

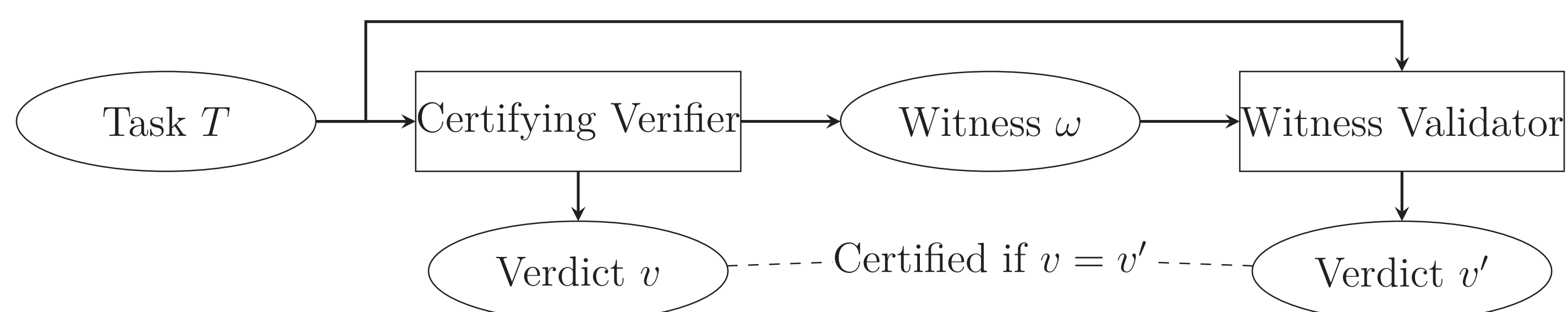


Zsófia Ádám, Dirk Beyer, Po-Chun Chien,
Nian-Ze Lee, and Nils Sirrenberg

adamzsofi@edu.bme.hu, nils.sirrenberg@campus.lmu.de,
{dirk.beyer,po-chun.chien,nian-ze.lee}@sosy.ifi.lmu.de

