

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

Kristin Yvonne Rozier

Iowa State University



NSF CIRC/CCRI PI Meeting

March 10, 2025



Evolution of Model-Checking Algorithms

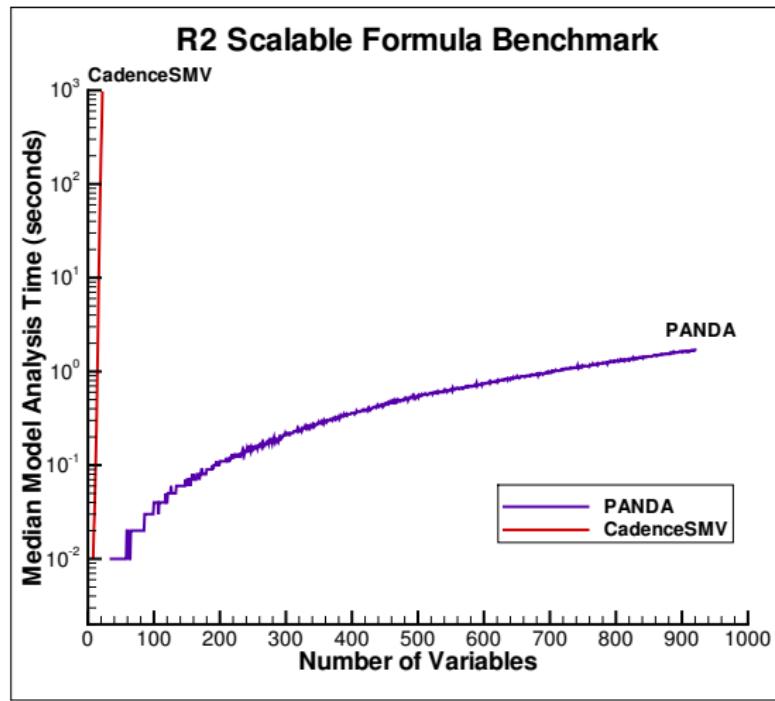
- ➊ BDD-based
- ➋ SAT-based / bounded model checking
- ➌ IC3 / k-liveness

Evolution of Model-Checking Algorithms

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

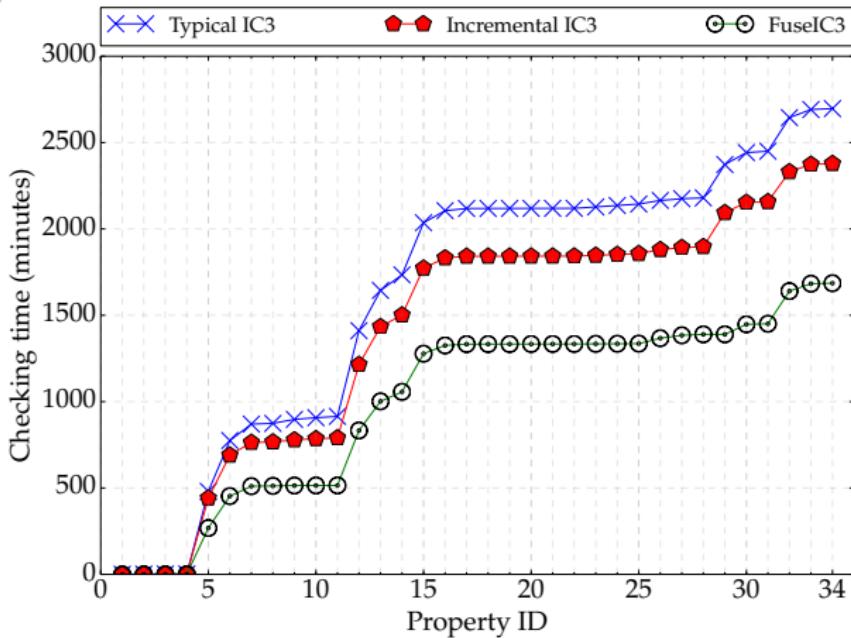
Symbolic model checking progressed from bit-level to word level

The Problem¹



¹ 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 Spaces²



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

² 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

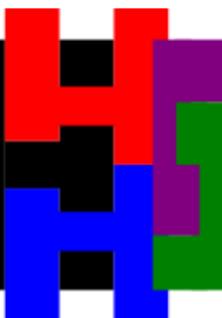
- nuXmv **closed source**: can't check internal algorithm
- IBM's tools **closed source**: same problem
- **Can't build** on either tool
- Have **models in SMV** language: can't MC with IBM's algorithm
- Have **new algorithm**; also can't check SMV models with it

