Bridging Hardware and Software Formal Verification

Po-Chun Chien

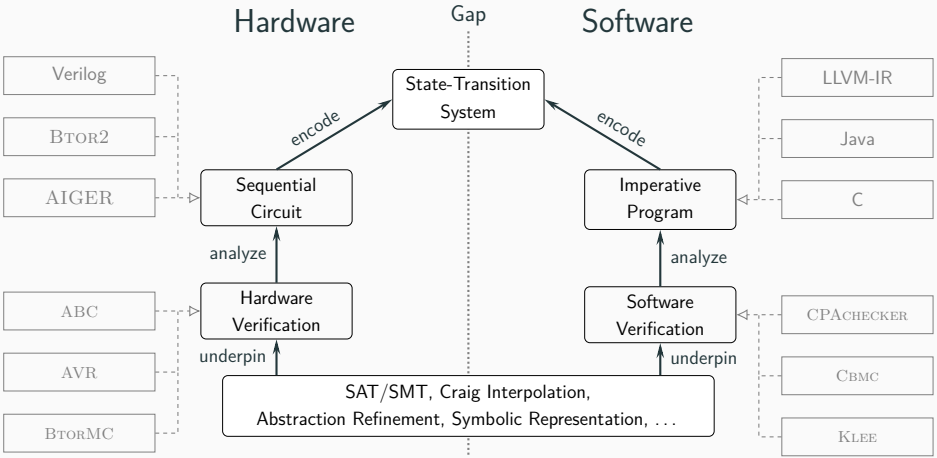
SoSy-Lab, LMU Munich

FM Doctoral Symposium
2024-09-10 @ Milan, Italy

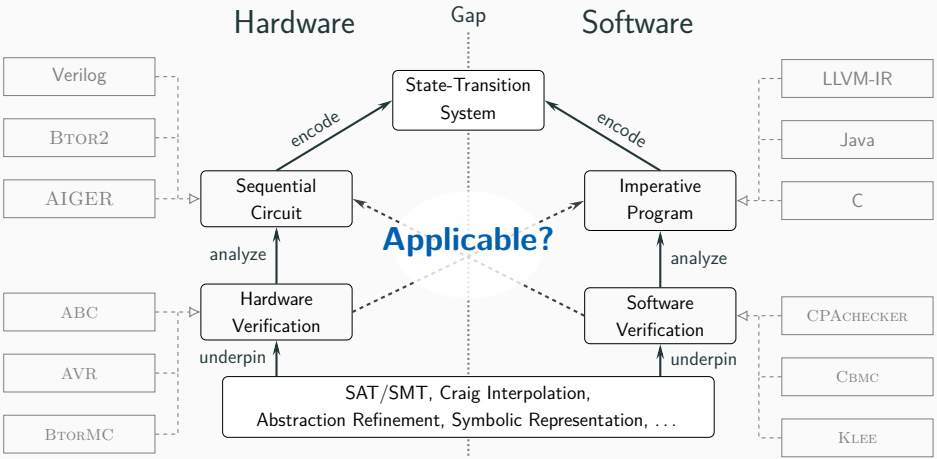
SoSy-Lab
Software Systems

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Motivation


Hardware
Gap
Software
Verilog
Btor2
AIGER
State-Transition System
encode
encode
LLVM-IR
Java
C
Sequential Circuit
Imperative Program
analyze
analyze
ABC
AVR
BtorMC
Hardware Verification
Software Verification
CPAchecker
Cbmc
Klee
underpin
underpin
SAT/SMT, Craig Interpolation,
Abstraction Refinement, Symbolic Representation, ...