Presentation at TACAS 2024: 12:00, Thursday, April 11, Room: TBD

CERTIFYING AND VALIDATING VERIFICATION



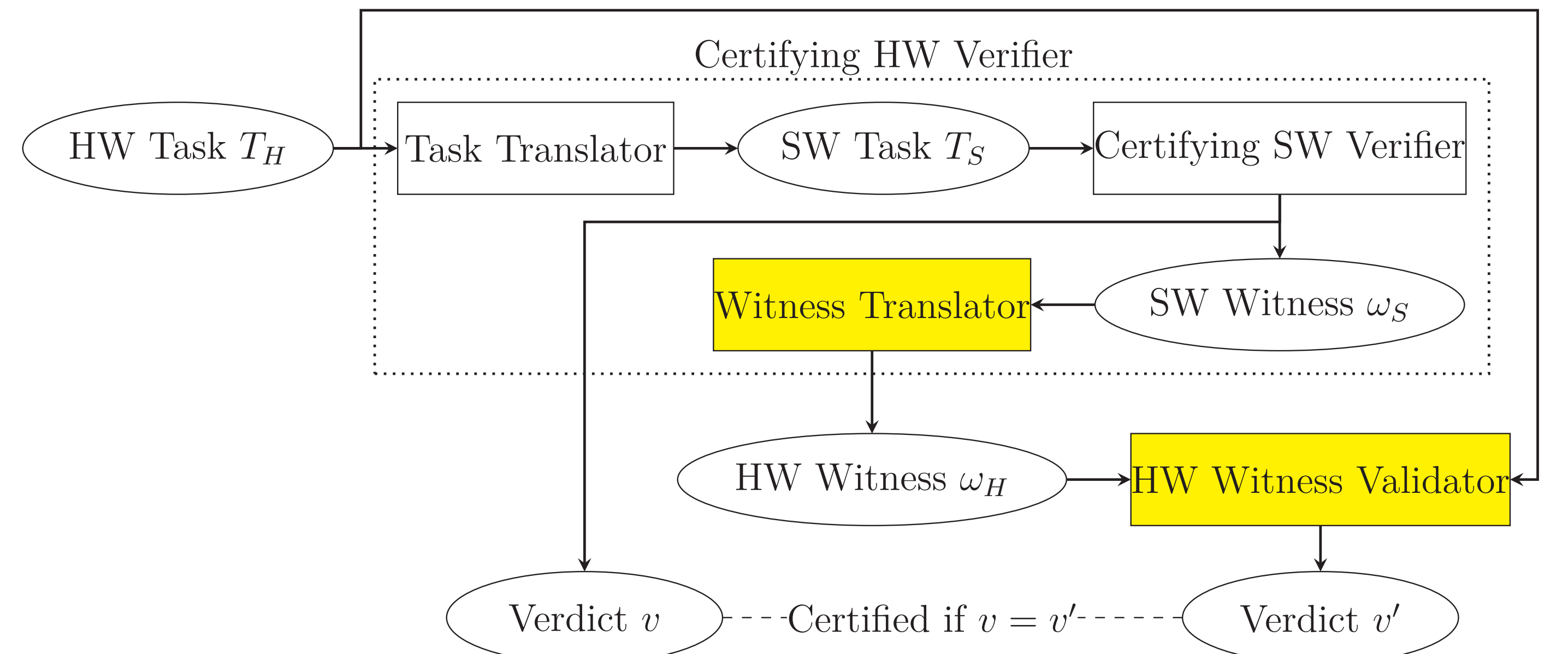
Our Motivation

- Explainable and trustworthy HW verification (HV)
- SW verification (SV) techniques for HW

Our Contributions

- A certifying HV framework using SV techniques
- A translator from SW witnesses to HW witnesses
- A witness validator for the BTOR2 HW modeling language [6]
- Complementing HV with certified results from SV

CERTIFYING HV USING TRANSLATION AND SV



BTOR2-CERT instantiates the framework with BTOR2C [1] as frontend and SW verifiers that export GraphML witnesses [2] as backend.

