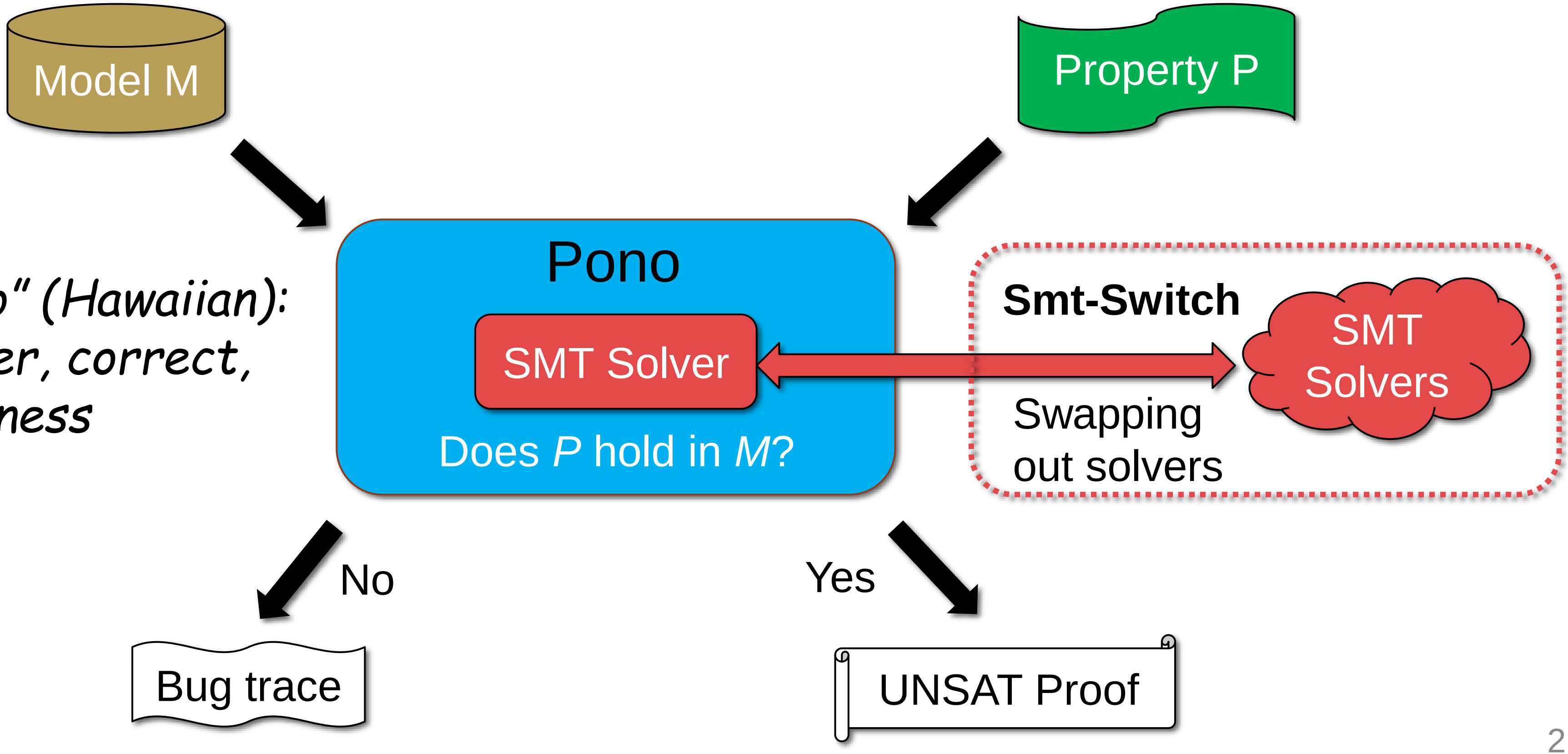
Pono: An SMT-Based Model Checker

FLORIAN LONSING

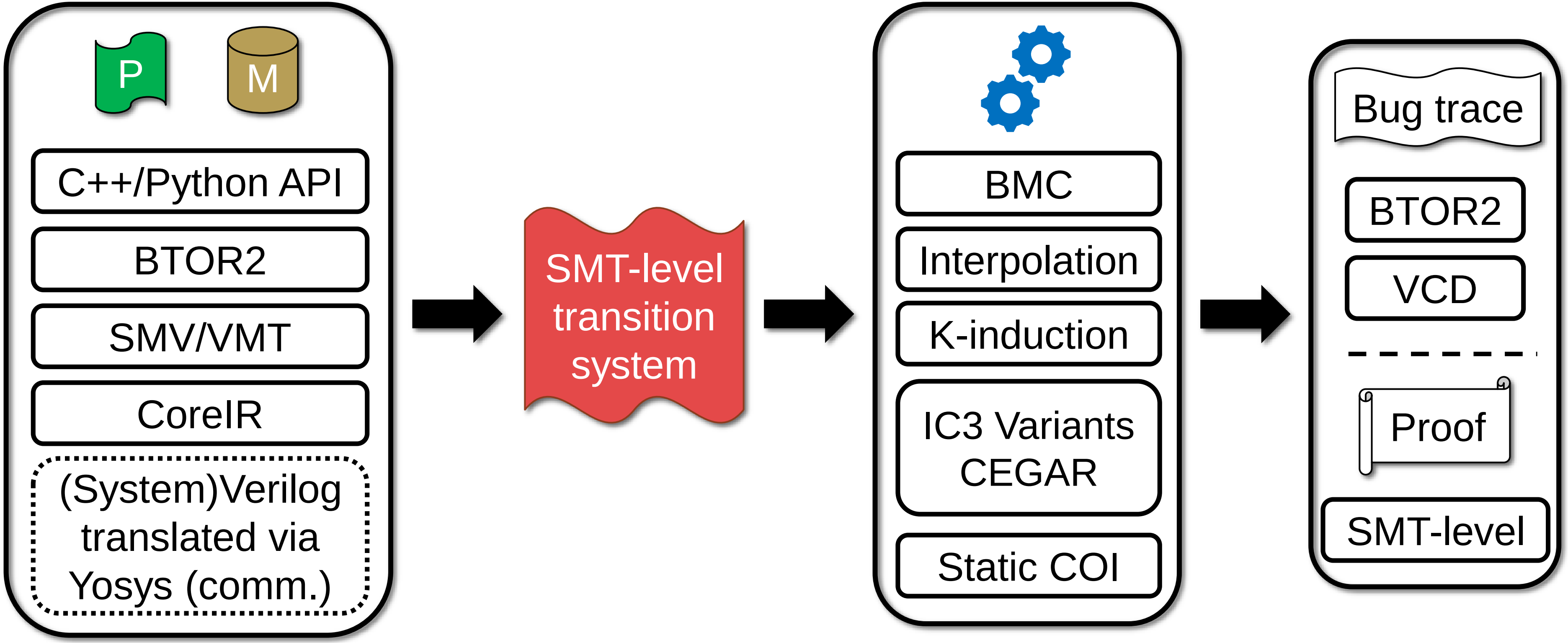
Joint work with Makai Mann, Ahmed Irfan, Yahan Yang,
Hongce Zhang, Kristopher Brown, Aarti Gupta, and Clark Barrett

Pono: Open-Source Model Checker

"Pono" (Hawaiian): proper, correct, goodness



High-Level Workflow



P

M

C++/Python API

BTOR2

SMV/VMT

CoreIR

(System)Verilog translated via Yosys (comm.)

