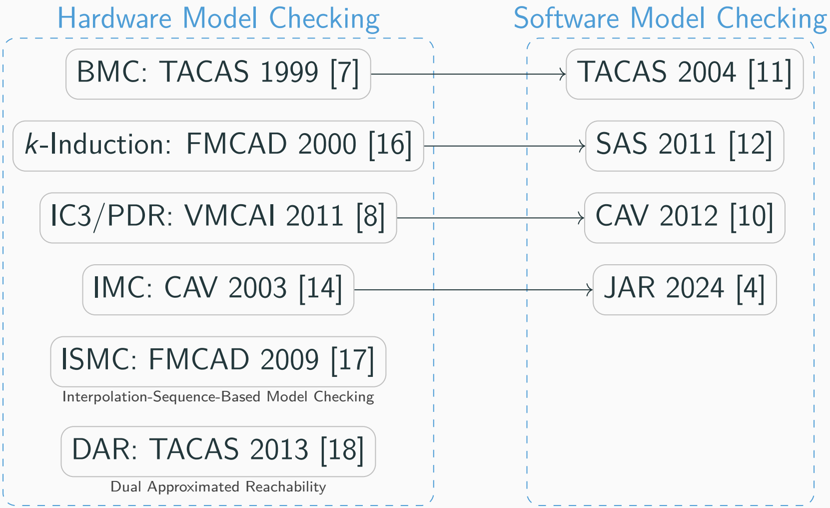
A Transferability Study of Interpolation-Based Hardware Model Checking for Software Verification

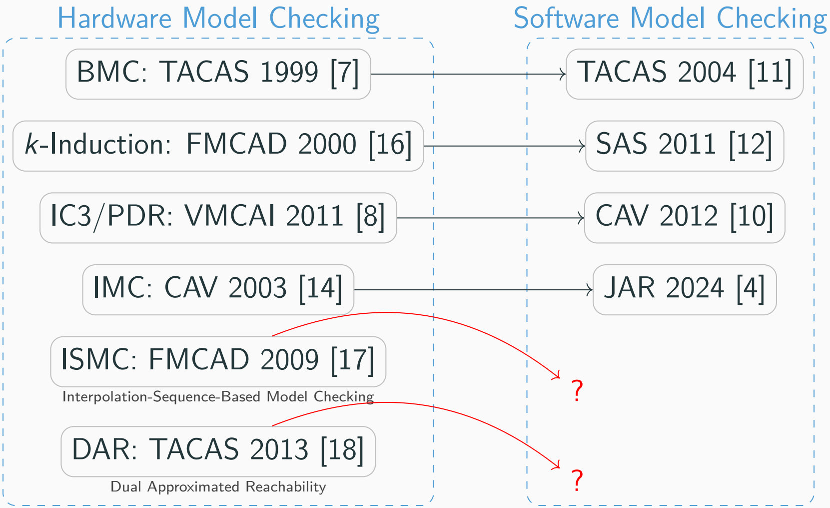
Dirk Beyer, **Po-Chun Chien**, Marek Jankola, and Nian-Ze Lee