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

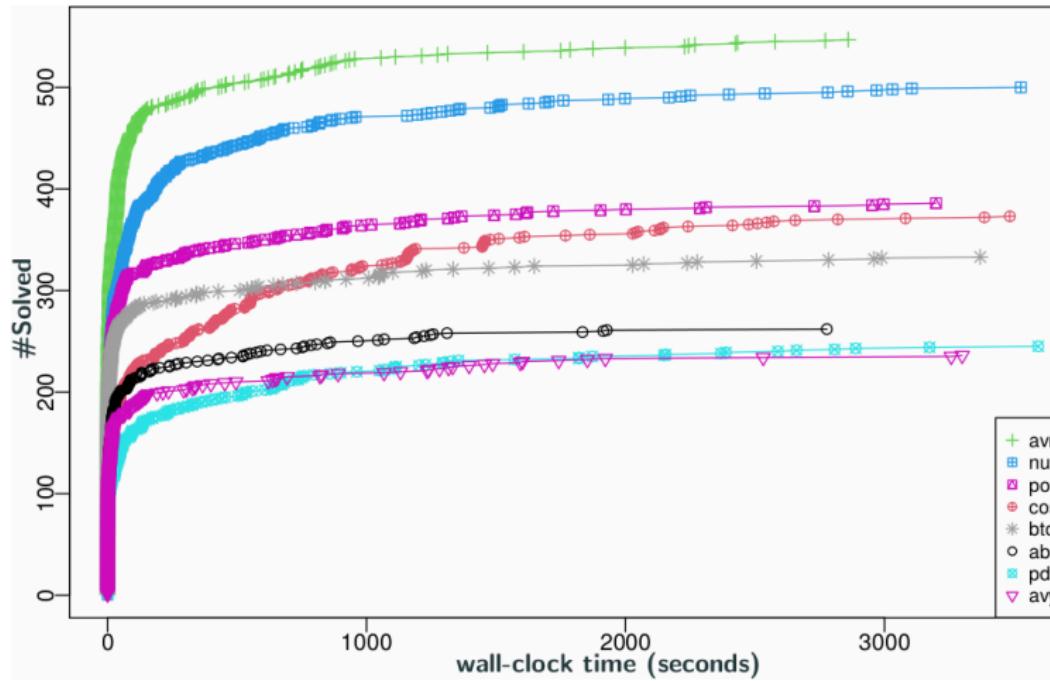
- not publishable, LOTS of time, hard to get right, waste of effort

How do we do better?

HWMCC: Hardware Model Checking Competition (2020)

