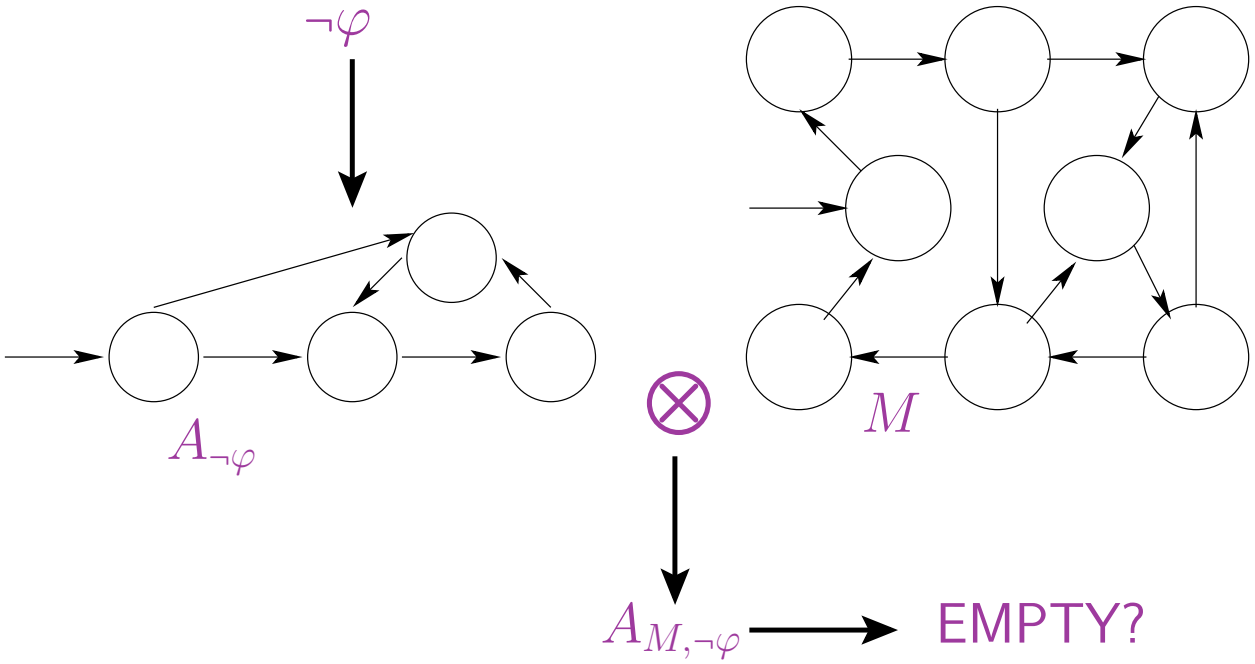
Developing an Open-Source, State-of-the-Art Symbolic Model-Checking Framework for the Model-Checking Research Community

Kristin Yvonne Rozier
Iowa State University

April 23, 2024

Automata-Theoretic Approach to Model Checking[1]