SMT-level transition system

BMC

Interpolation

K-induction

IC3 Variants CEGAR

Static COI

Bug trace

BTOR2

VCD

Proof

SMT-level

Example: Abstraction-Based Verification

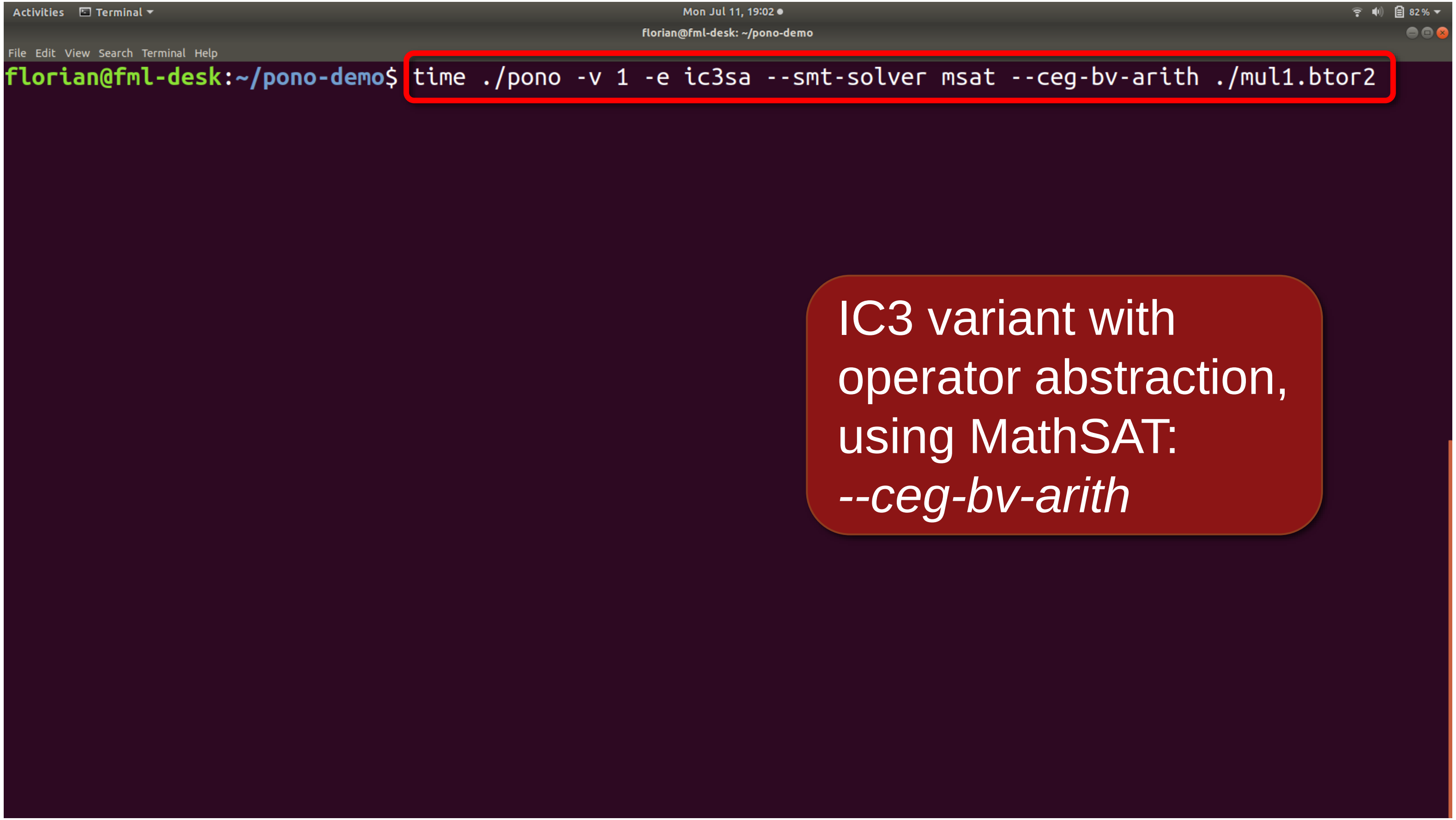
```
; BTOR description generated by Yosys, 'bv/2019/goel/industry/mul1/mul1.btor2'
1 sort bitvec 1
...
5 sort bitvec 32
...
9 sort bitvec 64
...
11 state 9
...
23 state 5
...
27 state 1
...
38 mul 9 36 37
...
```