HW-TO-SW TRANSLATION VIA BTOR2C [1]

```

1 sort bitvec 8
2 sort bitvec 1
3 constd 1 42
4 constd 1 2
5 zero 1
6 state 1 ; a
7 state 1 ; b
8 input 1 ; in
9 init 1 6 4 ; a init to 2
10 init 1 7 5 ; b init to 0
11 eq 2 6 5 ; a == 0
12 eq 2 7 4 ; b == 2
13 eq 2 8 3 ; in == 42
14 and 2 11 12
15 and 2 13 14
16 bad 15
17 one 1
18 srl 1 6 17
19 xor 1 7 17
20 next 1 6 18
21 next 1 7 19

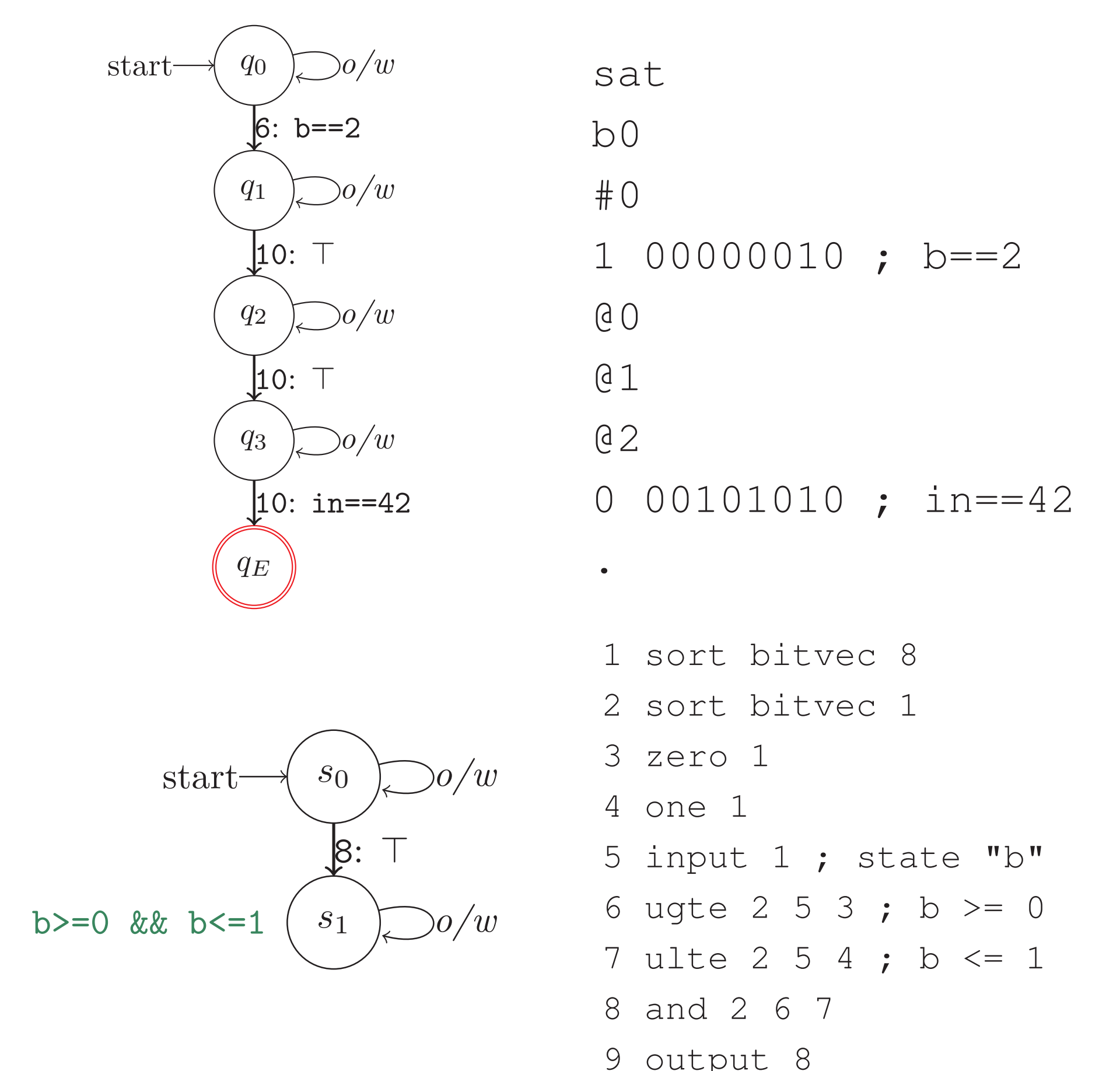
```

```

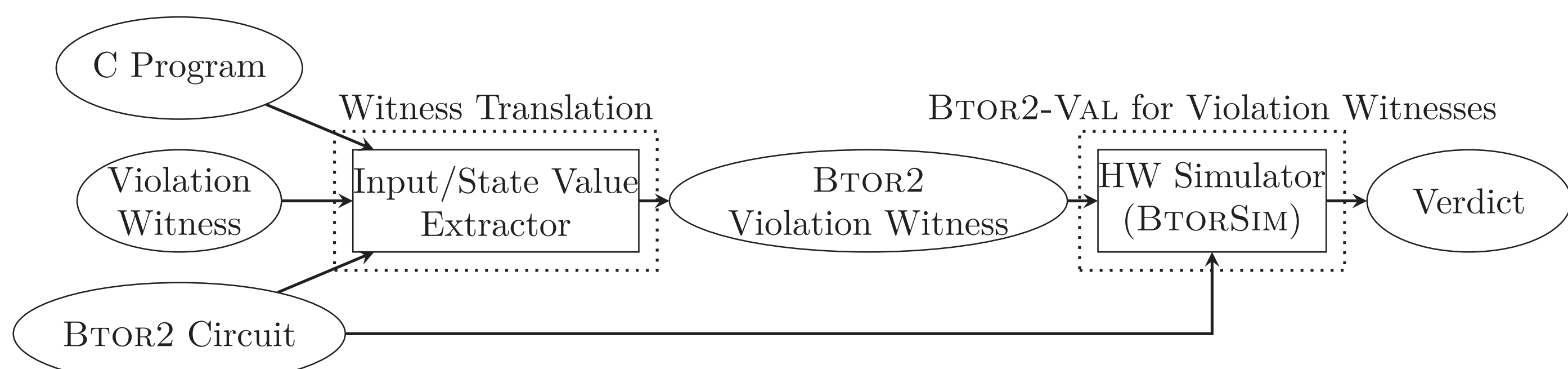
1 extern void abort(void);
2 extern unsigned char nondet_uchar();
3 void main() {
4     typedef unsigned char SORT_1;
5     SORT_1 a = nondet_uchar();
6     SORT_1 b = nondet_uchar();
7     a = 2;
8     b = 0; // Omit for unsafe version
9     for (;;) {
10        SORT_1 in = nondet_uchar();
11        if (a == 0 && b == 2 && in == 42) {
12            ERROR:
13                abort();
14        }
15        a = a >> 1;
16        b = b ^ 1;
17    }
18 }