CPA 2024 @ Milan, Italy
LMU Munich, Germany

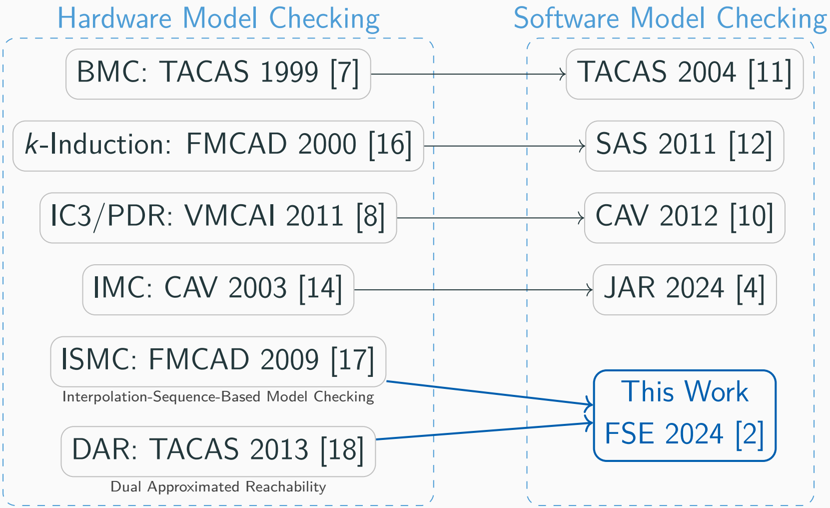
Motivation


Hardware Model Checking
Software Model Checking
BMC: TACAS 1999 [7]
TACAS 2004 [11]
k-Induction: FMCAD 2000 [16]
SAS 2011 [12]
IC3/PDR: VMCAI 2011 [8]
CAV 2012 [10]
IMC: CAV 2003 [14]
JAR 2024 [4]
ISMC: FMCAD 2009 [17]
Interpolation-Sequence-Based Model Checking
DAR: TACAS 2013 [18]
Dual Approximated Reachability

Motivation


Hardware Model Checking
Software Model Checking
BMC: TACAS 1999 [7]
TACAS 2004 [11]
k-Induction: FMCAD 2000 [16]
SAS 2011 [12]
IC3/PDR: VMCAI 2011 [8]
CAV 2012 [10]
IMC: CAV 2003 [14]
JAR 2024 [4]
ISMC: FMCAD 2009 [17]
Interpolation-Sequence-Based Model Checking
?
DAR: TACAS 2013 [18]
Dual Approximated Reachability
?

Motivation


Hardware Model Checking
Software Model Checking
BMC: TACAS 1999 [7]
TACAS 2004 [11]
k-Induction: FMCAD 2000 [16]
SAS 2011 [12]
IC3/PDR: VMCAI 2011 [8]
CAV 2012 [10]
IMC: CAV 2003 [14]
JAR 2024 [4]
ISMC: FMCAD 2009 [17]
Interpolation-Sequence-Based Model Checking
This Work
FSE 2024 [2]
DAR: TACAS 2013 [18]
Dual Approximated Reachability

Highlights

- First systematic study on the transferability of hardware model checking
- For software verification
  - ISMC and DAR are useful
  - Evaluation confirms that algorithmic characteristics remain

Highlights

- First systematic study on the transferability of hardware model checking
- For software verification
  - ISMC and DAR are useful
  - Evaluation confirms that algorithmic characteristics remain
- Open-source implementation of ISMC and DAR in CPAchecker
- Reproduction package available on Zenodo [1]

Agenda

1. Background
2. Interpolation-Based Model Checking
3. Study Design
4. Experimental Evaluation
5. Conclusion

Agenda

1. Background
2. Interpolation-Based Model Checking
3. Study Design
4. Experimental Evaluation
5. Conclusion

Bounded Model Checking

- State-transition system: $Init(s), T(s, s')$
- Safety property: $P(s)$

Bounded Model Checking

- State-transition system: $Init(s), T(s, s')$
- Safety property: $P(s)$
- BMC query with unrolling bound k:
  $Init(s_0) \land T(s_0, s_1) \land T(s_1, s_2) \land \ldots \land T(s_{k-1}, s_k) \land \neg P(s_k)$

Bounded Model Checking

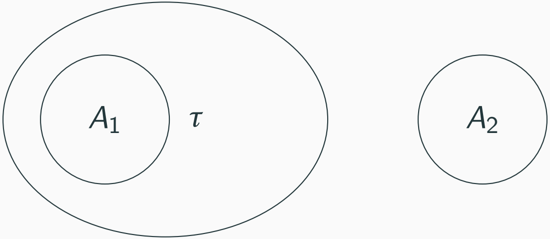
- State-transition system: $Init(s), T(s, s')$
- Safety property: $P(s)$
- BMC query with unrolling bound k:
  $Init(s_0) \land T(s_0, s_1) \land T(s_1, s_2) \land \ldots \land T(s_{k-1}, s_k) \land \neg P(s_k)$
  - if SAT, counterexample found
  - if UNSAT, no counterexample at bound k

