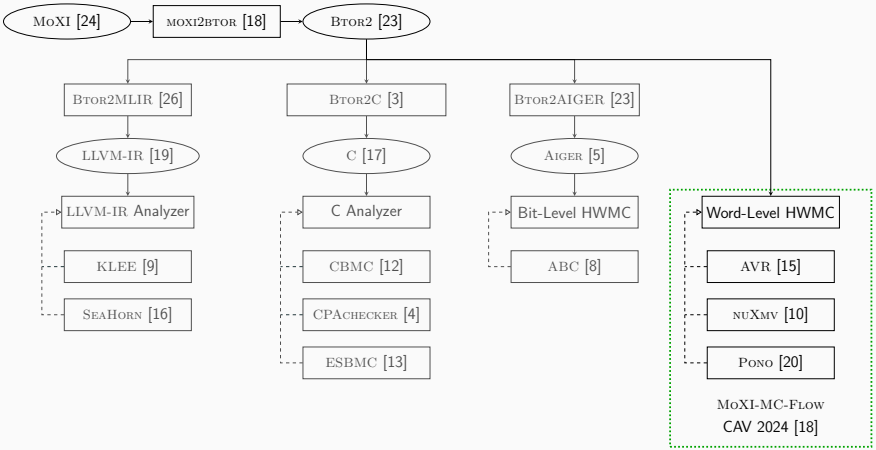
MoXIchecker: An Extensible Model Checker for MoXI

Salih Ates[1], Dirk Beyer[1], **Po-Chun Chien**[1], and Nian-Ze Lee[2,1]

[1]LMU Munich · [2]National Taiwan University

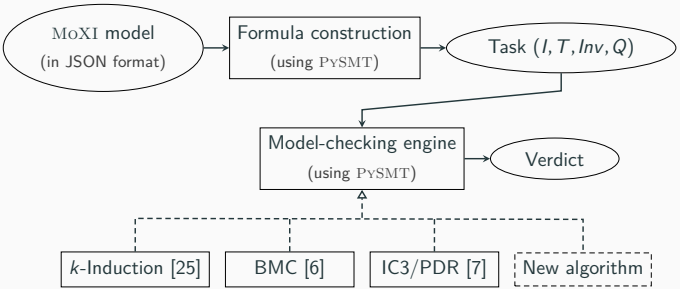
SPIN 2025 @ Hamilton, Canada

Previous Work: Transformation-Based Verification for MoXI



- Btor2 [23] only supports `QF_ABV` → restricted theories

Our Work: MoXIchecker

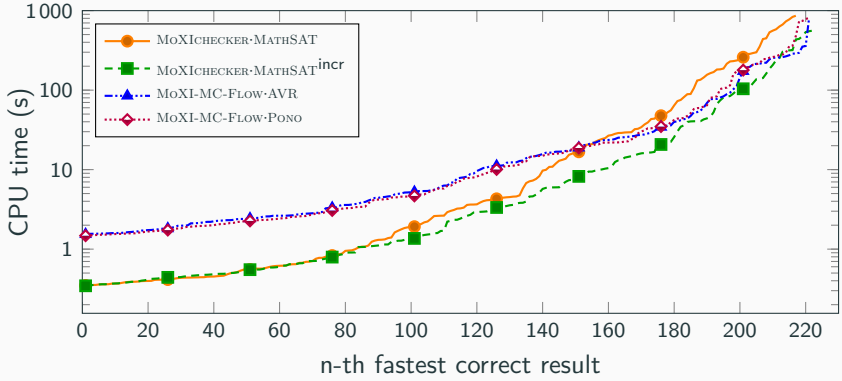


VSTTE 2024 [1]