Definition of bitvector sorts of size 1, 32, 64

Definition of bitvector state variables

64-bit multiplication used in state update

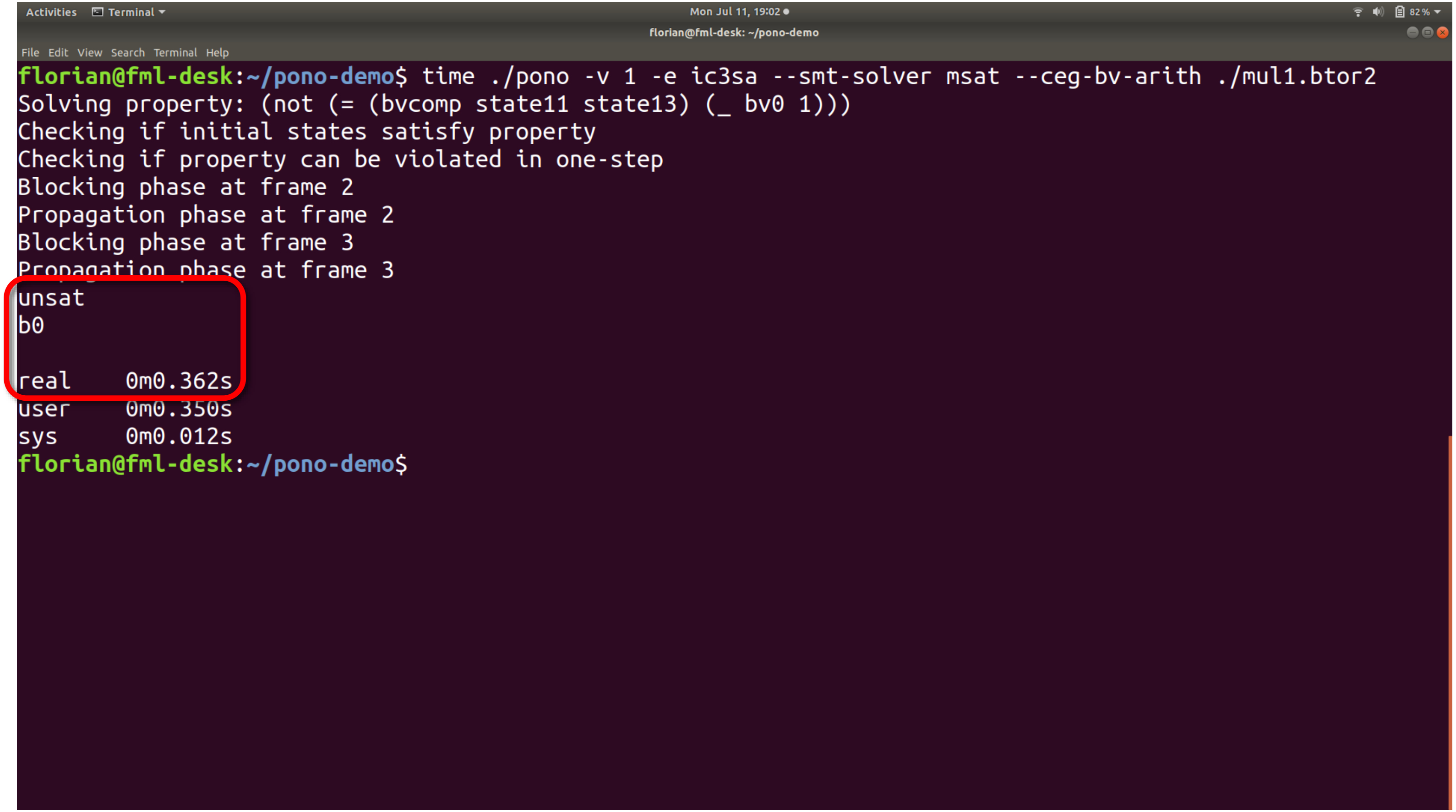
Example: Abstraction-Based Verification


Activities
Terminal
Mon Jul 11, 19:02
82 %
florian@fml-desk: ~/pono-demo
File Edit View Search Terminal Help

```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e ic3sa --smt-solver msat --ceg-bv-arith ./mul1.btor2
```