[1] Vardi, Wolper, LICS, 1986

Evolution of Model-Checking Algorithms

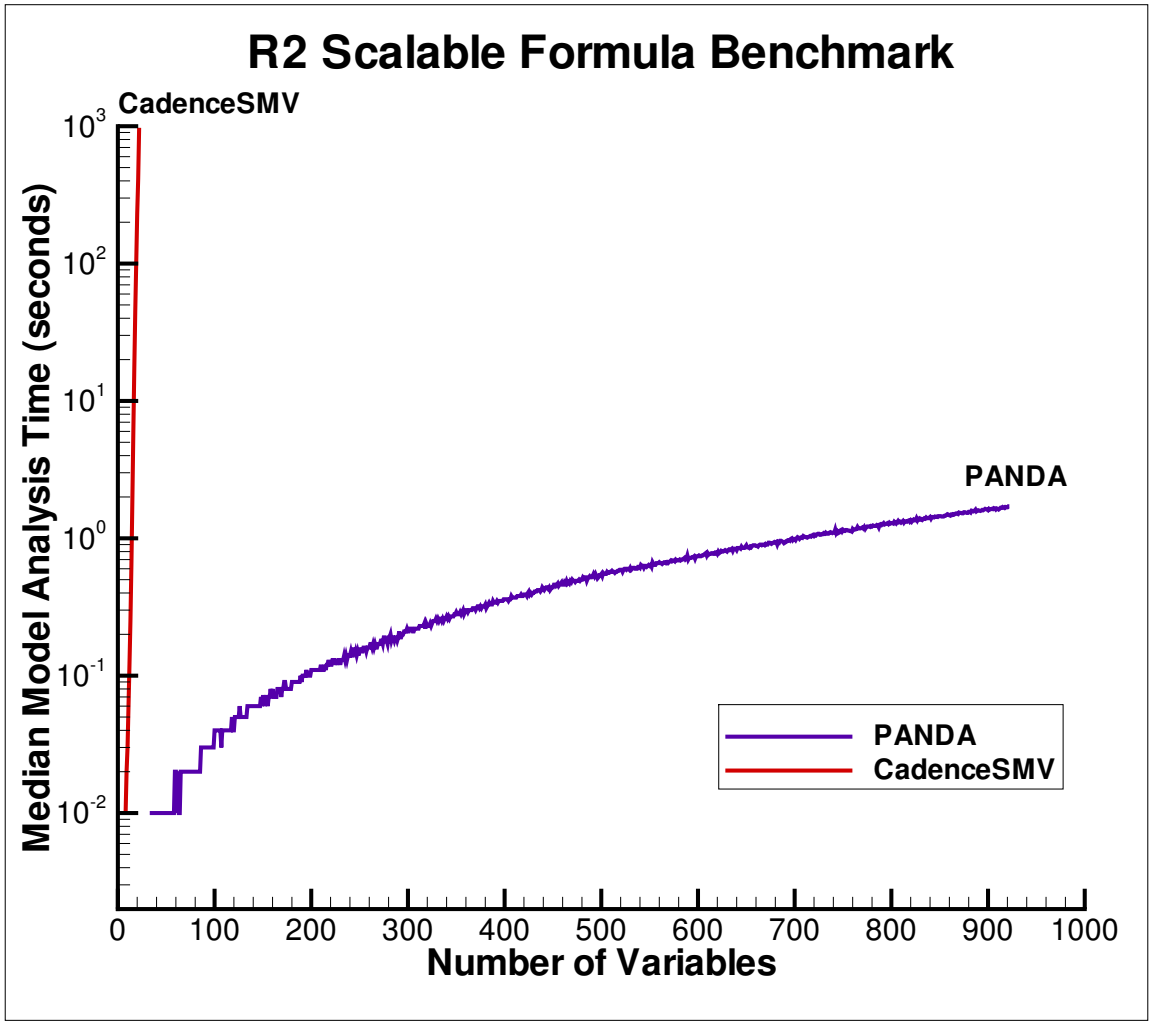
1. BDD-based
2. SAT-based / bounded model checking
3. IC3 / k-liveness