```

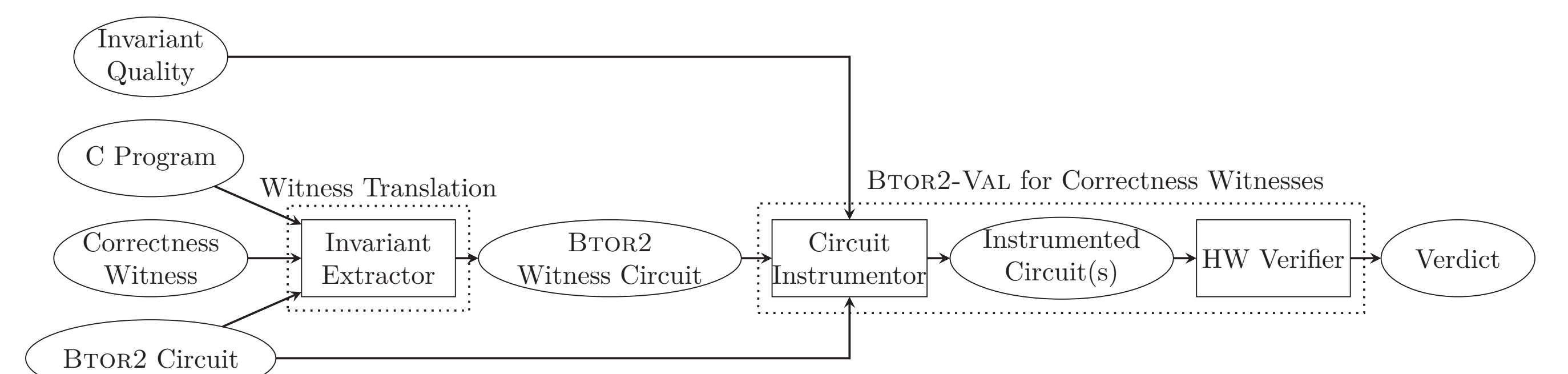
WITNESS TRANSLATION



VIOLATION WITNESS VALIDATION



CORRECTNESS WITNESS VALIDATION



SUMMARY OF EXPERIMENTAL RESULTS

On 758 safe and 456 unsafe BTOR2 verification tasks, **BTOR2-CERT** achieved:

- Translation of all violation and 97 % correctness witnesses,
- Effective and efficient validation vs. compared validators, e.g., LIV [4] and CPA-w2T [3], and
- Certified bugs in 8 % of the unsafe tasks with CBMC [5] that HV overlooked

INVARIANT QUALITY

Three user-defined quality levels for invariants:

- Invariant (containing all reachable states)
- Safe invariant (implying safety property)
- Safe and inductive invariant

REFERENCES

- [1] Beyer, D., Chien, P.C., Lee, N.Z.: Bridging hardware and software analysis with BTOR2C: A word-level-circuit-to-C translator. In: Proc. TACAS. pp. 1–21. LNCS 13994 (2023)
- [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., Dangl, M., Lemberger, T., Tautschnig, M.: Tests from witnesses: Execution-based validation of verification results. In: Proc. TAP. pp. 3–23. LNCS 10889 (2018)
- [4] Beyer, D., Spiessl, M.: LIV: A loop-invariant validation using straight-line programs. In: Proc. ASE. pp. 2074–2077 (2023)
- [5] Clarke, E.M., Kröning, D., Lerda, F.: A tool for checking ANSI-C programs. In: Proc. TACAS. pp. 168–176. LNCS 2988 (2004)
- [6] Niemetz, A., Preiner, M., Wolf, C., Biere, A.: BTOR2, BTORMC, and BOLECTOR 3.0. In: Proc. CAV. pp. 587–595. LNCS 10981 (2018)

