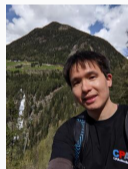


# CPV: A Circuit-Based Program Verifier

---

Po-Chun Chien and Nian-Ze Lee



SV-COMP 2024 @ Luxembourg  
LMU Munich, Germany

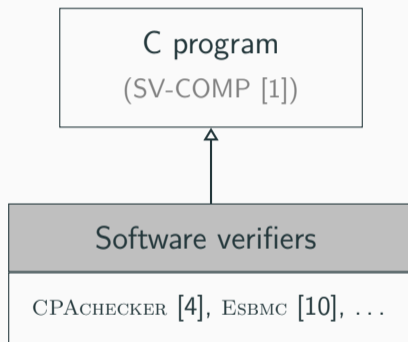
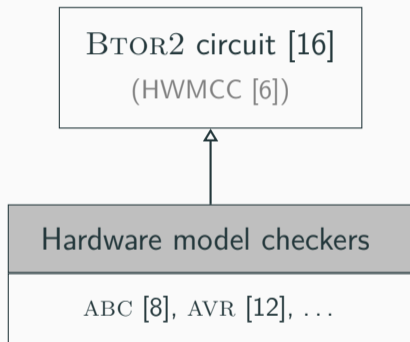


# Motivation

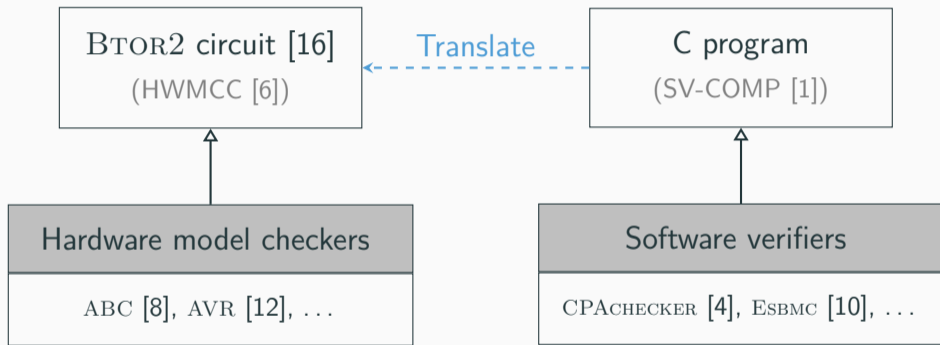
BTOR2 circuit [16]  
(HWMCC [6])

C program  
(SV-COMP [1])