Evolution of Model-Checking Algorithms

1. BDD-based
2. SAT-based / bounded model checking
3. IC3 / k-liveness

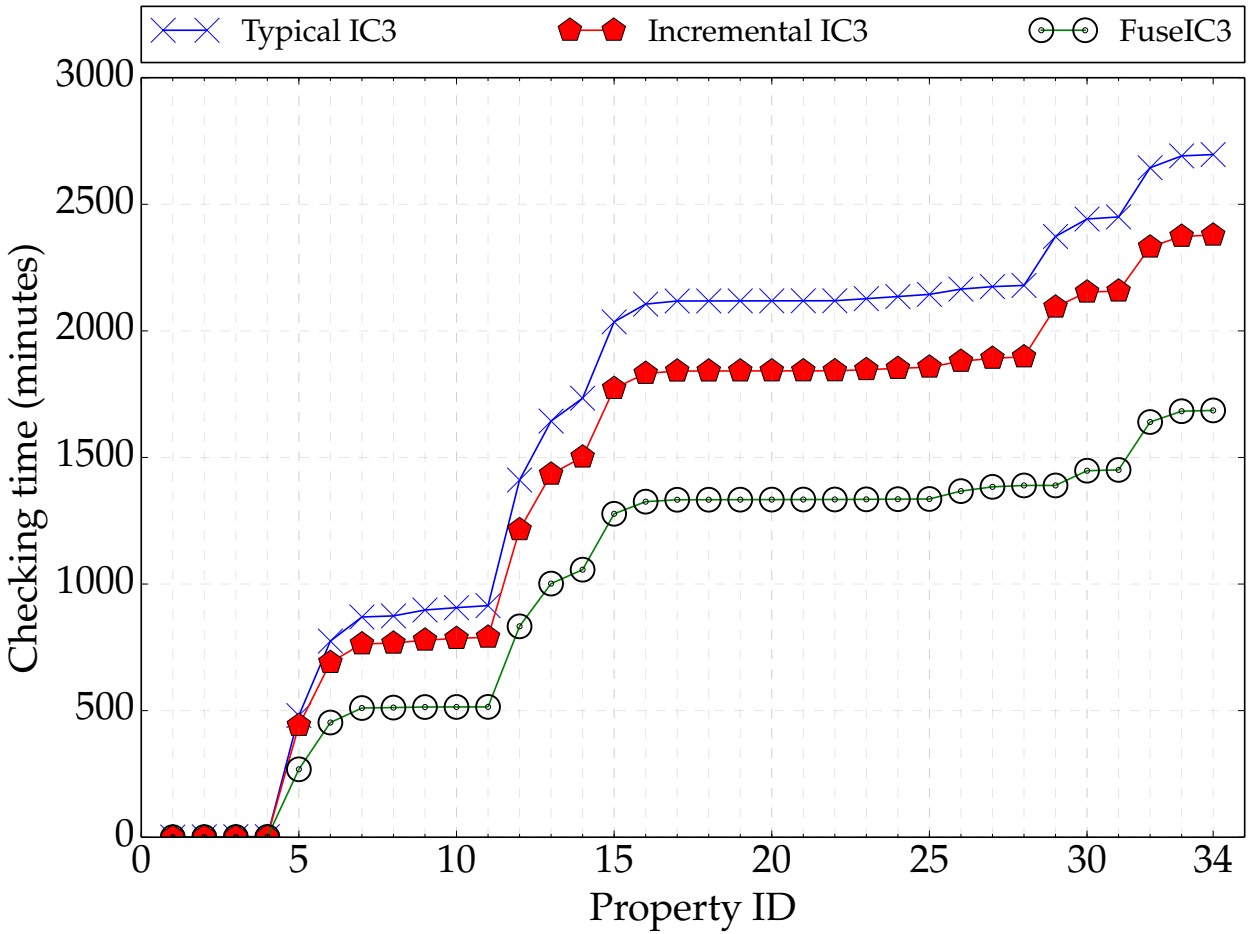
Symbolic model checking progressed from bit-level to word level

The Problem[2]



[2] K.Y.Rozier and M.Y.Vardi, "A Multi-Encoding Approach for LTL Symbolic Satisfiability Checking," FM 2011.

FuseIC3: An Algorithm for Checking Large Design Spaces[3]



Model checking **34 formulas** over **1,620 models** is **5.48x faster**

[3] Rohit Dureja and Kristin Yvonne Rozier. "FuseIC3: An Algorithm for Checking Large Design Spaces." In Formal Methods in Computer-Aided Design (FMCAD), IEEE/ACM, Vienna, Austria, October 2-6, 2017.

The Major Problems

- nuXmv **closed source**: can't check internal algorithm

The Major Problems

- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem

The Major Problems

- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem
- **Can't build** on either tool

The Major Problems

- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem
- **Can't build** on either tool
- Have **models in SMV** language: can't MC with IBM's algorithm

The Major Problems

- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem
- **Can't build** on either tool
- Have **models in SMV** language: can't MC with IBM's algorithm
- Have **new algorithm**; also can't check SMV models with it

The Major Problems

- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem
- **Can't build** on either tool
- Have **models in SMV** language: can't MC with IBM's algorithm
- Have **new algorithm**; also can't check SMV models with it