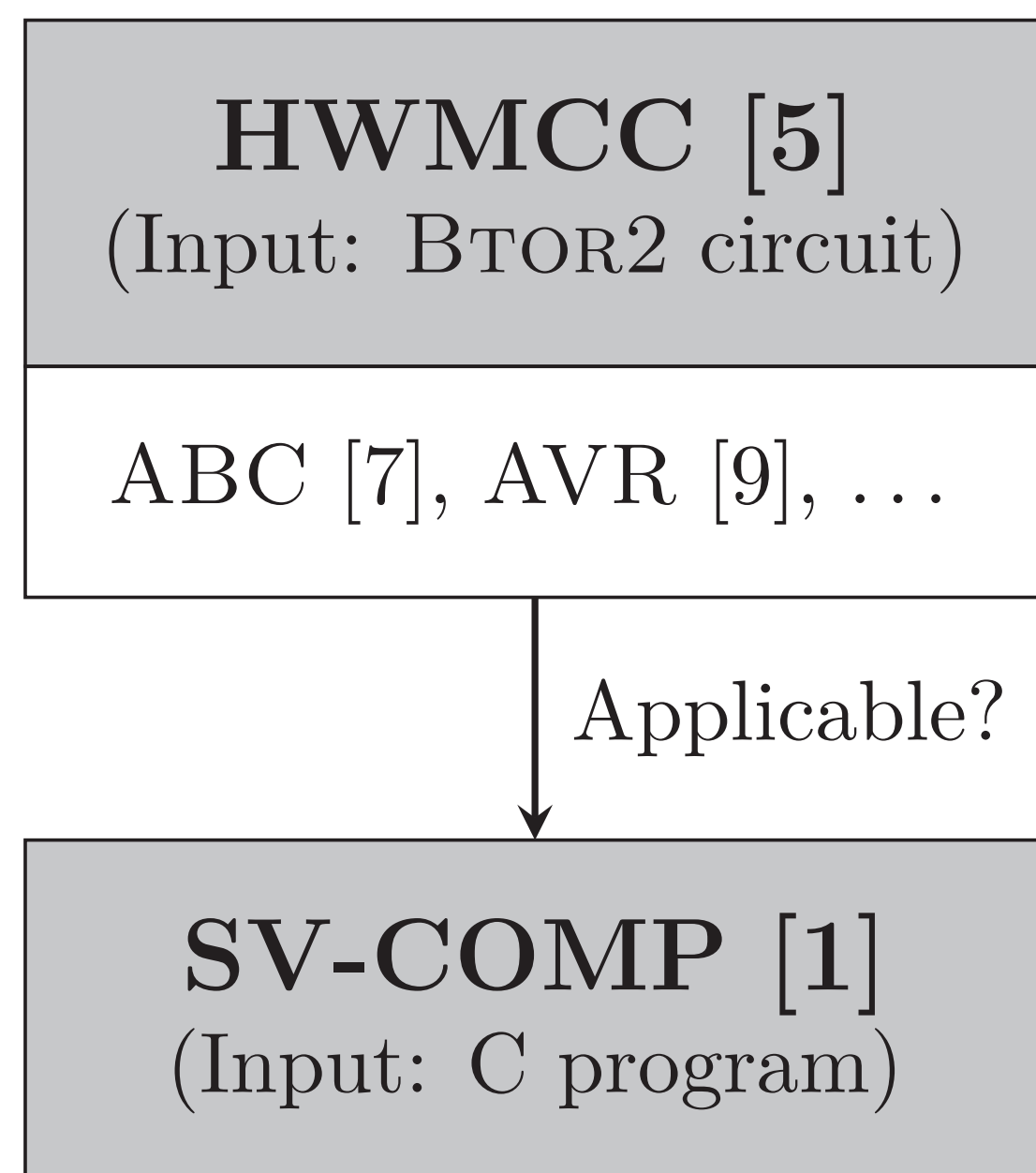
TRY BTOR2-CERT!