Bounded Model Checking

- State-transition system: $Init(s), T(s, s')$
- Safety property: $P(s)$
- BMC query with unrolling bound k:
  $Init(s_0) \wedge T(s_0, s_1) \wedge T(s_1, s_2) \wedge \ldots \wedge T(s_{k-1}, s_k) \wedge \neg P(s_k)$
  - if SAT, counterexample found
  - if UNSAT, no counterexample at bound k
    → compute abstraction using *interpolation*

Craig Interpolation

- If $A_1(X,Y) \wedge A_2(Y,Z)$ is UNSAT: interpolant $\tau(Y)$
  - $A_1(X,Y) \Rightarrow \tau(Y)$ is valid
  - $\tau(Y) \wedge A_2(Y,Z)$ is UNSAT



Agenda

1. Background
2. Interpolation-Based Model Checking
3. Study Design
4. Experimental Evaluation
5. Conclusion

IMC (McMillan 2003 [14])

- At unrolling bound k:
  $Init(s_0)\,T(s_0,s_1)\;T(s_1,s_2)\ldots T(s_{k-1},s_k)(\neg P(s_1)\vee\ldots\vee\neg P(s_k))$

IMC (McMillan 2003 [14])

- At unrolling bound k:
  $$\underbrace{Init(s_0)\,T(s_0,s_1)}_{A_1(s_0,s_1)}\;\underbrace{T(s_1,s_2)\ldots T(s_{k-1},s_k)(\neg P(s_1)\vee\ldots\vee\neg P(s_k))}_{A_2(s_1,s_2,\ldots,s_k)}$$
  - Interpolant $\tau_1(s_1)$: 1-step safe overapproximation

IMC (McMillan 2003 [14])

- At unrolling bound k:
  $$\underbrace{Init(s_0)\,T(s_0,s_1)}_{A_1(s_0,s_1)}\,\underbrace{T(s_1,s_2)\ldots T(s_{k-1},s_k)(\neg P(s_1)\vee\ldots\vee\neg P(s_k))}_{A_2(s_1,s_2,\ldots,s_k)}$$
  - Interpolant $\tau_1(s_1)$: 1-step safe overapproximation
- $$\underbrace{\tau_1(s_0)\,T(s_0,s_1)}_{A'_1(s_0,s_1)}\,\underbrace{T(s_1,s_2)\ldots T(s_{k-1},s_k)(\neg P(s_1)\vee\ldots\vee\neg P(s_k))}_{A_2(s_1,s_2,\ldots,s_k)}$$
  - Interpolant $\tau_2(s_1)$: 2-step safe overapproximation

IMC (McMillan 2003 [14])

- At unrolling bound k:
  $$\underbrace{Init(s_0)\,T(s_0,s_1)}_{A_1(s_0,s_1)}\ \underbrace{T(s_1,s_2)\ldots T(s_{k-1},s_k)(\neg P(s_1)\vee\ldots\vee\neg P(s_k))}_{A_2(s_1,s_2,\ldots,s_k)}$$
  - Interpolant $\tau_1(s_1)$: 1-step safe overapproximation
- $$\underbrace{\tau_1(s_0)\,T(s_0,s_1)}_{A_1'(s_0,s_1)}\ \underbrace{T(s_1,s_2)\ldots T(s_{k-1},s_k)(\neg P(s_1)\vee\ldots\vee\neg P(s_k))}_{A_2(s_1,s_2,\ldots,s_k)}$$
  - Interpolant $\tau_2(s_1)$: 2-step safe overapproximation
  - Derive τ_n iteratively until $\bigvee_{i=1}^{n}\tau_i$ reaches a fixed point

ISMC (Vizel and Grumberg, 2009 [17])

- Forward reachability sequence: $\langle R_1, R_2, \ldots, R_k = \top \rangle$
  - R_i: i-step overapproximation

ISMC (Vizel and Grumberg, 2009 [17])

- Forward reachability sequence: $\langle R_1, R_2, \ldots, R_k = \top \rangle$
  - R_i: i-step overapproximation
- Increment unrolling bound from 1 to k:
  $Init(s_0)\, T(s_0, s_1) \wedge T(s_1, s_2) \wedge \ldots \wedge T(s_{k-1}, s_k) \wedge \neg P(s_k)$

ISMC (Vizel and Grumberg, 2009 [17])