Motivation

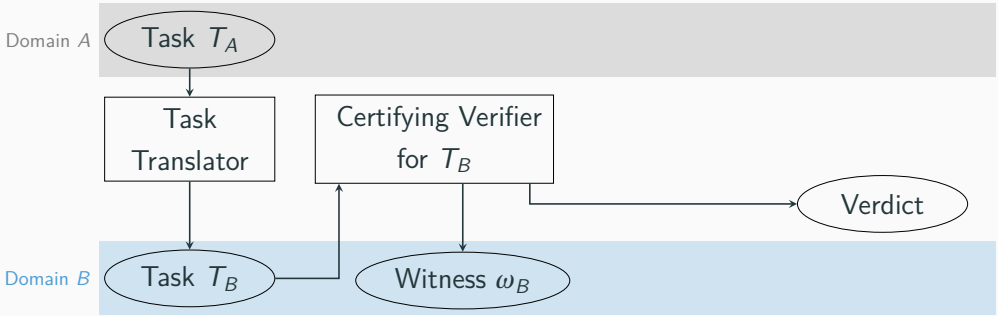

Hardware
Gap
Software
Verilog
Btor2
AIGER
ABC
AVR
BtorMC
State-Transition System
encode
encode
Sequential Circuit
Imperative Program
Applicable?
analyze
analyze
Hardware Verification
Software Verification
underpin
underpin
SAT/SMT, Craig Interpolation,
Abstraction Refinement, Symbolic Representation, ...
LLVM-IR
Java
C
CPAchecker
Cbmc
Klee

Translating Verification Tasks

- BTOR2-CERT [1]
  BTOR2-to-C translation by BTOR2C [3]
- CPV [11]
  C-to-BTOR2 translation by KRATOS2 [16]

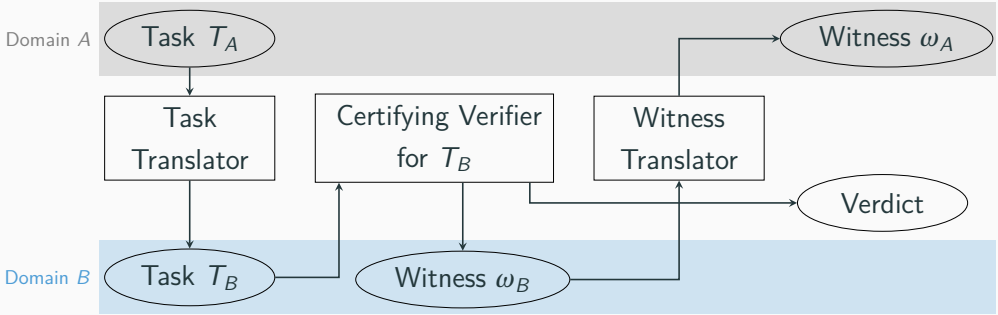
Translating Verification Tasks

- BTOR2-CERT [1]
  BTOR2-to-C translation by BTOR2C [3]
- CPV [11]
  C-to-BTOR2 translation by KRATOS2 [16]