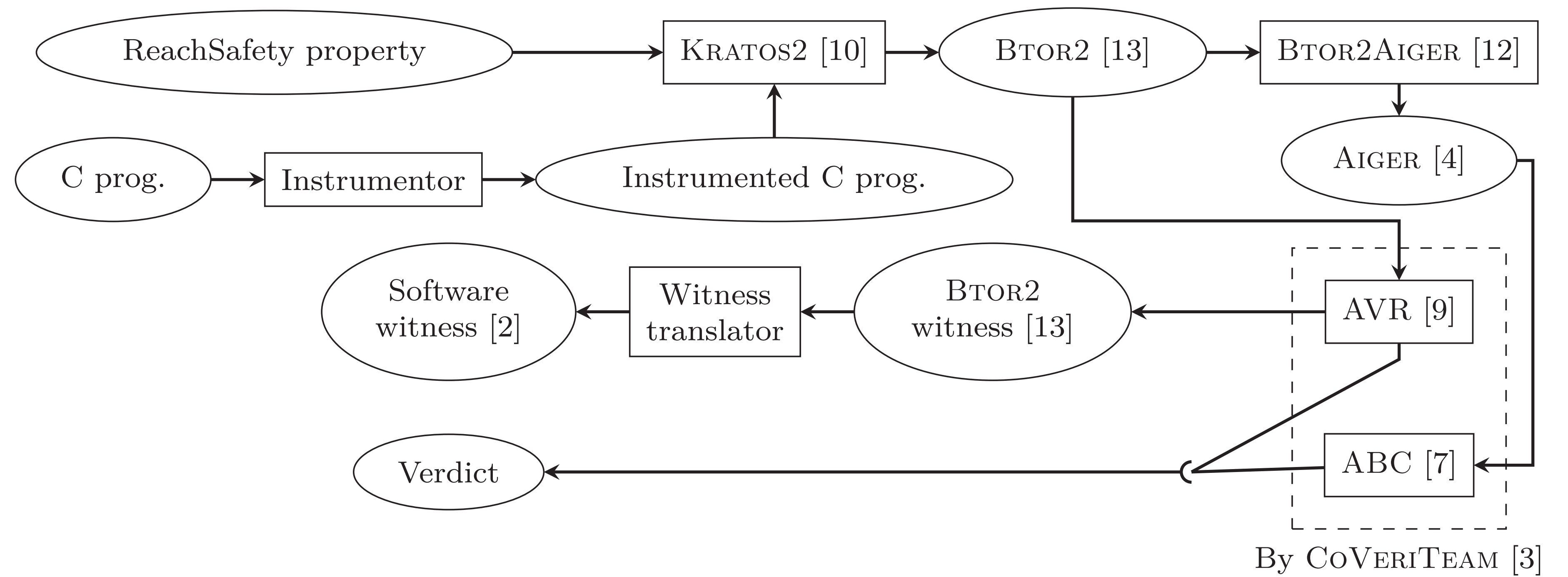
Artifact DOI: 10.5281/zenodo.10548597



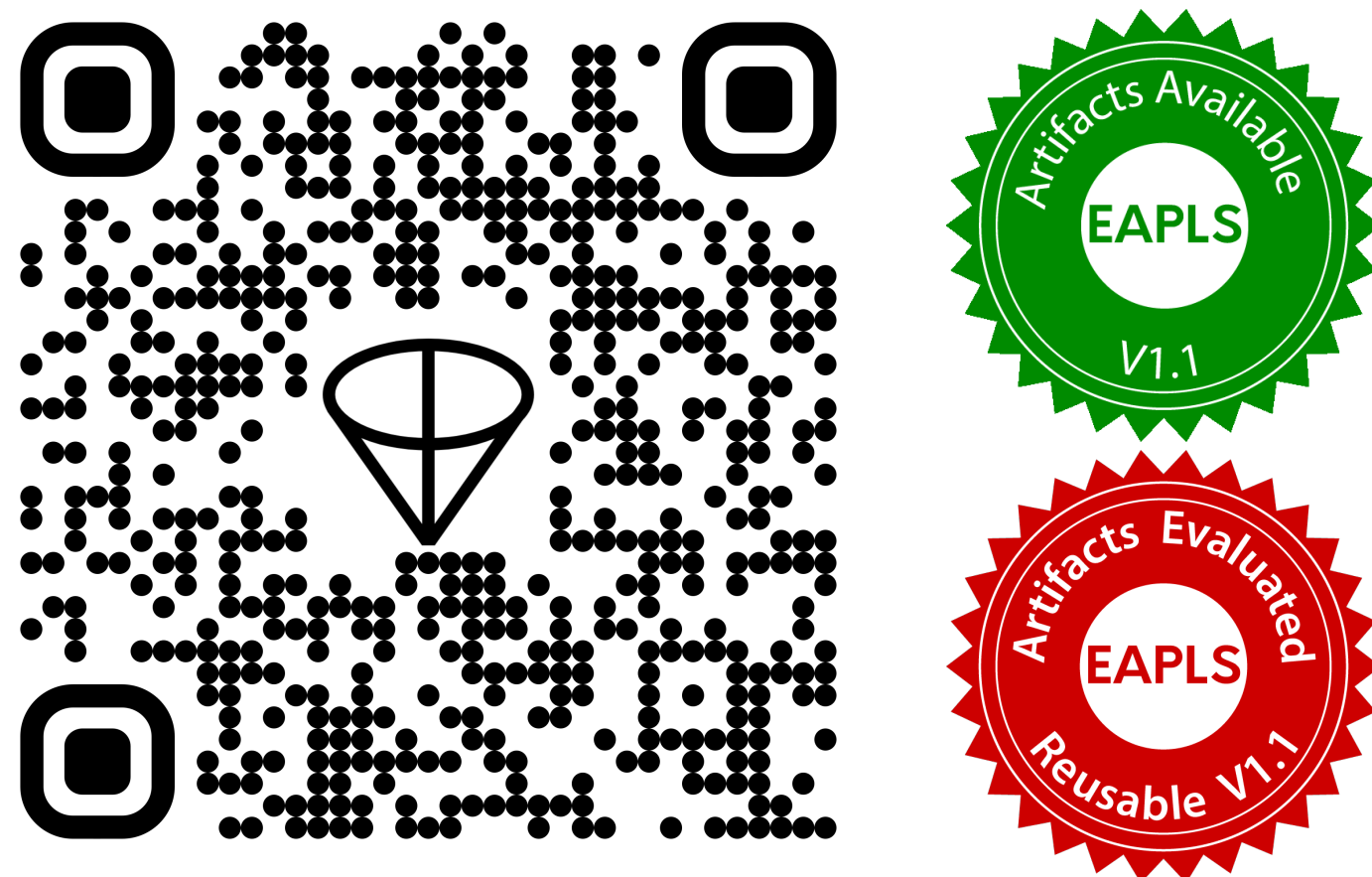