- Forward reachability sequence: $\langle R_1, R_2, \ldots, R_k = \top \rangle$
  - R_i: i-step overapproximation
- Increment unrolling bound from 1 to k:

$$\underbrace{Init(s_0)\,T(s_0, s_1)}_{A_1} \land \underbrace{T(s_1, s_2)}_{A_2} \land \ldots \land \underbrace{T(s_{k-1}, s_k)}_{A_k} \land \underbrace{\neg P(s_k)}_{A_{k+1}}$$

  - Interpolation sequence $\langle \tau_1^k, \tau_2^k, \ldots, \tau_k^k \rangle$: 1- to k-step overapproximations

ISMC (Vizel and Grumberg, 2009 [17])

- Forward reachability sequence: $\langle R_1, R_2, \ldots, R_k = \top \rangle$
  - R_i: i-step overapproximation
- Increment unrolling bound from 1 to k:
  $$\underbrace{Init(s_0)\,T(s_0,s_1)}_{A_1} \land \underbrace{T(s_1,s_2)}_{A_2} \land \ldots \land \underbrace{T(s_{k-1},s_k)}_{A_k} \land \underbrace{\neg P(s_k)}_{A_{k+1}}$$
  - Interpolation sequence $\langle \tau_1^k, \tau_2^k, \ldots, \tau_k^k \rangle$: 1- to k-step overapproximations
  - Update reachability sequence: $R_i' = R_i \land \tau_i^k$ (reuse previous interpolants)

ISMC (Vizel and Grumberg, 2009 [17])

- Forward reachability sequence: $\langle R_1, R_2, \ldots, R_k = \top \rangle$
  - R_i: i-step overapproximation
- Increment unrolling bound from 1 to k:

$$\underbrace{Init(s_0)\,T(s_0,s_1)}_{A_1} \wedge \underbrace{T(s_1,s_2)}_{A_2} \wedge \ldots \wedge \underbrace{T(s_{k-1},s_k)}_{A_k} \wedge \underbrace{\neg P(s_k)}_{A_{k+1}}$$

  - Interpolation sequence $\langle \tau_1^k, \tau_2^k, \ldots, \tau_k^k \rangle$: 1- to k-step overapproximations
  - Update reachability sequence: $R_i' = R_i \wedge \tau_i^k$ (reuse previous interpolants)
  - Check if $\bigvee_{i=1}^{k} R_i'$ is a fixed point

DAR (Vizel, Grumberg, and Shoham, 2013 [18])

- Forward *and backward* reachability sequences
  - $\overline{F} = \langle F_0 = \mathit{Init}, F_1, F_2, \ldots, F_k \rangle$: forward reachability from *Init*
  - $\overline{B} = \langle B_0 = \neg P, B_1, B_2, \ldots, B_k \rangle$: backward reachability from $\neg P$

DAR (Vizel, Grumberg, and Shoham, 2013 [18])

- Forward *and backward* reachability sequences
  - $\overline{F} = \langle F_0 = Init, F_1, F_2, \ldots, F_k \rangle$: forward reachability from *Init*
  - $\overline{B} = \langle B_0 = \neg P, B_1, B_2, \ldots, B_k \rangle$: backward reachability from $\neg P$
- *Local* strengthening
  - If $F_i \wedge T \wedge B_{k-i}$ is UNSAT: no counterexample within $k+1$ steps

DAR (Vizel, Grumberg, and Shoham, 2013 [18])

- Forward *and backward* reachability sequences
  - $\overline{F} = \langle F_0 = Init, F_1, F_2, \ldots, F_k \rangle$: forward reachability from *Init*
  - $\overline{B} = \langle B_0 = \neg P, B_1, B_2, \ldots, B_k \rangle$: backward reachability from $\neg P$
- *Local* strengthening
  - If $F_i \wedge T \wedge B_{k-i}$ is UNSAT: no counterexample within $k+1$ steps
  - Compute forward and backward interpolants to refine and extend $\overline{F}$ and $\overline{B}$

DAR (Vizel, Grumberg, and Shoham, 2013 [18])