Solution: program an **entire model checker** taking SMV as input and implementing all three algorithms from scratch to compare them

The Major Problems

- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem
- **Can't build** on either tool
- Have **models in SMV** language: can't MC with IBM's algorithm
- Have **new algorithm**; also can't check SMV models with it

Solution: program an **entire model checker** taking SMV as input and implementing all three algorithms from scratch to compare them

- not publishable, LOTS of time, hard to get right, waste of effort

The Major Problems

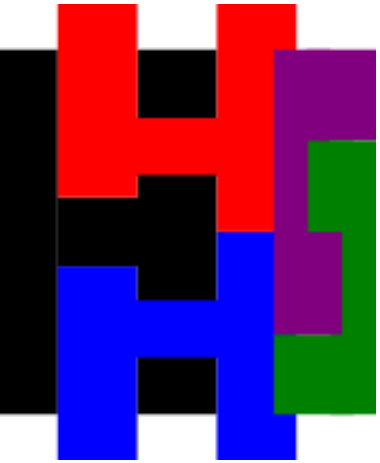
- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem
- **Can't build** on either tool
- Have **models in SMV** language: can't MC with IBM's algorithm
- Have **new algorithm**; also can't check SMV models with it

Solution: program an **entire model checker** taking SMV as input and implementing all three algorithms from scratch to compare them

- not publishable, LOTS of time, hard to get right, waste of effort

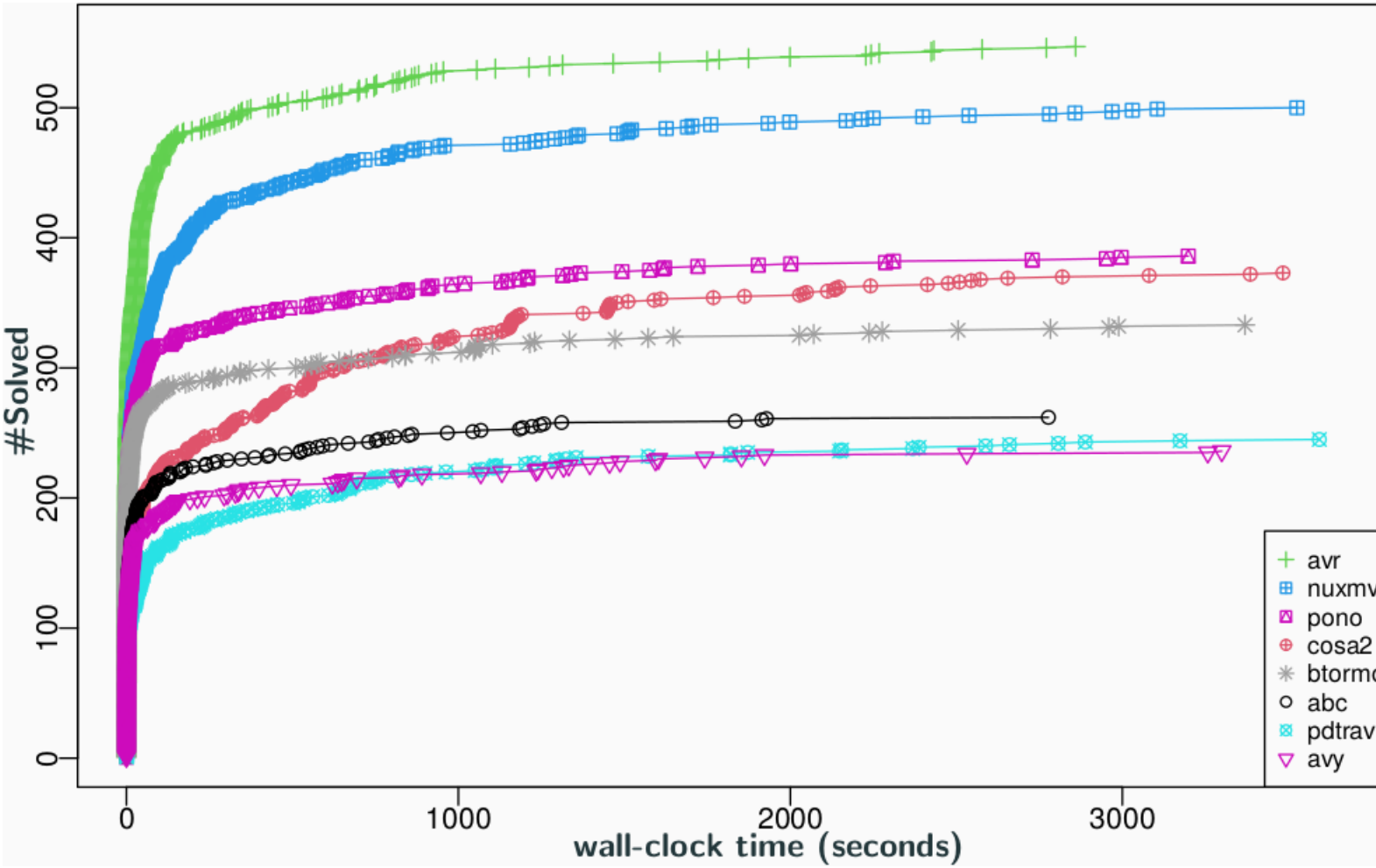
How do we do better?