Word-level
reasoning

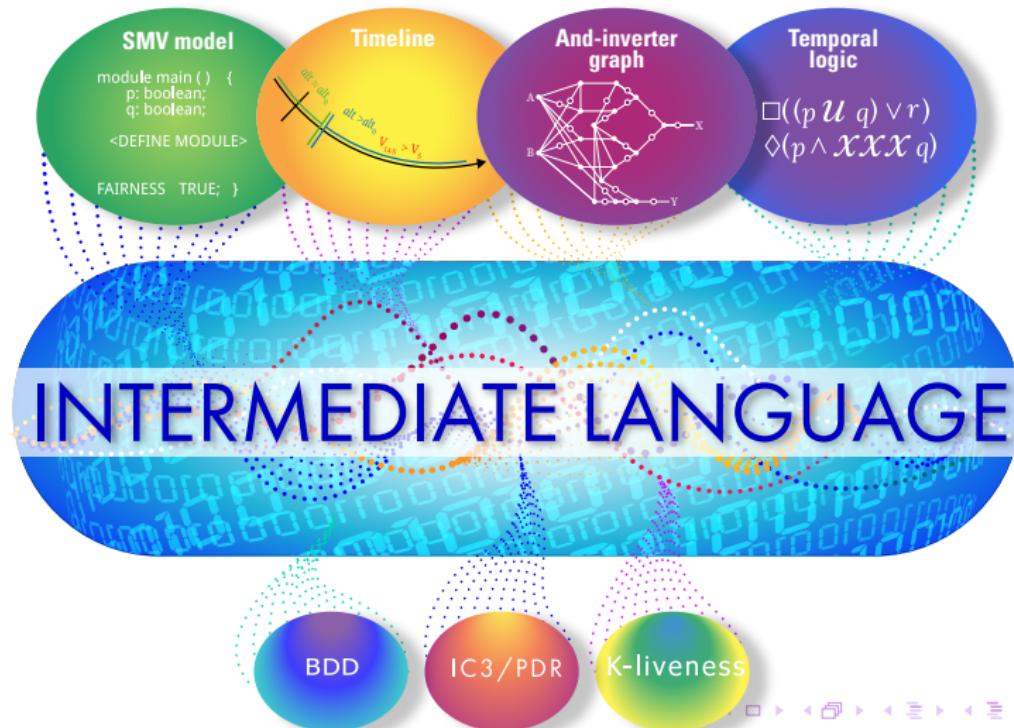
BTOR2



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

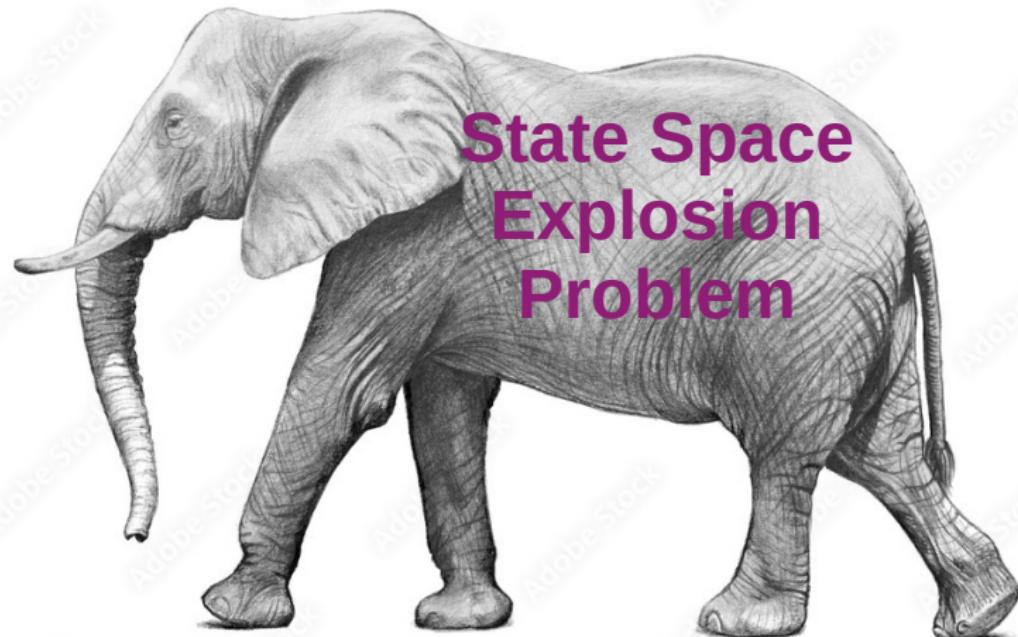


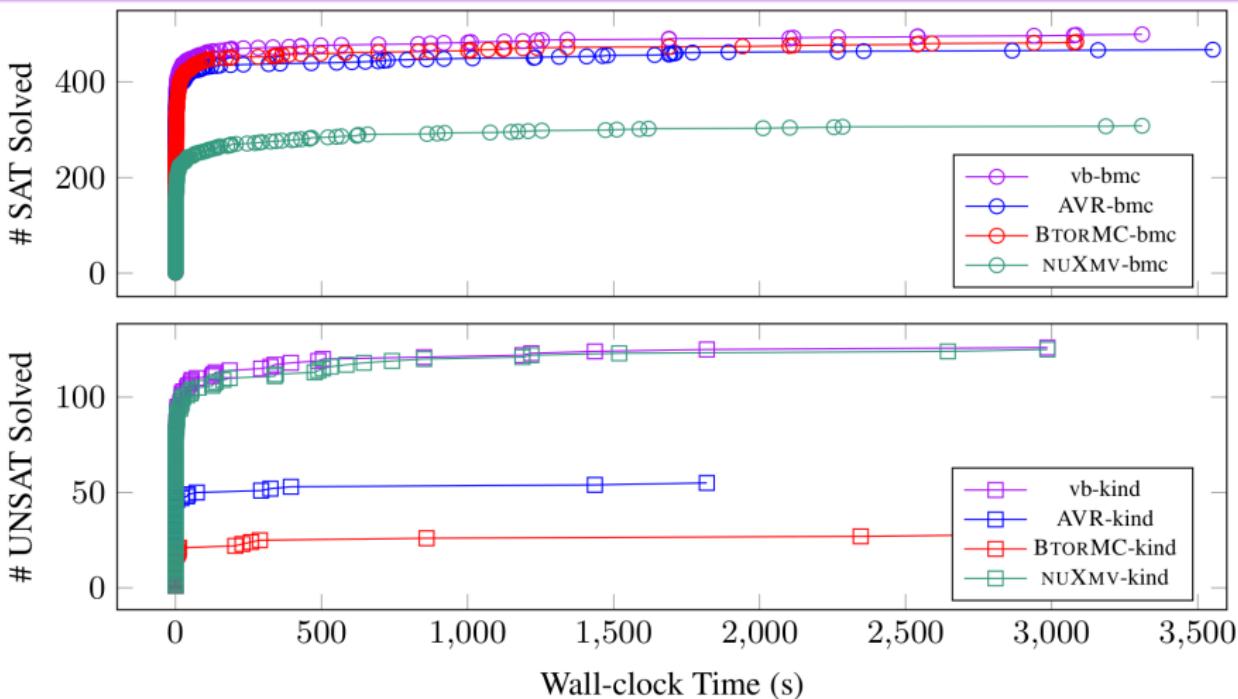
Goals for Intermediate Language

- 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**