- Forward *and backward* reachability sequences
  - $\overline{F} = \langle F_0 = Init, F_1, F_2, \ldots, F_k \rangle$: forward reachability from *Init*
  - $\overline{B} = \langle B_0 = \neg P, B_1, B_2, \ldots, B_k \rangle$: backward reachability from $\neg P$
- *Local* strengthening
  - If $F_i \wedge T \wedge B_{k-i}$ is UNSAT: no counterexample within $k+1$ steps
  - Compute forward and backward interpolants to refine and extend $\overline{F}$ and $\overline{B}$
- *Global* strengthening
  - $\overline{F}$ and $\overline{B}$ not strong enough to refute counterexamples of length $k+1$

DAR (Vizel, Grumberg, and Shoham, 2013 [18])

- Forward *and backward* reachability sequences
  - $\overline{F} = \langle F_0 = Init, F_1, F_2, \ldots, F_k \rangle$: forward reachability from *Init*
  - $\overline{B} = \langle B_0 = \neg P, B_1, B_2, \ldots, B_k \rangle$: backward reachability from $\neg P$
- *Local* strengthening
  - If $F_i \wedge T \wedge B_{k-i}$ is UNSAT: no counterexample within $k+1$ steps
  - Compute forward and backward interpolants to refine and extend $\overline{F}$ and $\overline{B}$
- *Global* strengthening
  - $\overline{F}$ and $\overline{B}$ not strong enough to refute counterexamples of length $k+1$
  - Compute interpolation sequence to refine and extend $\overline{F}$ (similar to ISMC)

DAR (Vizel, Grumberg, and Shoham, 2013 [18])

- Forward *and backward* reachability sequences
  - $\overline{F} = \langle F_0 = Init, F_1, F_2, \ldots, F_k \rangle$: forward reachability from *Init*
  - $\overline{B} = \langle B_0 = \neg P, B_1, B_2, \ldots, B_k \rangle$: backward reachability from $\neg P$
- *Local* strengthening
  - If $F_i \wedge T \wedge B_{k-i}$ is UNSAT: no counterexample within $k+1$ steps
  - Compute forward and backward interpolants to refine and extend $\overline{F}$ and $\overline{B}$
- *Global* strengthening
  - $\overline{F}$ and $\overline{B}$ not strong enough to refute counterexamples of length $k+1$
  - Compute interpolation sequence to refine and extend $\overline{F}$ (similar to ISMC)
- Until $\overline{F}$ or $\overline{B}$ reaches a fixed point

Conceptual Differences between IMC, ISMC, and DAR

	Fixed Point	Reuse Interpolants	Local/Global Queries
IMC	$\bigvee_i \tau_i$	No	Global
ISMC	$\bigvee_i R_i$	Yes	Global
DAR	$\bigvee_i F_i$ or $\bigvee_i B_i$	Yes	Local + Global

Local/Global: query involves one/multiple transition(s)

Adoption for Software Reachability Checking

- Implement IMC, ISMC, and DAR in CPAchecker
- Use *large-block encoding* [3] to obtain transition relations
- Encode program semantics as SMT formulas (bit-precise theories)
- Employ MathSAT5 [9] for SMT solving and interpolation

Agenda

1. Background
2. Interpolation-Based Model Checking
3. Study Design
4. Experimental Evaluation
5. Conclusion

Steps

- Extract findings/claims from original papers of ISMC and DAR
  - Run-time efficiency and effectiveness
  - Algorithmic characteristics: #unrollings, #interpolants, ...
  - Comparisons with IMC
- Perform experiments to validate these findings

ISMC vs. IMC

Findings in original paper [17]	Our study
H1.A: ISMC is faster at finding bugs	✓
H1.B: ISMC proves property faster at high unrolling bounds	
H1.C: ISMC is overall faster	

DAR vs. IMC

Findings in original paper [18]	Our study
H2.A: DAR performs more local than global strengthenings	✓
H2.B: DAR is faster at proving property	
H2.C: DAR computes more interpolants	✓
H2.D: DAR's run-time is more sensitive to sizes of interpolants	
H2.E: DAR is overall faster	