HWMCC: Hardware Model Checking Competition (2020)



Word-level reasoning

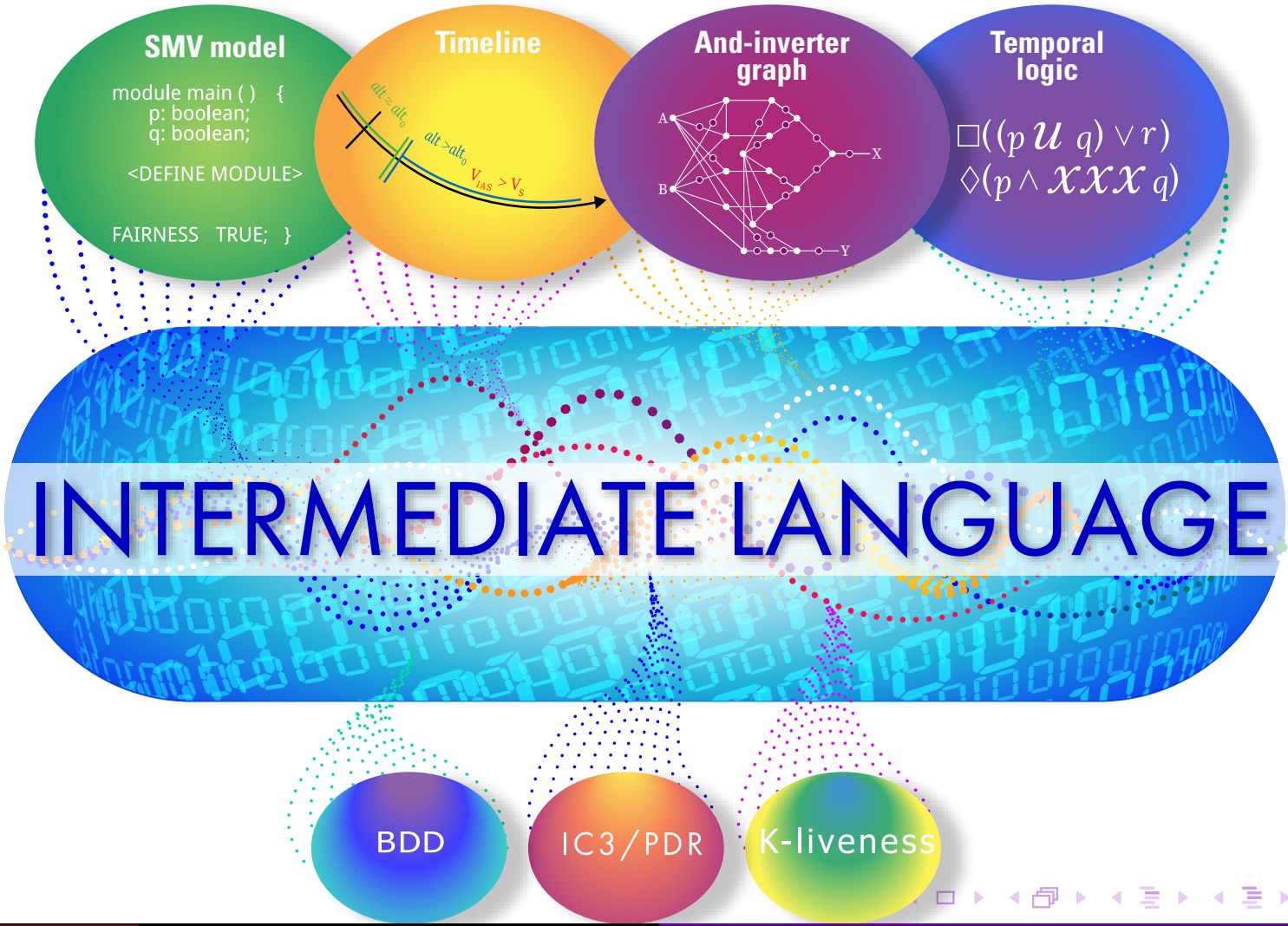
BTOR2



The Problem Continues...

- nuXmv, CadenceSMV, others are **closed source**
- ABC, HWMCC tools are **limited to low-level modeling languages**
- No **open-source, research-enabling connection** between:
  - Rich modeling languages with real-world benchmark models
  - State-of-the-art back-end MC algorithms

MoXI: Model eXchange Interlingua



Goals for Intermediate Language

- Allow adding a **modeling language** via **translation to/from MoXI**
- Allow adding an **MC algorithm** via **translation to/from MoXI**
- MoXI is efficient/accessible so as to **encourage usage in future MCs**
- MoXI: suitable for on-going **community standard**

Basis for MoXI as the Intermediate Language

- **"Best of" previous work**: SMT-LIB, VMT-LIB-Iowa, VMT-LIB-FBK, SAL/Sally, Cryptol, OCRA, synchronous reactive components, ...
- Relatively **simple**, but **complete**
- **Easy to parse**
- Allow **different encodings** (from/to higher/lower levels)

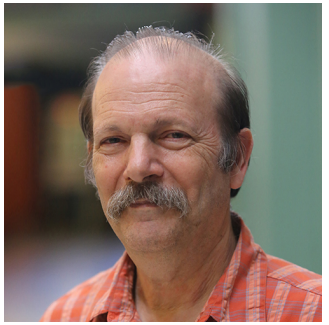
Core Design Team

Investigators:

K.Y. Rozier

Natarajan Shankar

Cesare Tinelli

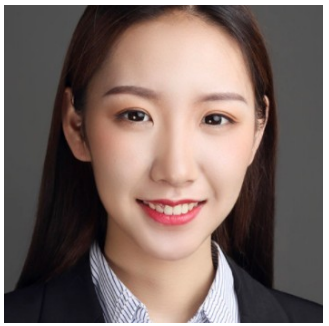


Moshe Vardi

Students:

Laura Gamboa Guzman

Chris Johannsen

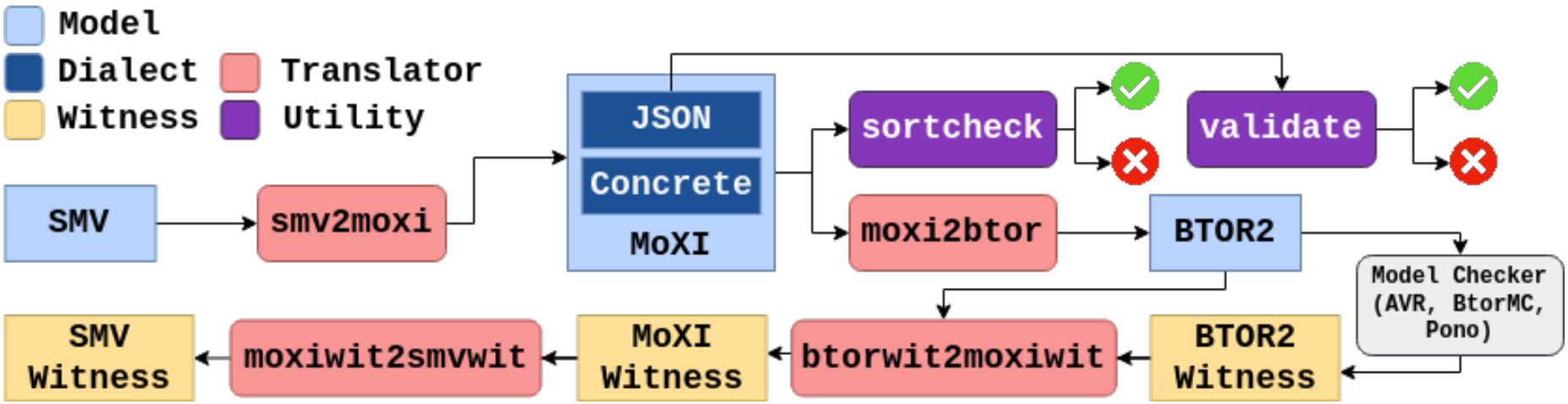


Yi Lin

Technical Advisory Board (TAB)

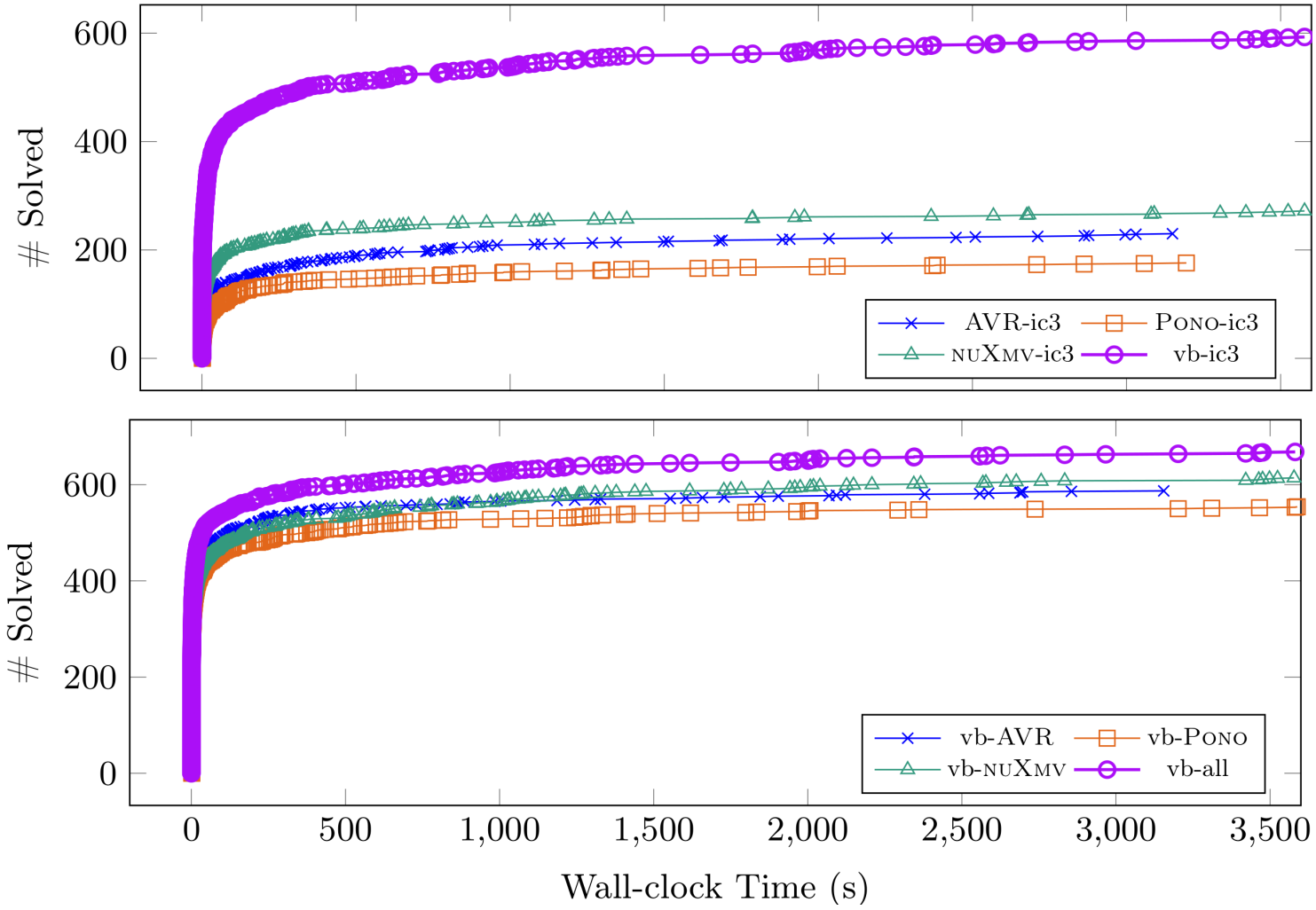
Rajeev Alur (Univ of Pennsylvania)
Clark Barrett (Stanford)
Dirk Beyer (LMU Munich)
Armin Biere (Albert-Ludwigs Univ)
Nikolaj Bjorner (Microsoft Research)
Dimitra Giannakopoulou (Amazon)
Alberto Griggio (FBK)
Orna Grumberg (Technion)
Aarti Gupta (Princeton)
Arie Gurfinkel (Univ of Waterloo)
Ahmed Irfan (SRI)
John Matthews (Intel)
Ken McMillan (UT Austin)
Alan Mishchenko (Berkeley)
Karem Sakallah (Univ of Michigan)
Bernhard Steffen (TU Dortmund)
Aaron Tomb (Amazon)
Stefano Tonetta (FBK)