MOTIVATION



SOFTWARE ARCHITECTURE



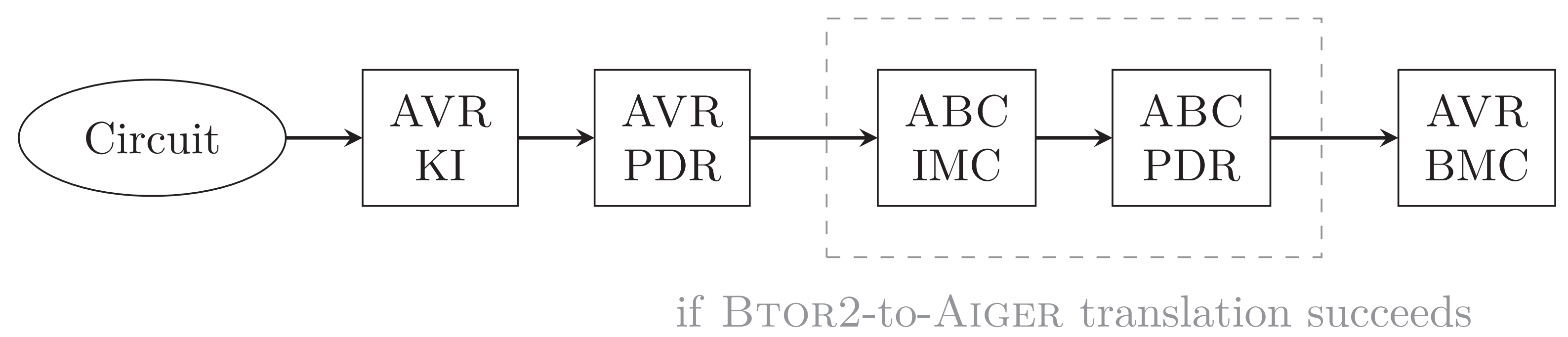
TRY CPV!