IC3 variant with operator abstraction, using MathSAT: *--ceg-bv-arith*

Example: Abstraction-Based Verification

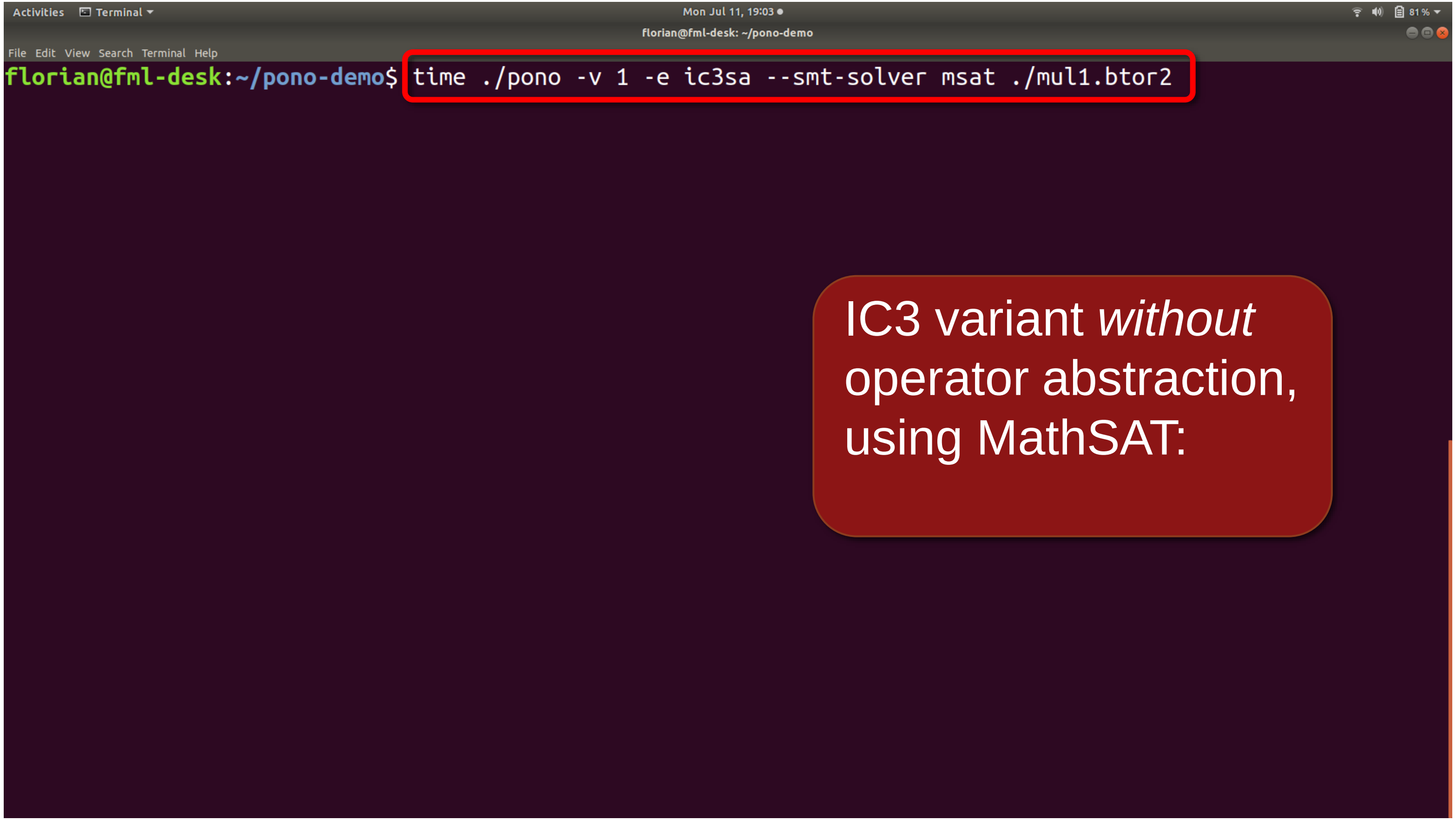


Activities
Terminal
Mon Jul 11, 19:02
82 %
florian@fml-desk: ~/pono-demo
File Edit View Search Terminal Help

```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e ic3sa --smt-solver msat --ceg-bv-arith ./mul1.btor2
Solving property: (not (= (bvcomp state11 state13) (_ bv0 1)))
Checking if initial states satisfy property
Checking if property can be violated in one-step
Blocking phase at frame 2
Propagation phase at frame 2
Blocking phase at frame 3
Propagation phase at frame 3
unsat
b0

real	0m0.362s
user	0m0.350s
sys	0m0.012s
florian@fml-desk:~/pono-demo$
```