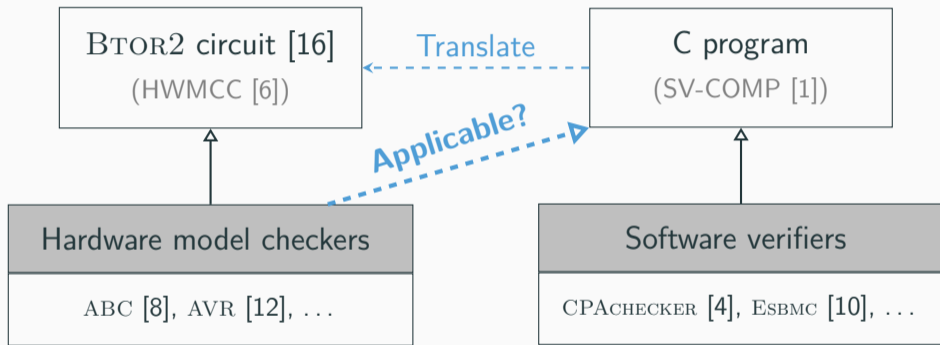
# Motivation



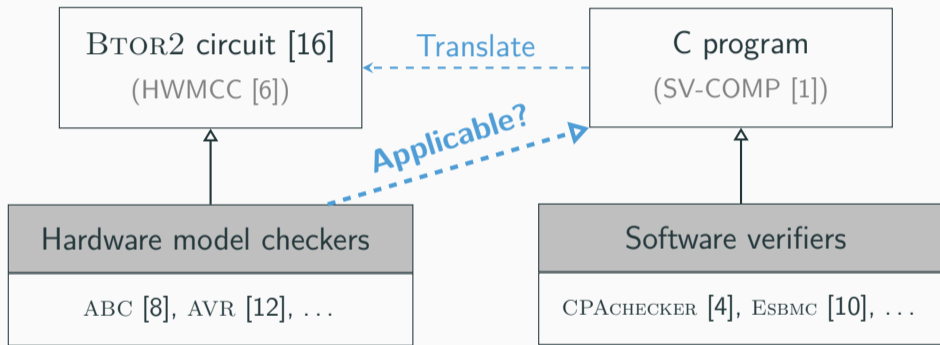
# Motivation



# Motivation

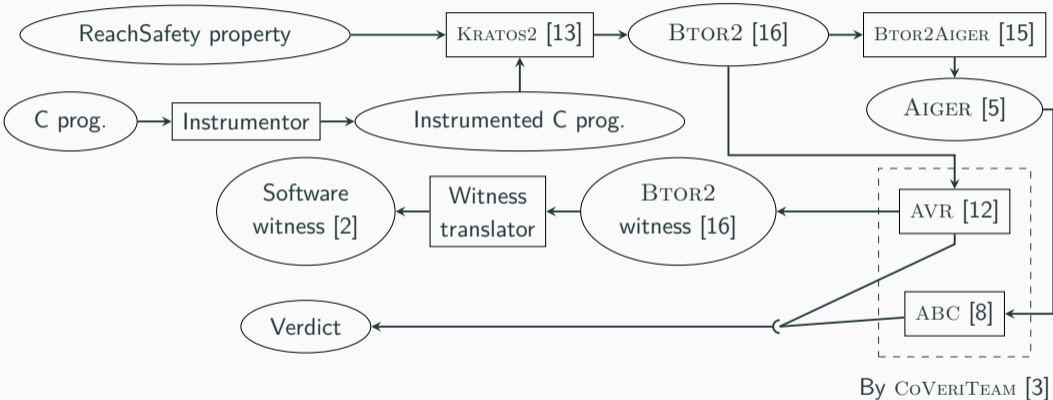


# Motivation

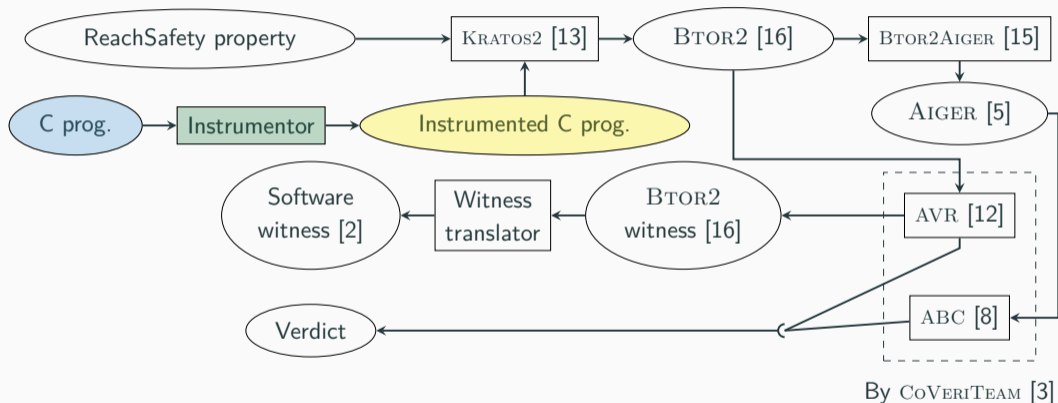


**Could circuits serve as IR for program verification?**

# System Architecture



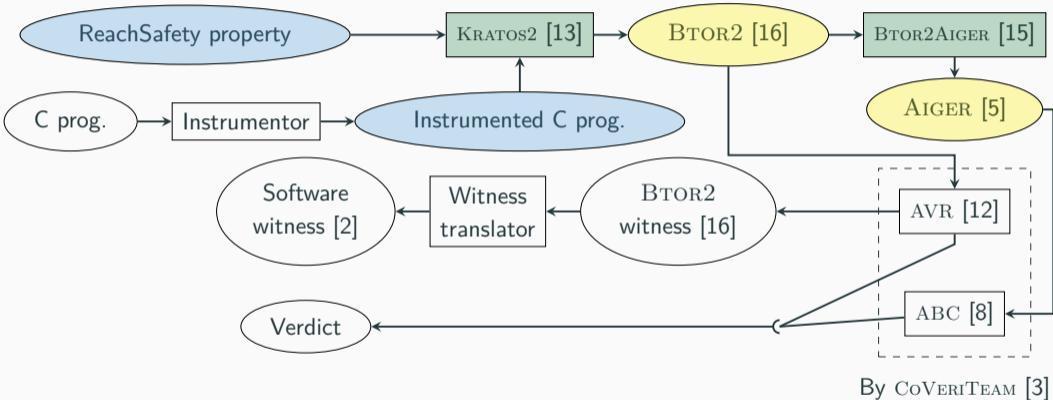
# System Architecture



## 1. Instrument the input program

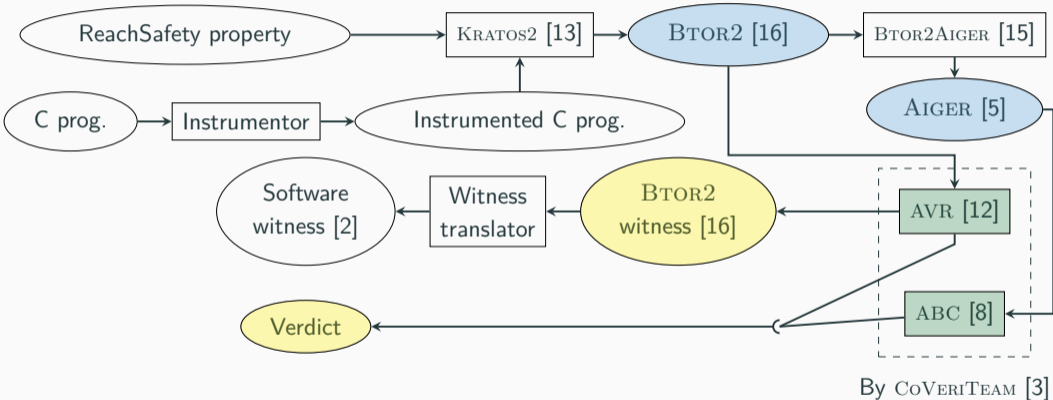