Differences in Experimental Settings

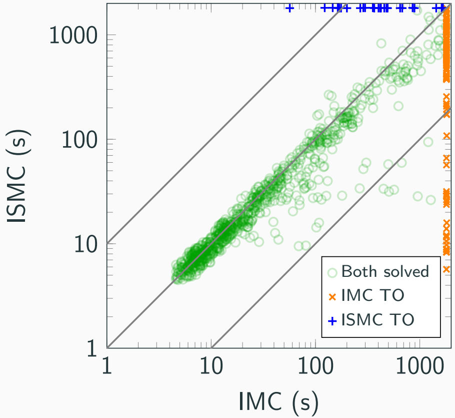
		Ours [2]	ISMC [17]	DAR [18]
	Platform	CPAchecker	Intel's tool	Cadence's Jasper
	Solver	MathSAT5 (SMT)	Eureka (SAT)	SAT solver
Benchmark set	Type	C program	HW circuit	HW circuit
	Source	AWS-C-Comm., Linux, ...	Intel CPUs	Industrial designs
	#safe	6 020	69	37
	#unsafe	2 793	67	≥ 4
Limit	Time	1 800 s	10 000 s	1 800 s
	Memory	15 GB	N/A	N/A

Agenda

1. Background
2. Interpolation-Based Model Checking
3. Study Design
4. Experimental Evaluation
5. Conclusion

Evaluation: ISMC vs. IMC

H1.A: ISMC is faster than IMC at finding property violations ✓

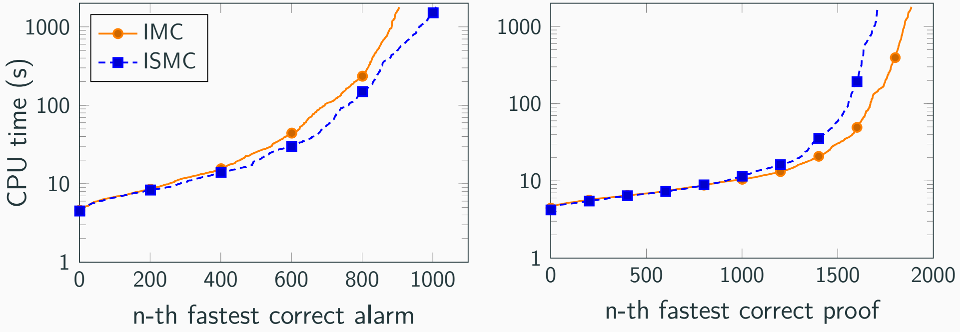


- Both solved: 873
- Accumulated CPU time

ISMC: 67 300 s
IMC: 93 300 s
Ratio: 0.72

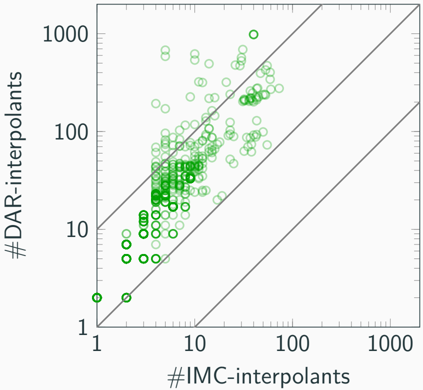
Evaluation: ISMC vs. IMC

H1.C: Overall, ISMC is faster than IMC (by 30 % in the original publication). ?



Evaluation: DAR vs. IMC

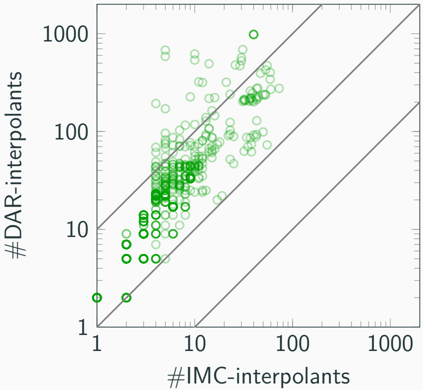
H2.C: DAR computes more interpolants than IMC. ✓