Building the First Translators[4]



[4] C. Johannsen, K. Nukala, R. Dureja, A. Irfan, N. Shankar, C. Tinelli, M. Y. Vardi, K. Y. Rozier. "Symbolic Model-Checking Intermediate-Language Tool Suite." CAV 2024.

960 QF_BV & QF_ABV benchmarks (nuXmv release)[5]



[5] K. Y. Rozier, R. Dureja, A. Irfan, C. Johannsen, K. Nukala, N. Shankar, C. Tinelli, M. Y. Vardi. "MoXI: An Intermediate Language for Symbolic Model Checking." SPIN 2024.

Join the Conversation!

OSSyM

Technical Information | CAV 2024 | Workshop Agenda | Organizational Information

July 23rd 2024

The OSSyM workshop aims to introduce the progress to-date on this collaborative effort and involve the full international research community to reduce barriers to developing new model-checking algorithms and research platforms. The workshop will include tutorials and feedback from the international Model Checking Technical Advisory Board on a design for an extensible framework centering on an intermediate language that will unify popular front-end modeling languages with state-of-the-art back-end model-checking tools.

Open-Source, State-of-the-Art Symbolic Model-Checking Framework for the Model-Checking Research Community

Project Links

Home: https://modelchecker.temporallogic.org

GitHub Organization: https://modelchecker.github.io/

MoXI Language Definition:
https://github.com/ModelChecker/IL/blob/main/description.md

CAV 2024 Workshop: OSSyM:
https://laboratory.temporallogic.org/ossym/

FMCAD 2023 Workshop:
https://github.com/ModelChecker/FMCAD23-Tutorial

SPIN 2024 paper: MoXI semantics:
https://research.temporallogic.org/papers/SPIN2024.pdf

CAV 2024 tool paper: MoXI translators:
https://research.temporallogic.org/papers/CAV2024.pdf
artifact: https://zenodo.org/records/10946779

Summary

The time has come for model-checking community standards

- **Participate:** email list, language design feedback, community forum: **OSSyM@CAV 2024**
- Available Now: **SMV** $\leftrightarrow$ **MoXI** $\leftrightarrow$ **BTOR2**
- **Contribute** future translators:
  - Your Modeling Language $\leftrightarrow$ **MoXI**
  - **MoXI** $\leftrightarrow$ Your Back-end MC Algorithm
  - **MoXI** $\leftrightarrow$ Your Proof Assistant
- **Optimize! Extend! Benchmark! Compare! Research!**

modelchecker.temporallogic.org