# System Architecture



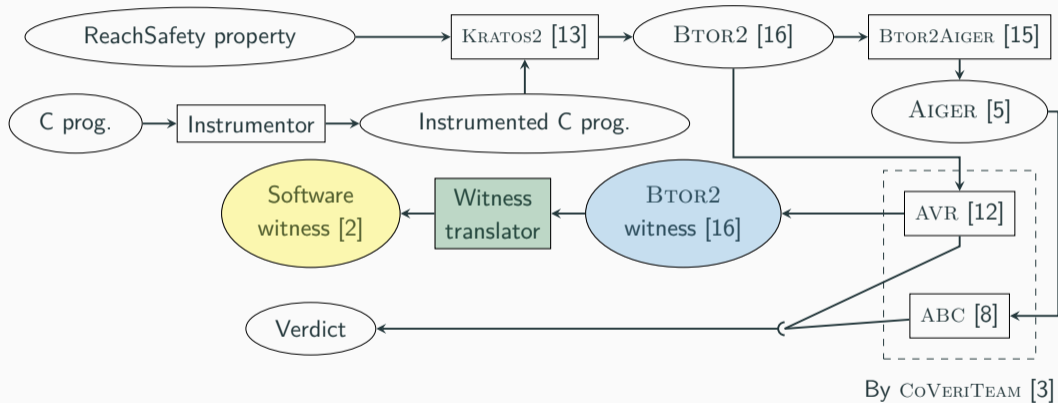
## 2. Translate the program to a circuit

# System Architecture



3. Verify the translated circuit with hardware model checkers

# System Architecture



4. Translate the BTOR2 witness back to software domain

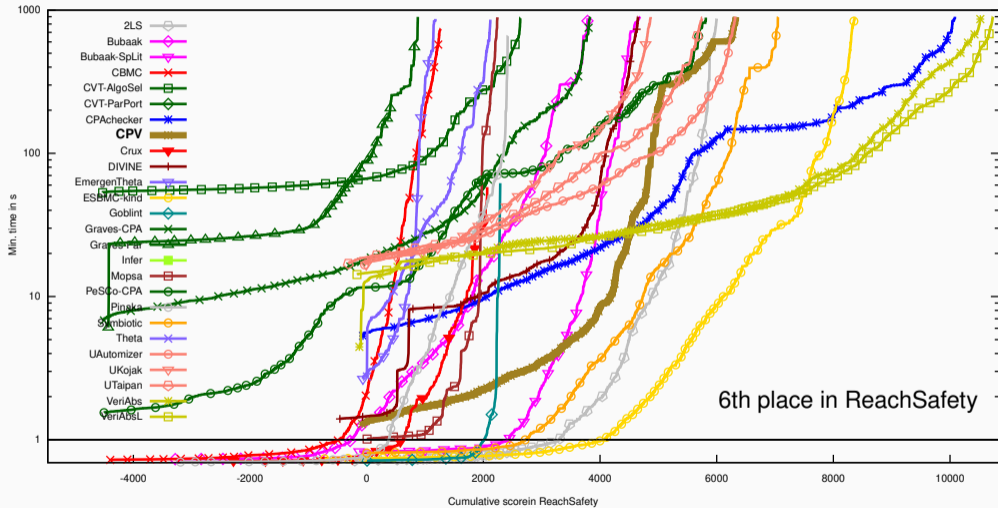
# Strategy for SV-COMP

A sequential portfolio consisting of:

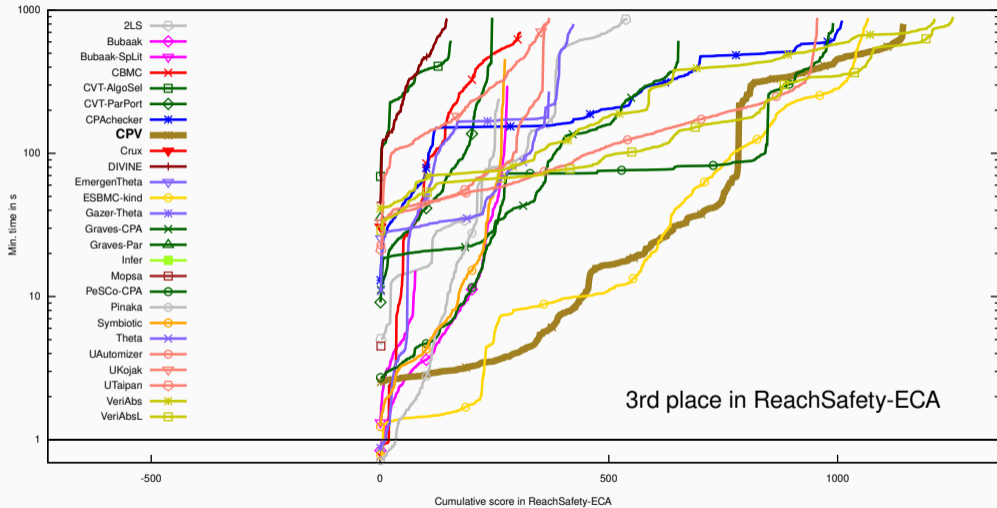


if BTOR2-to-AIGER translation succeeds

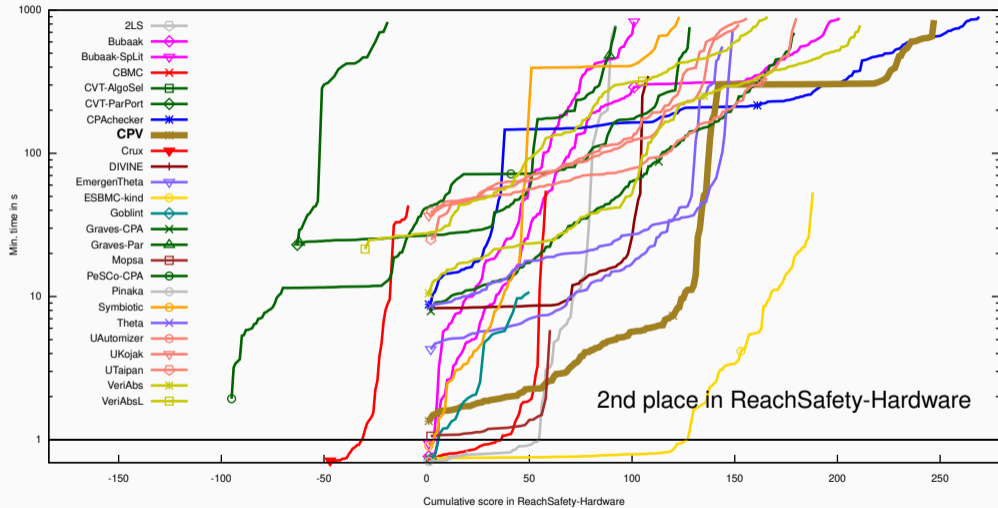
# Results in SV-COMP



# Results in SV-COMP




# Results in SV-COMP



# Conclusion

- Our new tool CPV uses
  - circuits as IR and
  - hardware verifiers as backend.
- Good performance as a 1st time participant in SV-COMP!