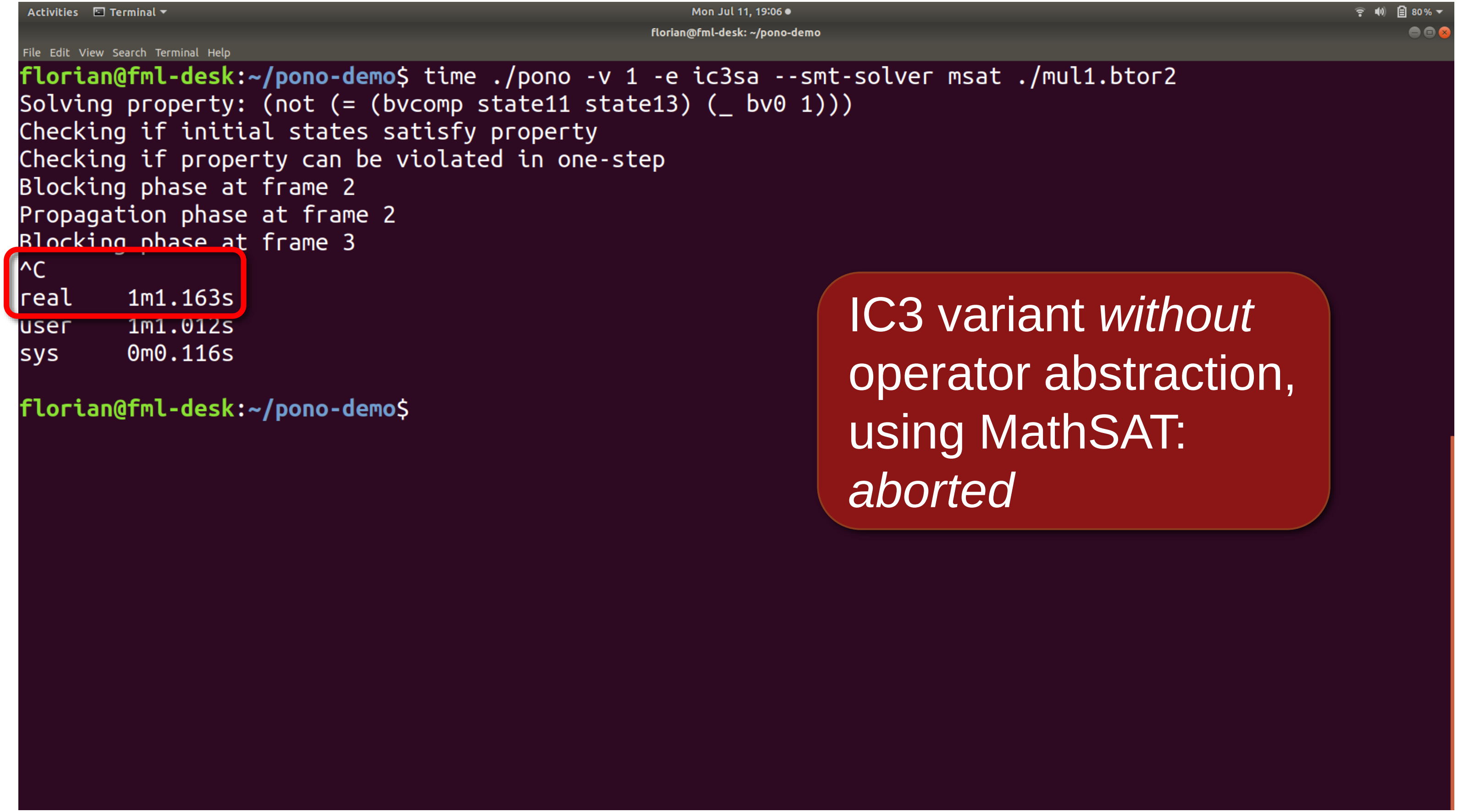
Example: Abstraction-Based Verification


Activities
Terminal
Mon Jul 11, 19:03
81 %
florian@fml-desk: ~/pono-demo
File Edit View Search Terminal Help

```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e ic3sa --smt-solver msat ./mul1.btor2
```

IC3 variant *without* operator abstraction, using MathSAT:

Example: Abstraction-Based Verification



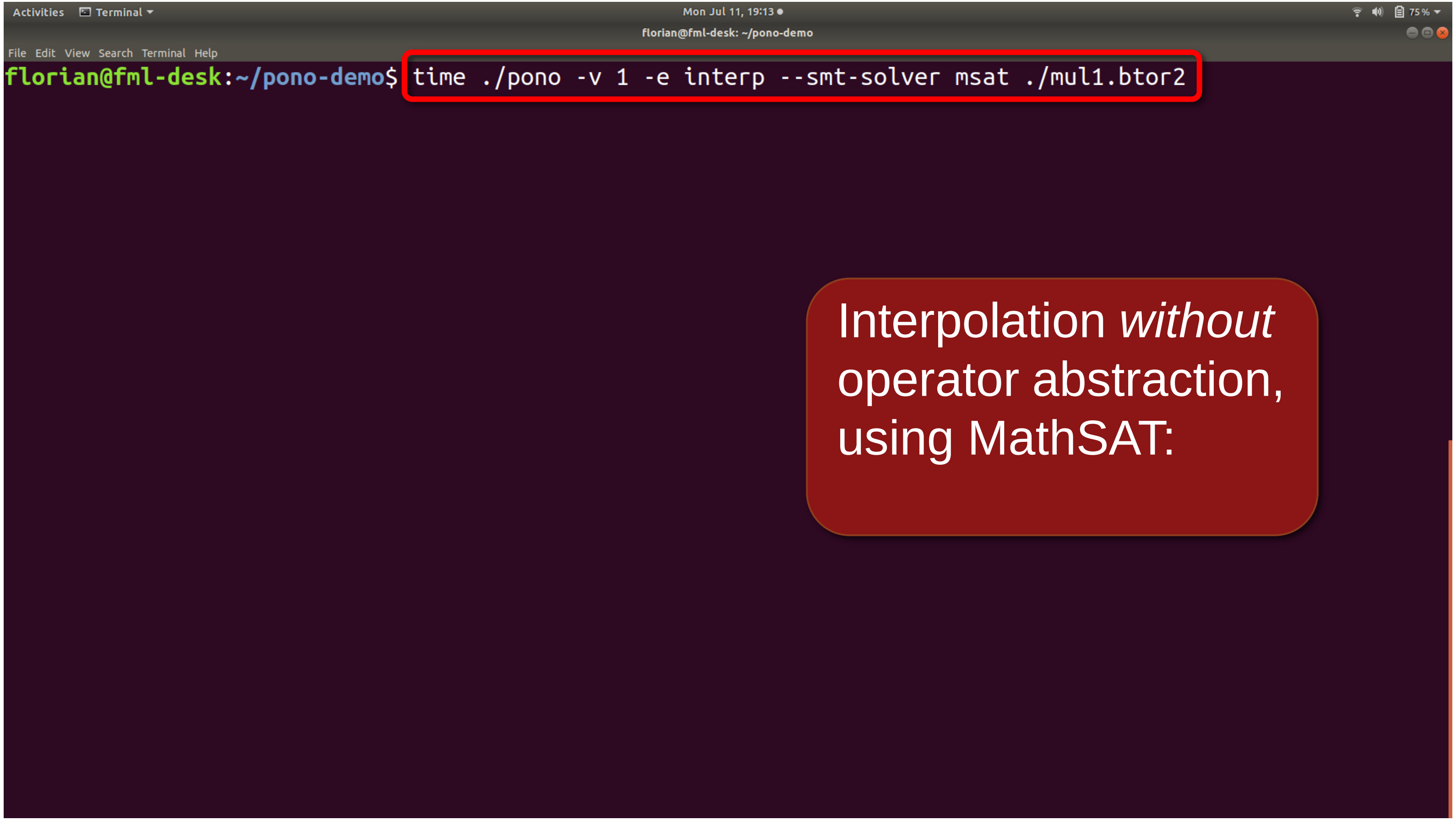
Activities
Terminal
Mon Jul 11, 19:06
80 %
florian@fml-desk: ~/pono-demo
File Edit View Search Terminal Help

```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e ic3sa --smt-solver msat ./mul1.btor2
Solving property: (not (= (bvcomp state11 state13) (_ bv0 1)))
Checking if initial states satisfy property
Checking if property can be violated in one-step
Blocking phase at frame 2
Propagation phase at frame 2
Blocking phase at frame 3
^C
real	1m1.163s
user	1m1.012s
sys	0m0.116s

florian@fml-desk:~/pono-demo$
```