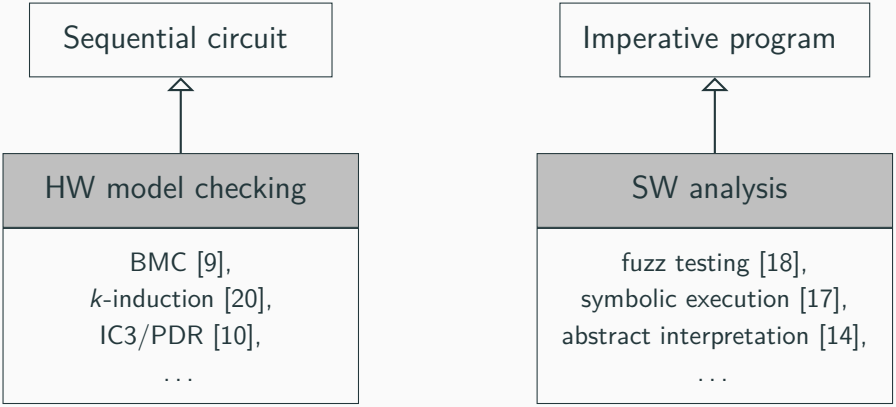


Translating Verification Tasks

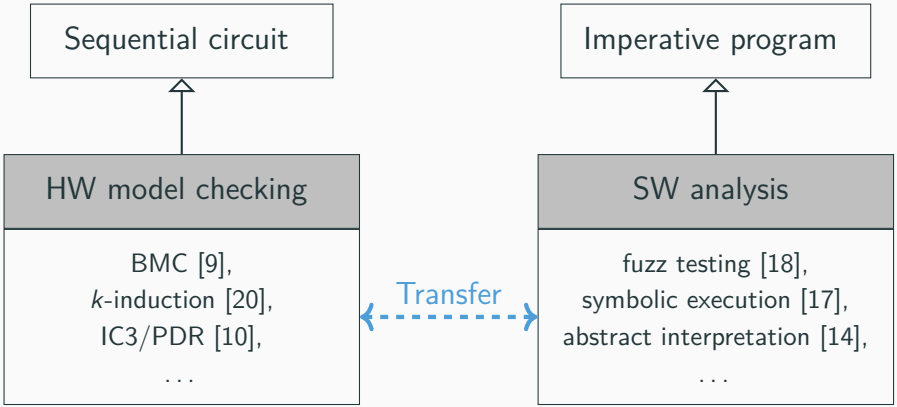
- BTOR2-CERT [1]
  BTOR2-to-C translation by BTOR2C [3]
- CPV [11]
  C-to-BTOR2 translation by KRATOS2 [16]



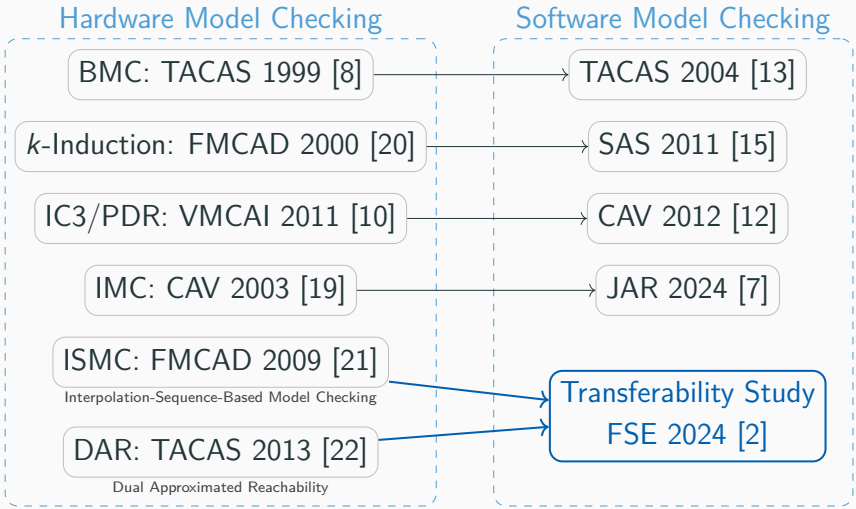
Transferring Verification Techniques Across Domains



Transferring Verification Techniques Across Domains



Verification Algorithm Adoption



Joing Forces of Hardware and Software Verification

- Strengthen Craig interpolants with auxiliary invariants [5]

Joing Forces of Hardware and Software Verification

- Strengthen Craig interpolants with auxiliary invariants [5]
- Augmented vs. plain IMC
  - Improve effectiveness
  - Reduce elapsed wall-time

Data Availability and Reproducibility

- Tools available on `gitlab.com/sosy-lab/software/`



BTOR2-CERT [1] CPV [11] CPACHECKER [6]

- Research artifacts available on Zenodo

Conclusion

- Transformation between different representations to leverage their unique strengths
- Cooperative and cross-disciplinary approaches are beneficial
- Ultimate goal:
  - HW/SW co-verification
  - Tackle more complex heterogeneous systems

References i

[1] Ádám, Z., Beyer, D., Chien, P.C., Lee, N.Z., Sirrenberg, N.: Btor2-Cert: A certifying hardware-verification framework using software analyzers. In: Proc. TACAS. pp. 129–149. LNCS 14572, Springer (2024). https://doi.org/10.1007/978-3-031-57256-2_7

