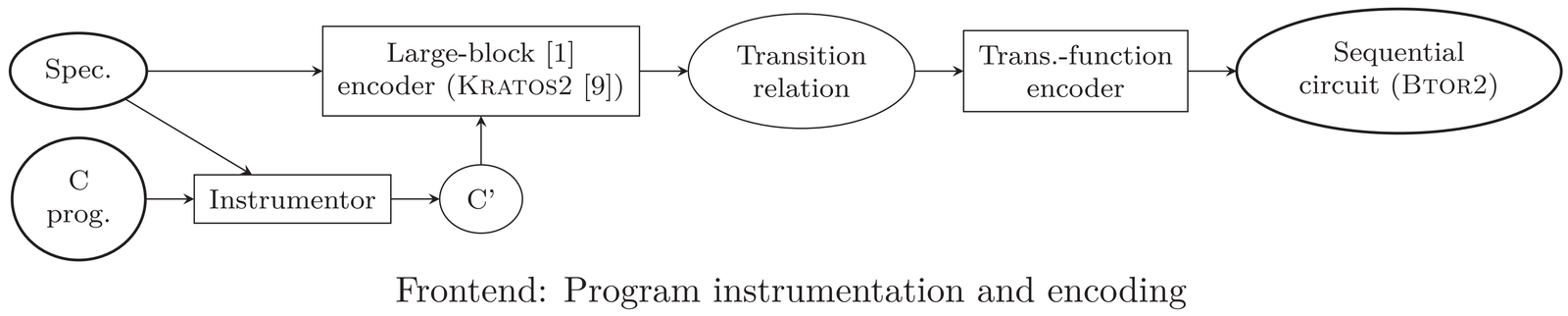
[1]LMU Munich [2]National Taiwan University

po-chun.chien@sosy.ifi.lmu.de nzlee@ntu.edu.tw

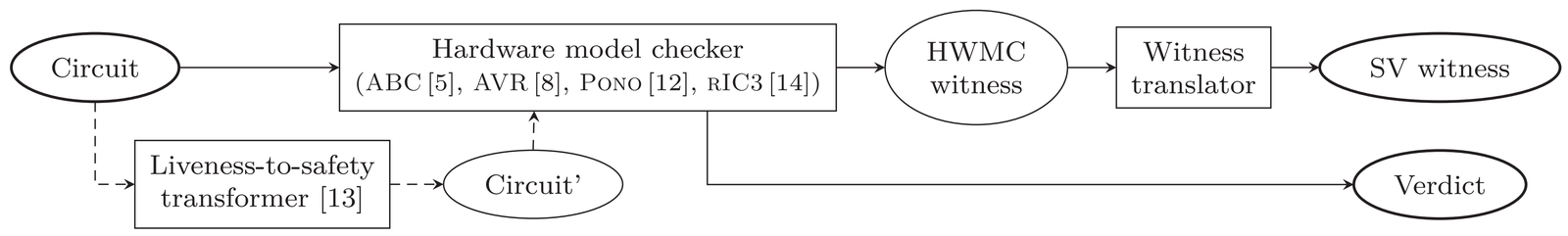
Motivation



CPV's Verification Pipeline



Frontend: Program instrumentation and encoding



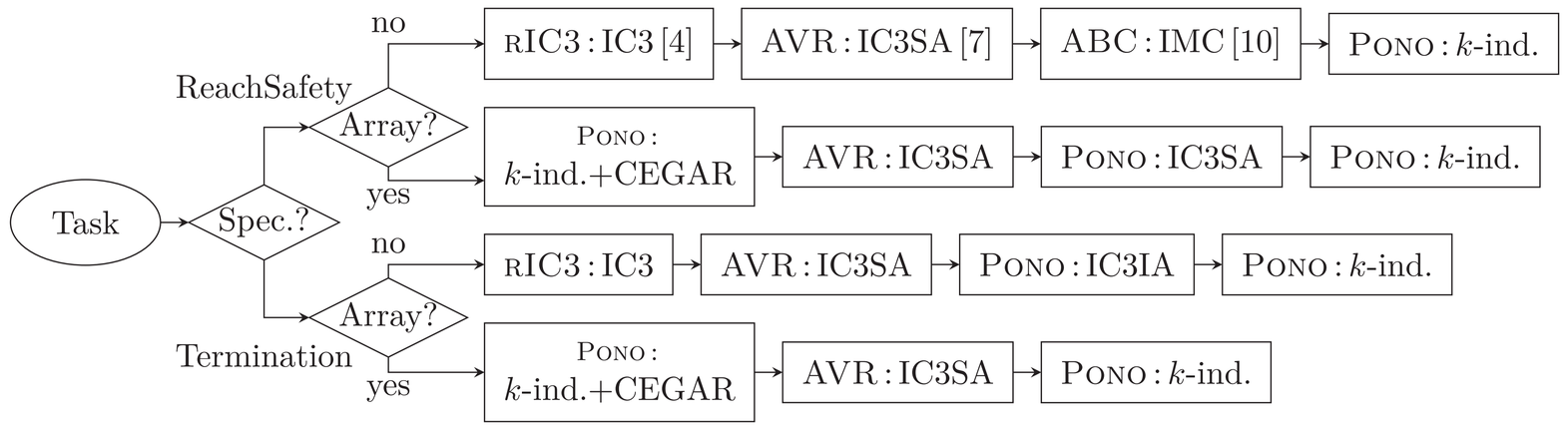
Backend: Model checking and witness translation

Try CPV!