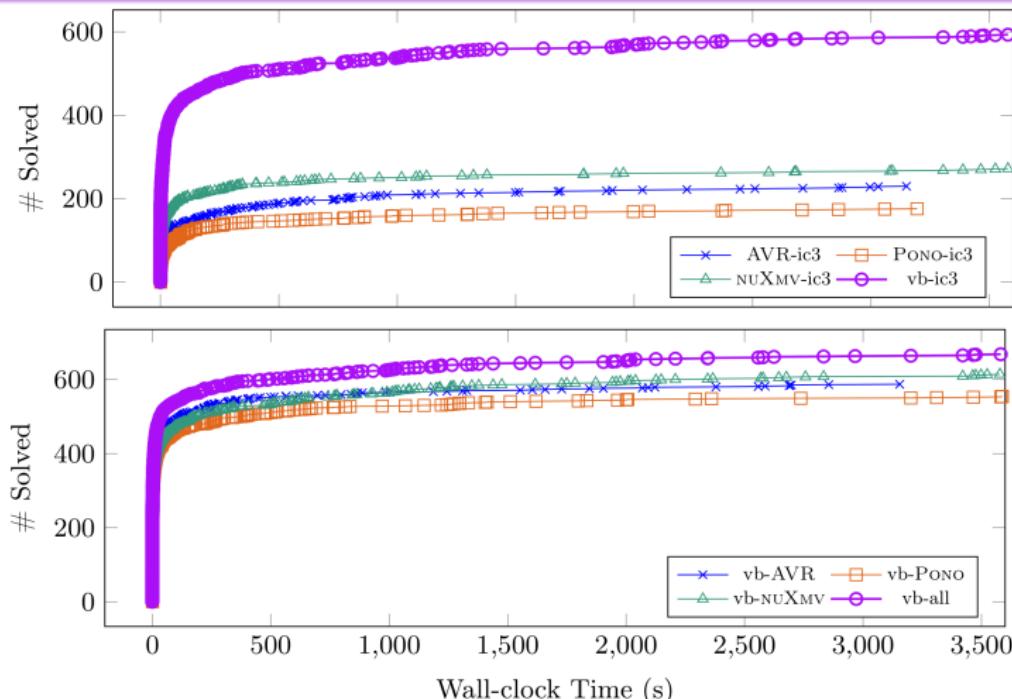
Basis for MoXI as the Intermediate Language

- “**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)



Experiments: 960 Boolean/word/array SMV Models³

³ 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)⁴

⁴ 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.