[2] Beyer, D., Chien, P.C., Jankola, M., Lee, N.Z.: A transferability study of interpolation-based hardware model checking to software verification. Proc. ACM Softw. Eng. (2024). https://doi.org/Unpublished:Lastchecked:2024-03-15, conditionally accepted (major revision) at the ACM International Conference on the Foundations of Software Engineering

[3] 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. 152–172. LNCS 13994, Springer (2023). https://doi.org/10.1007/978-3-031-30820-8_12

[4] Beyer, D., Chien, P.C., Lee, N.Z.: CPA-DF: A tool for configurable interval analysis to boost program verification. In: Proc. ASE. pp. 2050–2053. IEEE (2023). https://doi.org/10.1109/ASE56229.2023.00213

References ii

[5] Beyer, D., Chien, P.C., Lee, N.Z.: Augmenting interpolation-based model checking with auxiliary invariants. In: Proc. SPIN. Springer (2024)

[6] 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

[7] Beyer, D., Lee, N.Z., Wendler, P.: Interpolation and SAT-based model checking revisited: Adoption to software verification. J. Autom. Reasoning (2024), accepted, preprint available via https://doi.org/10.48550/arXiv.2208.05046

[8] Biere, A., Cimatti, A., Clarke, E.M., Zhu, Y.: Symbolic model checking without BDDs. In: Proc. TACAS. pp. 193–207. LNCS 1579, Springer (1999). https://doi.org/10.1007/3-540-49059-0_14

[9] 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

[10] Bradley, A.R.: SAT-based model checking without unrolling. In: Proc. VMCAI. pp. 70–87. LNCS 6538, Springer (2011). https://doi.org/10.1007/978-3-642-18275-4_7

References iii

[11] Chien, P.C., Lee, N.Z.: CPV: A circuit-based program verifier (competition contribution). In: Proc. TACAS. pp. 365–370. LNCS 14572, Springer (2024). https://doi.org/10.1007/978-3-031-57256-2_22

[12] Cimatti, A., Griggio, A.: Software model checking via IC3. In: Proc. CAV. pp. 277–293. LNCS 7358, Springer (2012). https://doi.org/10.1007/978-3-642-31424-7_23

[13] Clarke, E.M., Kröning, D., Lerda, F.: A tool for checking ANSI-C programs. In: Proc. TACAS. pp. 168–176. LNCS 2988, Springer (2004). https://doi.org/10.1007/978-3-540-24730-2_15

[14] Cousot, P., Cousot, R.: Abstract interpretation: A unified lattice model for the static analysis of programs by construction or approximation of fixpoints. In: Proc. POPL. pp. 238–252. ACM (1977). https://doi.org/10.1145/512950.512973

[15] Donaldson, A.F., Haller, L., Kröning, D., Rümmer, P.: Software verification using k-induction. In: Proc. SAS. pp. 351–368. LNCS 6887, Springer (2011). https://doi.org/10.1007/978-3-642-23702-7_26

References iv

[16] 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

[17] King, J.C.: Symbolic execution and program testing. Commun. ACM **19**(7), 385–394 (1976). https://doi.org/10.1145/360248.360252

[18] Manès, V.J.M., Han, H., Han, C., Cha, S.K., Egele, M., Schwartz, E.J., Woo, M.: The art, science, and engineering of fuzzing: A survey. IEEE Trans. Software Eng. **47**(11), 2312–2331 (2021). https://doi.org/10.1109/TSE.2019.2946563

[19] 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

[20] 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

[21] Vizel, Y., Grumberg, O.: Interpolation-sequence based model checking. In: Proc. FMCAD. pp. 1–8. IEEE (2009). https://doi.org/10.1109/FMCAD.2009.5351148

References v

[22] Vizel, Y., Grumberg, O., Shoham, S.: Intertwined forward-backward reachability analysis using interpolants. In: Proc. TACAS. pp. 308–323. LNCS 7795, Springer (2013). https://doi.org/10.1007/978-3-642-36742-7_22