gitlab.com/sosy-lab/software/cpv

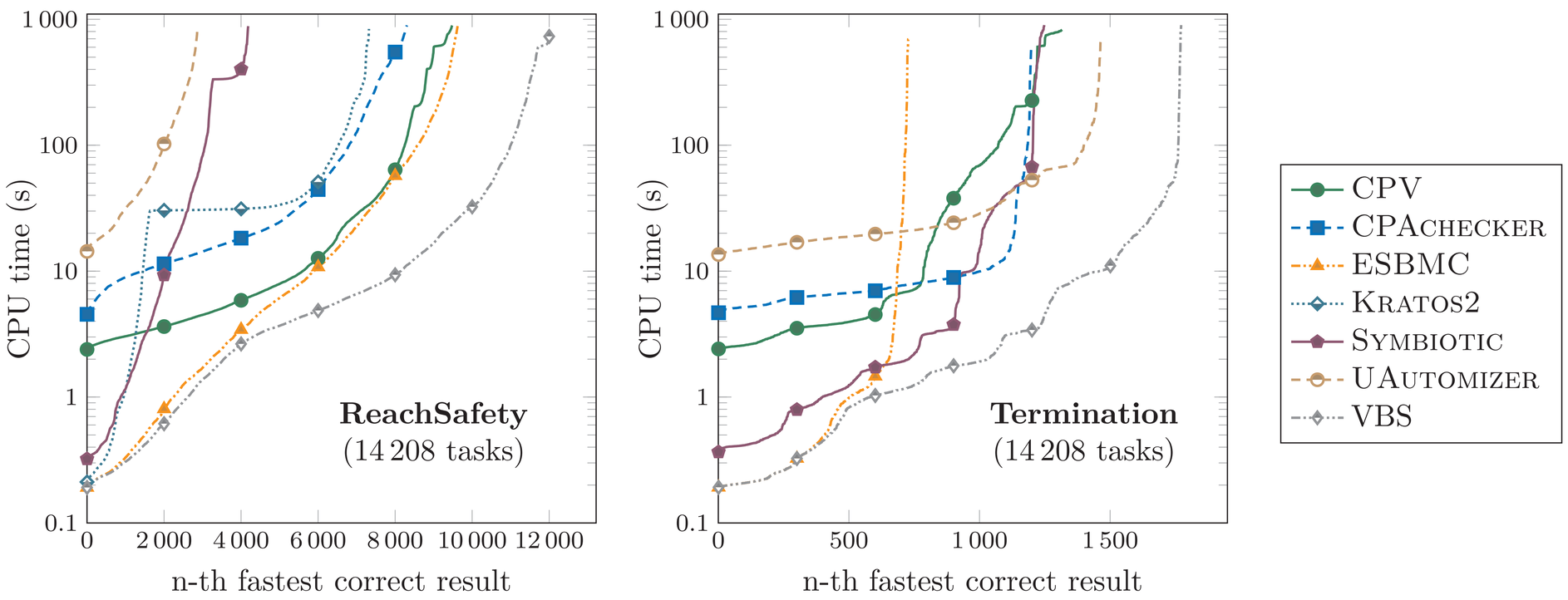
TACAS 2024 [6]

Strategy for SV-COMP 2026



Comparison with State-of-the-Art Software Verifiers

- Benchmark set: SV-COMP 2026 tasks without recursive functions
- Resource limits: 900 s CPU time and 15 GB memory
- Cactus plots show the number of solved tasks, without witness validation or subcategory weighting
- In the comparison, CPV
  - derived the **most proofs** for ReachSafety and
  - found the **most alarms** for Termination



Summary

- Sequential circuits can serve as an intermediate representation for software verification
- Offer various instrumentation, encoding and transformation options
- Leverage powerful word- and bit-level hardware model checkers as backend
- Perform competitively against state-of-the-art verifiers in SV-COMP
- Export violation witnesses in GraphML (v1) and YAML (v2) formats
- Next step: Produce correctness witnesses by translating invariants derived by hardware model checkers

References

[1] Beyer, D., Cimatti, A., Griggio, A., Keremoglu, M.E., Sebastiani, R.: Software model checking via large-block encoding. In: Proc. FMCAD. pp. 25–32 (2009)

[2] Beyer, D., Strejček, J.: Evaluating software verifiers for C, Java, and SV-LIB (report on SV-COMP 2026). In: Proc. TACAS (2). LNCS 16506 (2026)

[3] Biere, A., Froleyks, N., Preiner, M.: Hardware model checking competition 2025. In: Proc. FMCAD. p. 7 (2025)

[4] Bradley, A.R.: SAT-based model checking without unrolling. In: Proc. VMCAI. pp. 70–87. LNCS 6538 (2011)

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

[6] Chien, P.C., Lee, N.Z.: CPV: A circuit-based program verifier (competition contribution). In: Proc. TACAS (3). pp. 365–370. LNCS 14572 (2024)

[7] Goel, A., Sakallah, K.: Model checking of Verilog RTL using IC3 with syntax-guided abstraction. In: Proc. NFM. pp. 166–185 (2019)

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

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

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

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

[12] Perez-Lopez, A.R., Chien, P.C., Lonsing, F., Archer, S., Irfan, A., Barrett, C.: Pono 2.0: A versatile SMT-based model checker for safety and liveness (long tool paper). In: Proc. FM. LNCS (2026)

[13] Schuppan, V., Biere, A.: Liveness checking as safety checking for infinite state spaces. Electr. Notes Theor. Comput. Sci. **149**(1), 79–96 (2006)

[14] Su, Y., Yang, Q., Ci, Y., Bu, T., Huang, Z.: The rIC3 hardware model checker. In: Proc. CAV (1). pp. 185–199. LNCS 15931 (2025)