Artifact DOI: 10.5281/zenodo.10063681

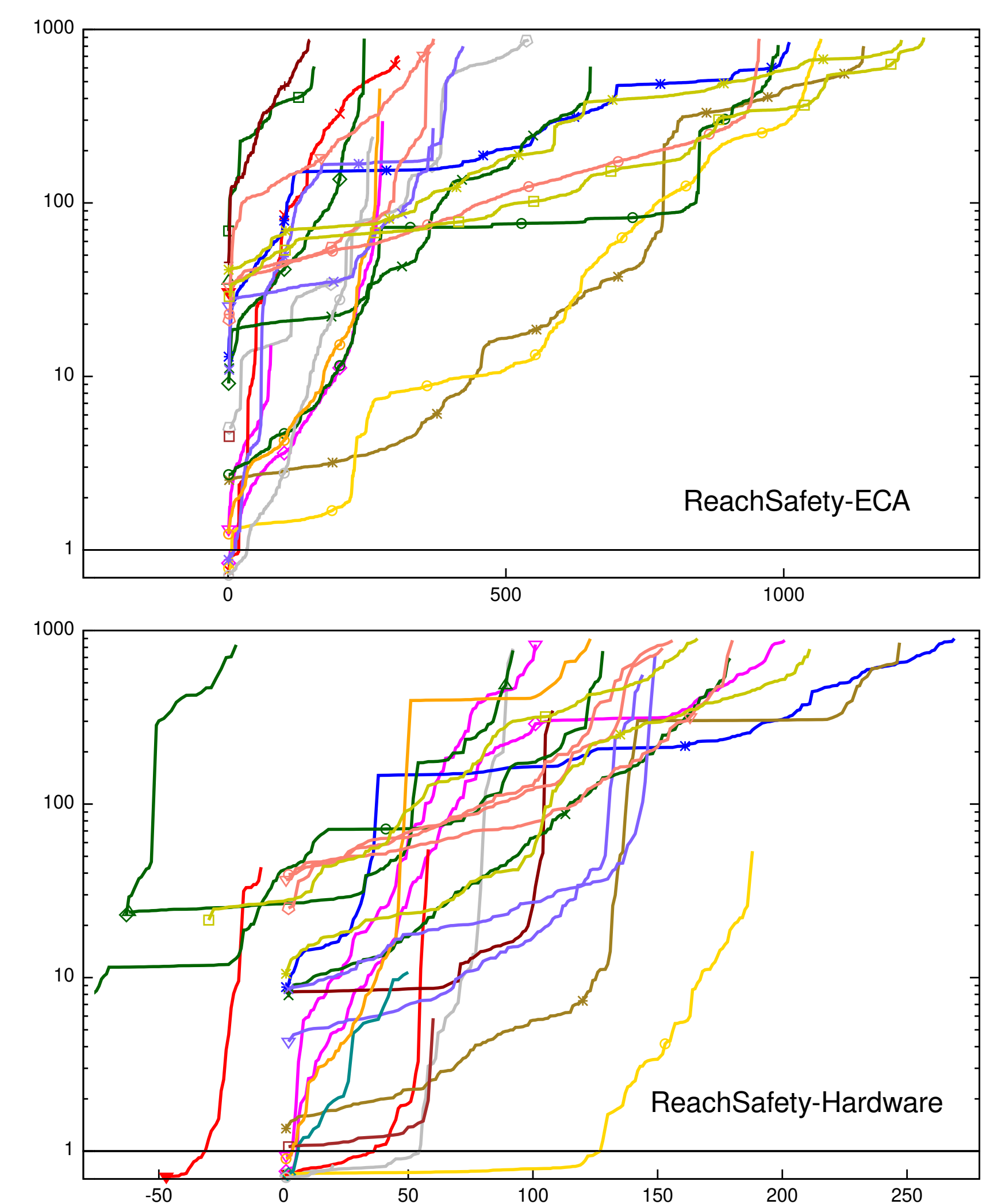
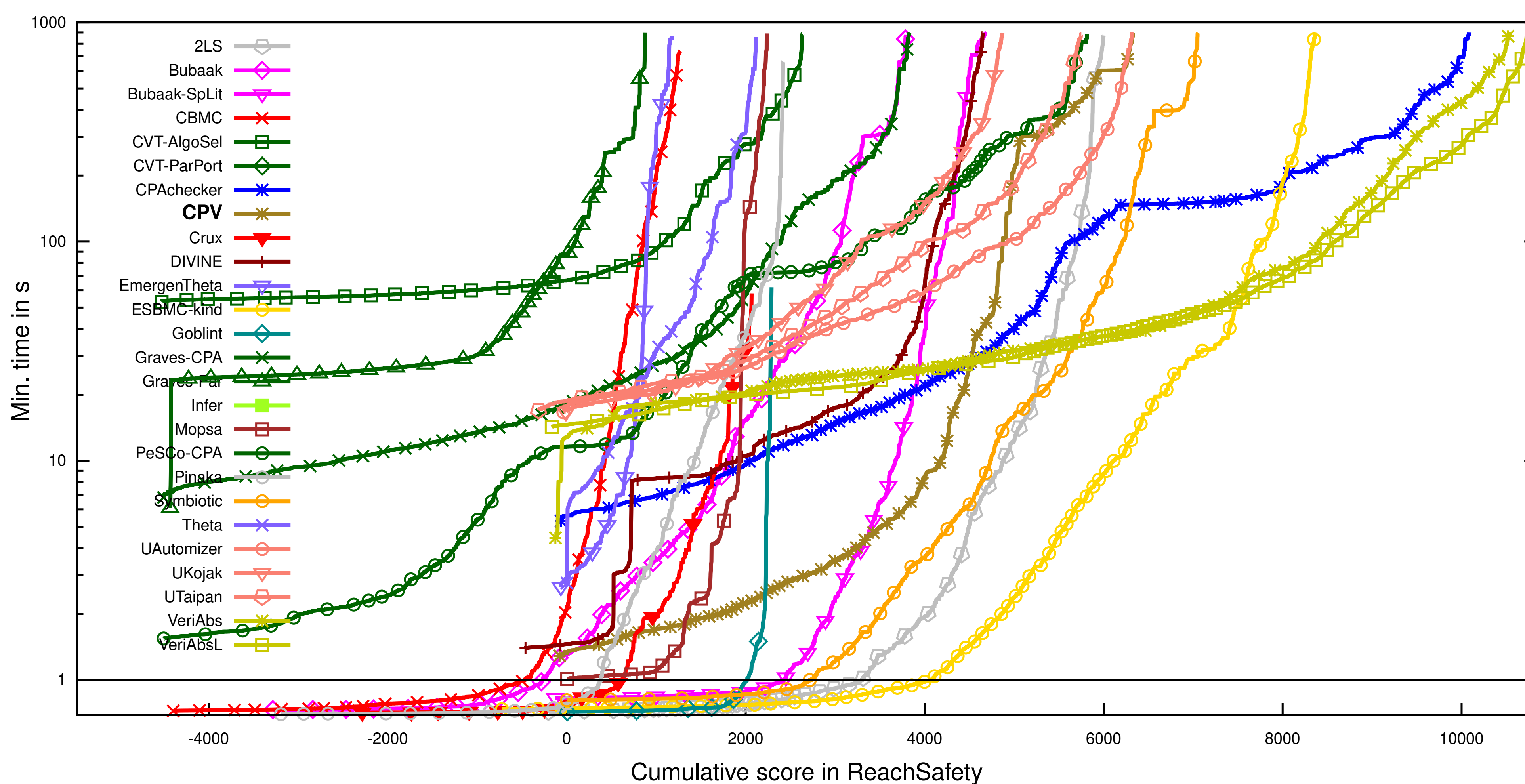
STRATEGY FOR SV-COMP 2024

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.: COVERTeAM: 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)