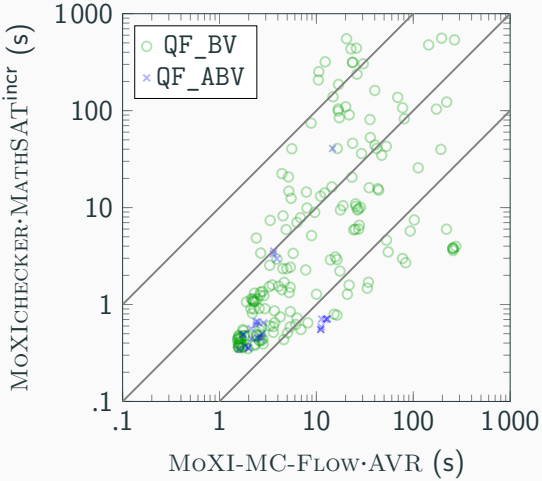
- Verify MoXI models directly (no translation step involved)
- Implemented in Python, based on PySMT [14]
- Supported theories: `QF_{BV,ABV,LIA,NIA,LRA,NRA}`
- Easily extensible: new theories and algorithms

MoXIchecker's Performance

- Compare to MoXI-MC-Flow on 453 QF_BV and QF_ABV tasks
- All configurations run k-induction



MoXIchecker's Run-Time Efficiency



- Avoids translation overhead
- Competitive against SOTA HWMC backends

MoXIchecker's Extensibility

Theories of Integers and Reals

Task	Theory	Verdict	MoXIchecker	MoXI-MC-Flow*
FibonacciSequence	QF_LIA	safe	safe	unsafe
IntIncrement	QF_LIA	unsafe	unsafe	safe
IntCounter	QF_LIA	safe	safe	timeout
IntMultiply	QF_NIA	safe	safe	unsafe
BoundedLinearGrowth	QF_LRA	safe	safe	unsupported
DoubleDelay2	QF_LRA	unsafe	unsafe	unsupported
OscillatingRatio	QF_NRA	safe	safe	unsupported
SafeNonlinearGrowth	QF_NRA	safe	safe	unsupported
NonlinearGrowth	QF_NRA	unsafe	unsafe	unsupported

*MoXI-MC-Flow approximates integers with fixed-size bit-vectors

Remark: Using Python and PySMT

Pros

- Slim code base (~700 LoC)
- East development
- Good for education

Cons

- Performance / efficiency
- Often exceed recursion limit when loading MoXI-JSON
- SMT solver support

Ongoing Development in MoXIchecker

- Preliminary support for `subsys`

gitlab.com/sosy-lab/software/moxichecker

Ongoing Development in MoXIchecker

- Preliminary support for `subsys`
- Enhancing existing engines and adding new ones
  - Implementing interpolation-based algorithms
  - Improving generalization for IC3/PDR

`gitlab.com/sosy-lab/software/moxichecker`

Ongoing Development in MoXIchecker

- Preliminary support for `subsys`
- Enhancing existing engines and adding new ones
  - Implementing interpolation-based algorithms
  - Improving generalization for IC3/PDR
- Performance optimization using Cython

`gitlab.com/sosy-lab/software/moxichecker`

Ongoing Development in Pono

- Pono: a SMT-based model checker
  - Frontends: BTOR2, SMV, VMT, and MoXI!
  - Backends: IC3/PDR, interpolation-based, BMC, *k*-induction, ...
- Joint work with Clark Barrett, Anna Eaton, Ahmed Irfan, Áron Ricardo Perez-Lopez, and Nestan Tsiskaridze (Stanford University)

Growing the MoXI Ecosystem

- Open-source tool suites, e.g., parser in C/C++ (cf. Btor2Tools and PyVMT)

Growing the MoXI Ecosystem

- Open-source tool suites, e.g., parser in C/C++ (cf. Btor2Tools and PyVMT)
- Reduce nesting depth in MoXI-JSON format
  - `ModelChecker/moxi-mc-flow#29`
  - Currently, many tasks cannot be loaded with Python's builtin JSON parser

Growing the MoXI Ecosystem

- Open-source tool suites, e.g., parser in C/C++ (cf. Btor2Tools and PyVMT)
- Reduce nesting depth in MoXI-JSON format
  - `ModelChecker/moxi-mc-flow#29`
  - Currently, many tasks cannot be loaded with Python's builtin JSON parser
- Expand benchmark set: fairness, more theories, quantifiers, ...
- Tool competition

Conclusion

- MoXIchecker: extensible framework for direct MoXI model checking
  - Supported theories: `QF_{BV,ABV,LIA,NIA,LRA,NRA}`
  - Available algorithms: BMC, *k*-induction, IC3/PDR