IC3 variant *without* operator abstraction, using MathSAT: *aborted*

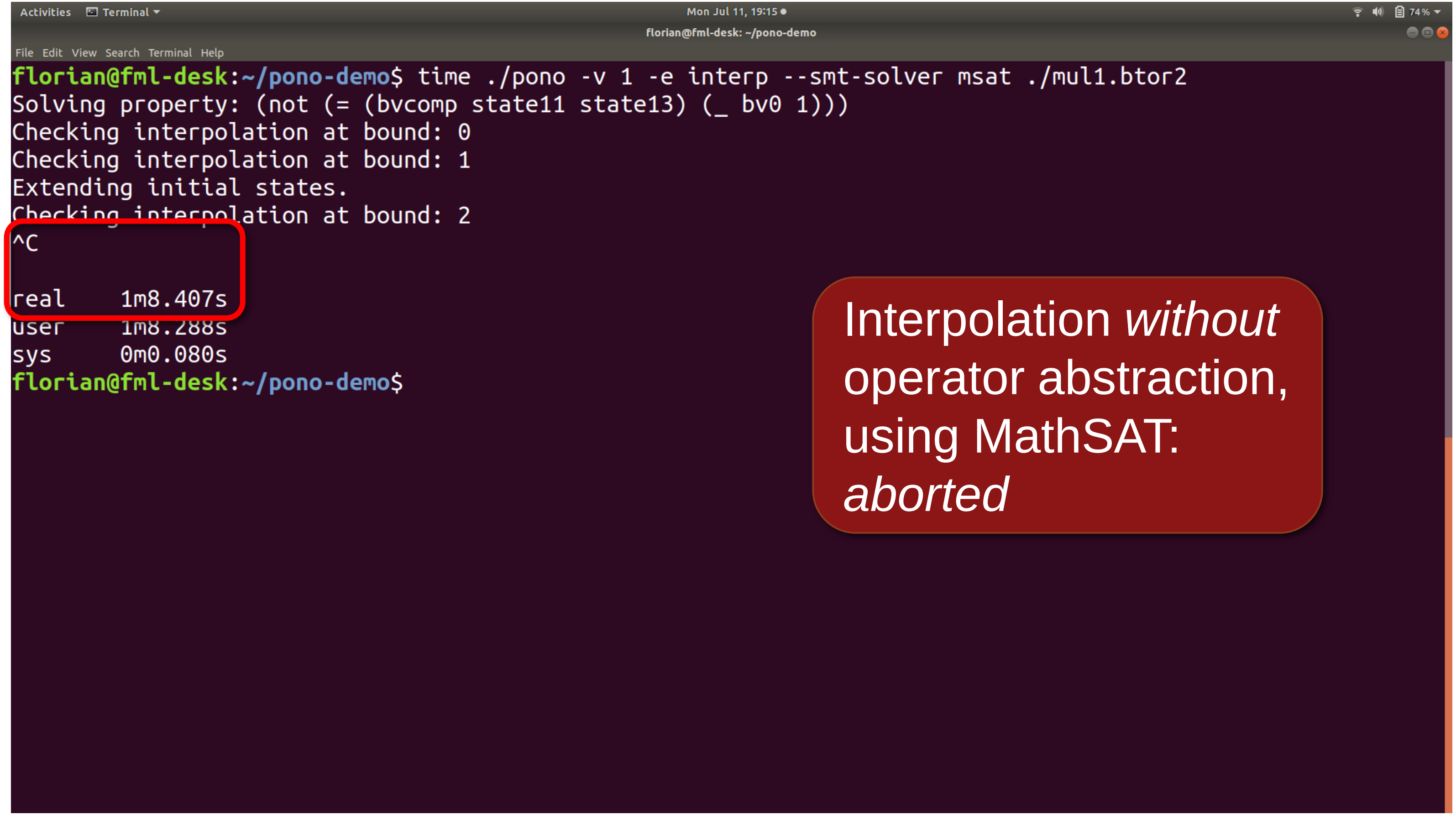
Example: Abstraction-Based Verification


Activities
Terminal
Mon Jul 11, 19:13
75 %
florian@fml-desk: ~/pono-demo
File Edit View Search Terminal Help

```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e interp --smt-solver msat ./mul1.btor2
```