 [gitlab.com/  
sosy-lab/software/cpv](https://gitlab.com/sosy-lab/software/cpv)



# References i

- [1] Beyer, D.: State of the art in software verification and witness validation: SV-COMP 2024. In: Proc. TACAS. pp. 299–329. LNCS 14572, Springer (2024). [https://doi.org/10.1007/978-3-031-57256-2\\_15](https://doi.org/10.1007/978-3-031-57256-2_15)
- [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). <https://doi.org/10.1145/3477579>
- [3] Beyer, D., Kanav, S.: COVERITEAM: On-demand composition of cooperative verification systems. In: Proc. TACAS. pp. 561–579. LNCS 13243, Springer (2022). [https://doi.org/10.1007/978-3-030-99524-9\\_31](https://doi.org/10.1007/978-3-030-99524-9_31)
- [4] Beyer, D., Keremoglu, M.E.: CPACHECKER: A tool for configurable software verification. In: Proc. CAV. pp. 184–190. LNCS 6806, Springer (2011). [https://doi.org/10.1007/978-3-642-22110-1\\_16](https://doi.org/10.1007/978-3-642-22110-1_16)
- [5] 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). <https://doi.org/10.35011/fmvtr.2007-1>

## References ii

- [6] Biere, A., Froylyks, N., Preiner, M.: 11th Hardware Model Checking Competition (HWMCC 2020). <http://fmv.jku.at/hwmcc20/>, accessed: 2023-01-29
- [7] Biere, A., Cimatti, A., Clarke, E.M., Strichman, O., Zhu, Y.: Bounded model checking. *Advances in Computers* **58**, 117–148 (2003). [https://doi.org/10.1016/S0065-2458\(03\)58003-2](https://doi.org/10.1016/S0065-2458(03)58003-2)
- [8] Brayton, R., Mishchenko, A.: ABC: An academic industrial-strength verification tool. In: *Proc. CAV*. pp. 24–40. LNCS 6174, Springer (2010). [https://doi.org/10.1007/978-3-642-14295-6\\_5](https://doi.org/10.1007/978-3-642-14295-6_5)
- [9] Eén, N., Mishchenko, A., Brayton, R.K.: Efficient implementation of property directed reachability. In: *Proc. FMCAD*. pp. 125–134. FMCAD Inc. (2011). <https://dl.acm.org/doi/10.5555/2157654.2157675>
- [10] Gadelha, M.R., Monteiro, F.R., Morse, J., Cordeiro, L.C., Fischer, B., Nicole, D.A.: ESBMC 5.0: An industrial-strength C model checker. In: *Proc. ASE*. pp. 888–891. ACM (2018). <https://doi.org/10.1145/3238147.3240481>
- [11] Goel, A., Sakallah, K.: Model checking of Verilog RTL using IC3 with syntax-guided abstraction. In: *Proc. NFM*. pp. 166–185. Springer (2019). [https://doi.org/10.1007/978-3-030-20652-9\\_11](https://doi.org/10.1007/978-3-030-20652-9_11)

## References iii

- [12] Goel, A., Sakallah, K.: AVR: Abstractly verifying reachability. In: Proc. TACAS. pp. 413–422. LNCS 12078, Springer (2020). [https://doi.org/10.1007/978-3-030-45190-5\\_23](https://doi.org/10.1007/978-3-030-45190-5_23)
- [13] Griggio, A., Jonáš, M.: KRATOS2: An SMT-based model checker for imperative programs. In: Proc. CAV. pp. 423–436. Springer (2023). [https://doi.org/10.1007/978-3-031-37709-9\\_20](https://doi.org/10.1007/978-3-031-37709-9_20)
- [14] McMillan, K.L.: Interpolation and SAT-based model checking. In: Proc. CAV. pp. 1–13. LNCS 2725, Springer (2003). [https://doi.org/10.1007/978-3-540-45069-6\\_1](https://doi.org/10.1007/978-3-540-45069-6_1)
- [15] 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
- [16] Niemetz, A., Preiner, M., Wolf, C., Biere, A.: BTOR2, BTORMC, and BOOLECTOR 3.0. In: Proc. CAV. pp. 587–595. LNCS 10981, Springer (2018). [https://doi.org/10.1007/978-3-319-96145-3\\_32](https://doi.org/10.1007/978-3-319-96145-3_32)
- [17] Sheeran, M., Singh, S., Stålmarck, G.: Checking safety properties using induction and a SAT-solver. In: Proc. FMCAD, pp. 127–144. LNCS 1954, Springer (2000). [https://doi.org/10.1007/3-540-40922-X\\_8](https://doi.org/10.1007/3-540-40922-X_8)