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>

MoXIchecker: An Extensible Model Checker for MoXI:

<https://gitlab.com/sosy-lab/software/moxichecker>

BACKUP SLIDES

Core Design Team

Investigators:



K.Y. Rozier



Natarajan Shankar



Cesare Tinelli



Moshe Vardi

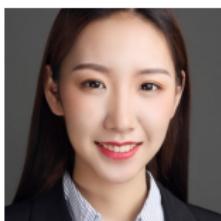
Students:



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)

MoXI: Base Logic

- **Many-sorted first-order logic** w/ equality, quantifiers, let binders, algebraic datatypes (like SMT-LIB V2)
- Extension to (first-order) **temporal logic** w/ discrete, linear time, finite & infinite trace-based semantics
- Commands for defining and verifying **multi-component reactive systems**
- Checks for **reachability conditions, deadlocks** (indirectly: state & transition invariants)
- Includes **fairness conditions** on system inputs

Language Design Highlights: Transition Systems

Transition system $S = (I_S[i, o, s], \quad T_S[i, o, s, i', o', s'])$

where

- i and i' are two tuples of *input variables* with the same length and type;
- o and o' are two tuples of *output variables* with the same length and type;
- s and s' are two tuples of *local variables* with the same length and type;
- I_S is the system's *initial state condition*, expressed as a formula with no (free) variables from i' , o' , and s' ;
- T_S is the system's *transition condition*, expressed as a formula over the variables i , o , s , i' , o' , and s' .

Language Design Highlights: System Checking Command

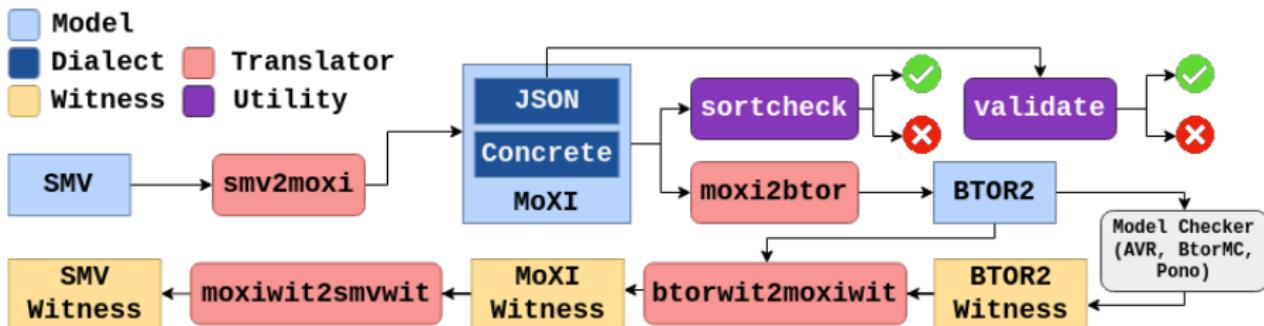
```
(check-system  $S$ 
  :input ( (  $i_1 \delta_1$  )  $\cdots$  (  $i_m \delta_m$  ) )
  :output ( (  $o_1 \tau_1$  )  $\cdots$  (  $o_n \tau_n$  ) )
  :local ( (  $s_1 \sigma_1$  )  $\cdots$  (  $s_p \sigma_p$  ) )
  :assumption (  $a A$  )
  :fairness (  $f F$  )
  :reachable (  $r R$  )
  :current (  $c C$  )
  :query (  $q ( g_1 \cdots g_q )$  )
  :queries ( (  $q_1 ( g_{1,1} \cdots g_{1,n_1} )$  )  $\cdots$  (  $q_t ( g_{t,1} \cdots g_{t,n_t} )$  ) )
)
```

Language Design Highlights: Check System Response

```
(check-system-response
:query (q1 :result sat :model m :trace t)
:query (q2 :result unsat :certificate c)
:query (q3 :result unknown) ; for timeouts and other cases
:trace (t :prefix p :lasso l) ; t = pl^omega
:model (m M) ; M is model in SMT-LIB format

; state/valuation
:trail (p ( ((i i_0) (o o_0) (s s_0) (r r_0) (f f_0))
...
((i i_k) (o o_k) (s s_k) (r r_k) (f f_k))
)
)
:trail (l ( ( ... ) ... ( ... ) ))
:certificate (c :inv F :k n)
)
```

Building the First Translators⁵



⁵ 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.

Summary

The time has come for model-checking community standards

- **Participate:** email list, language design feedback, online community forum, in-person workshops
- Available Now: **SMV \leftrightarrow MoXI \leftrightarrow BTOR2**
MoXIchecker: An Extensible Model Checker for MoXI
- **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