@ GitLab

@ VSTTE 2024 [1]

References i

[1] Ates, S., Beyer, D., Chien, P.C., Lee, N.Z.: MoXIchecker: An extensible model checker for MoXI. In: Proc. VSTTE 2024. pp. 1–14. LNCS 15525, Springer (2025). https://doi.org/10.1007/978-3-031-86695-1_1

[2] Barrett, C., Stump, A., Tinelli, C.: The SMT-LIB Standard: Version 2.0. Tech. rep., University of Iowa (2010), https://smtlib.cs.uiowa.edu/papers/smt-lib-reference-v2.0-r10.12.21.pdf

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

[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

References ii

[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

[6] 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

[7] 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

[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

[9] Cadar, C., Dunbar, D., Engler, D.R.: KLEE: Unassisted and automatic generation of high-coverage tests for complex systems programs. In: Proc. OSDI. pp. 209–224. USENIX Association (2008). https://dl.acm.org/doi/10.5555/1855741.1855756

References iii

[10] Cavada, R., Cimatti, A., Dorigatti, M., Griggio, A., Mariotti, A., Micheli, A., Mover, S., Roveri, M., Tonetta, S.: The NUXMV symbolic model checker. In: Proc. CAV. pp. 334–342. LNCS 8559, Springer (2014). https://doi.org/10.1007/978-3-319-08867-9_22

[11] 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

[12] 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

[13] 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

[14] Gario, M., Micheli, A.: PYSMT: A solver-agnostic library for fast prototyping of SMT-based algorithms. In: Proc. SMT (2015)

References iv

[15] 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

[16] Gurfinkel, A., Kahsai, T., Komuravelli, A., Navas, J.A.: The SeaHorn verification framework. In: Proc. CAV. pp. 343–361. LNCS 9206, Springer (2015). https://doi.org/10.1007/978-3-319-21690-4_20

[17] ISO/IEC JTC 1/SC 22: ISO/IEC 9899-2018: Information technology — Programming Languages — C. International Organization for Standardization (2018), https://www.iso.org/standard/74528.html

[18] Johannsen, C., Nukala, K., Dureja, R., Irfan, A., Shankar, N., Tinelli, C., Vardi, M.Y., Rozier, K.Y.: The MoXI model exchange tool suite. In: Proc. CAV. pp. 203–218. LNCS 14681, Springer (2024). https://doi.org/10.1007/978-3-031-65627-9_10

[19] Lattner, C., Adve, V.S.: Llvm: A compilation framework for lifelong program analysis and transformation. In: Proc. CGO. pp. 75–88. IEEE (2004). https://doi.org/10.1109/CGO.2004.1281665

References v

[20] Mann, M., Irfan, A., Lonsing, F., Yang, Y., Zhang, H., Brown, K., Gupta, A., Barrett, C.W.: Pono: A flexible and extensible SMT-based model checker. In: Proc. CAV. pp. 461–474. LNCS 12760, Springer (2021). https://doi.org/10.1007/978-3-030-81688-9_22

[21] McMillan, K.L.: The SMV system. In: Symbolic Model Checking, pp. 61–85 (1993). https://doi.org/10.1007/978-1-4615-3190-6_4

[22] de Moura, L.M., Bjørner, N.: Z3: An efficient SMT solver. In: Proc. TACAS. pp. 337–340. LNCS 4963, Springer (2008). https://doi.org/10.1007/978-3-540-78800-3_24

[23] 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

[24] Rozier, K.Y., Dureja, R., Irfan, A., Johannsen, C., Nukala, K., Shankar, N., Tinelli, C., Vardi, M.Y.: MoXI: An intermediate language for symbolic model checking. In: Proc. SPIN. pp. 26–46. LNCS 14624, Springer (2024). https://doi.org/10.1007/978-3-031-66149-5_2

References vi

[25] 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

[26] Tafese, J., Garcia-Contreras, I., Gurfinkel, A.: BTOR2MLIR: A format and toolchain for hardware verification. In: Proc. FMCAD. pp. 55–63. TU Wien Academic Press (2023). https://doi.org/10.34727/2023/ISBN.978-3-85448-060-0_13