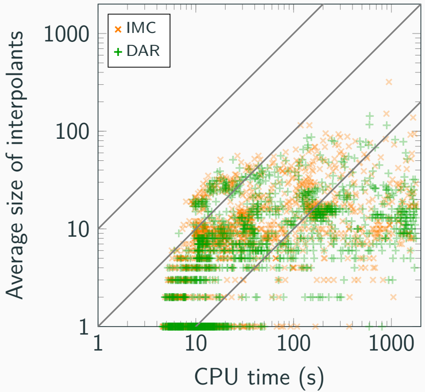


Evaluation: DAR vs. IMC

H2.C: DAR computes more interpolants than IMC. ✓

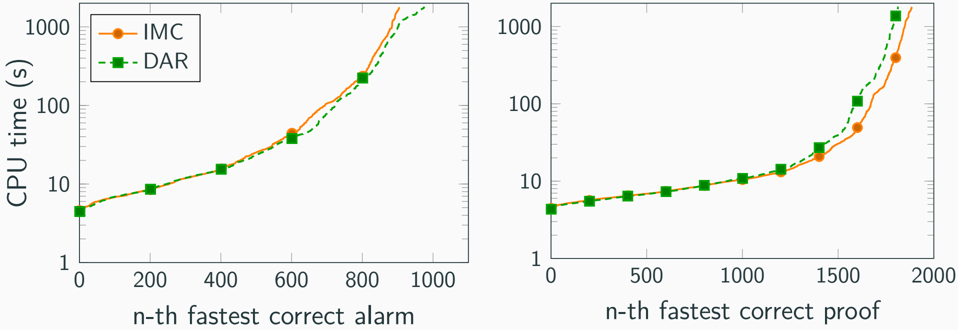
H2.D: DAR's run-time is more sensitive to the sizes of interpolants than IMC. ?





Evaluation: DAR vs. IMC

H2.E: Overall, DAR is faster than IMC (by 36% in the original publication). ?



Agenda

Conclusion

- A systematic study on the transferability of hardware model checking
- Revisiting previous literature is oftentimes fruitful
- ISMC and DAR are useful additions to software-verification methods

doi: 10.1145/3660797

References i

[1] Beyer, D., Chien, P.C., Jankola, M., Lee, N.Z.: Reproduction package for FSE 2024 article 'A transferability study of interpolation-based hardware model checking for software verification'. Zenodo (2024). https://doi.org/10.5281/zenodo.11070973

[2] Beyer, D., Chien, P.C., Jankola, M., Lee, N.Z.: A transferability study of interpolation-based hardware model checking for software verification. Proc. ACM Softw. Eng. **1**(FSE) (2024). https://doi.org/10.1145/3660797

[3] Beyer, D., Cimatti, A., Griggio, A., Keremoglu, M.E., Sebastiani, R.: Software model checking via large-block encoding. In: Proc. FMCAD. pp. 25–32. IEEE (2009). https://doi.org/10.1109/FMCAD.2009.5351147

References ii

[4] Beyer, D., Lee, N.Z., Wendler, P.: Interpolation and SAT-based model checking revisited: Adoption to software verification. J. Autom. Reasoning (2024). https://doi.org/10.1007/s10817-024-09702-9, preprint: https://doi.org/10.48550/arXiv.2208.05046

[5] Beyer, D., Löwe, S., Wendler, P.: Reliable benchmarking: Requirements and solutions. Int. J. Softw. Tools Technol. Transfer **21**(1), 1–29 (2019). https://doi.org/10.1007/s10009-017-0469-y

[6] Beyer, D., Wendler, P.: CPAchecker releases. Zenodo. https://doi.org/10.5281/zenodo.3816620

References iii

[7] 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

[8] 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

[9] Cimatti, A., Griggio, A., Schaafsma, B.J., Sebastiani, R.: The MathSAT5 SMT solver. In: Proc. TACAS. pp. 93–107. LNCS 7795, Springer (2013). https://doi.org/10.1007/978-3-642-36742-7_7

References iv

[10] 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

[11] 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

[12] 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 v

[13] Henzinger, T.A., Jhala, R., Majumdar, R., McMillan, K.L.: Abstractions from proofs. In: Proc. POPL. pp. 232–244. ACM (2004). https://doi.org/10.1145/964001.964021

[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

[15] McMillan, K.L.: Lazy abstraction with interpolants. In: Proc. CAV. pp. 123–136. LNCS 4144, Springer (2006). https://doi.org/10.1007/11817963_14

References vi

[16] 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

[17] 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

[18] 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