Interpolation *without* operator abstraction, using MathSAT:

Example: Abstraction-Based Verification

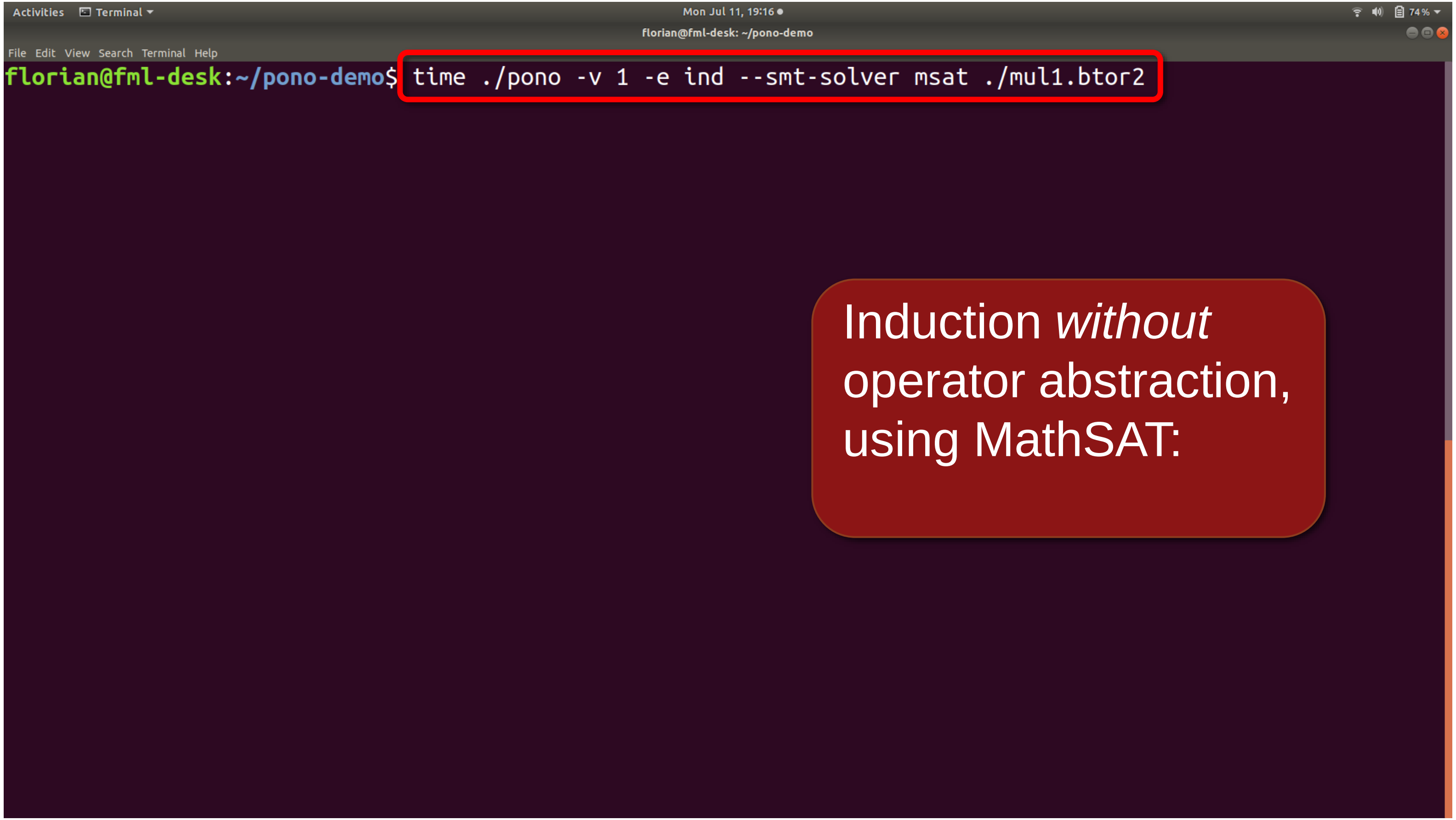


```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e interp --smt-solver msat ./mul1.btor2
Solving property: (not (= (bvcomp state11 state13) (_ bv0 1)))
Checking interpolation at bound: 0
Checking interpolation at bound: 1
Extending initial states.
Checking interpolation at bound: 2
^C

real	1m8.407s
user	1m8.288s
sys	0m0.080s
florian@fml-desk:~/pono-demo$
```

Interpolation *without* operator abstraction, using MathSAT: *aborted*

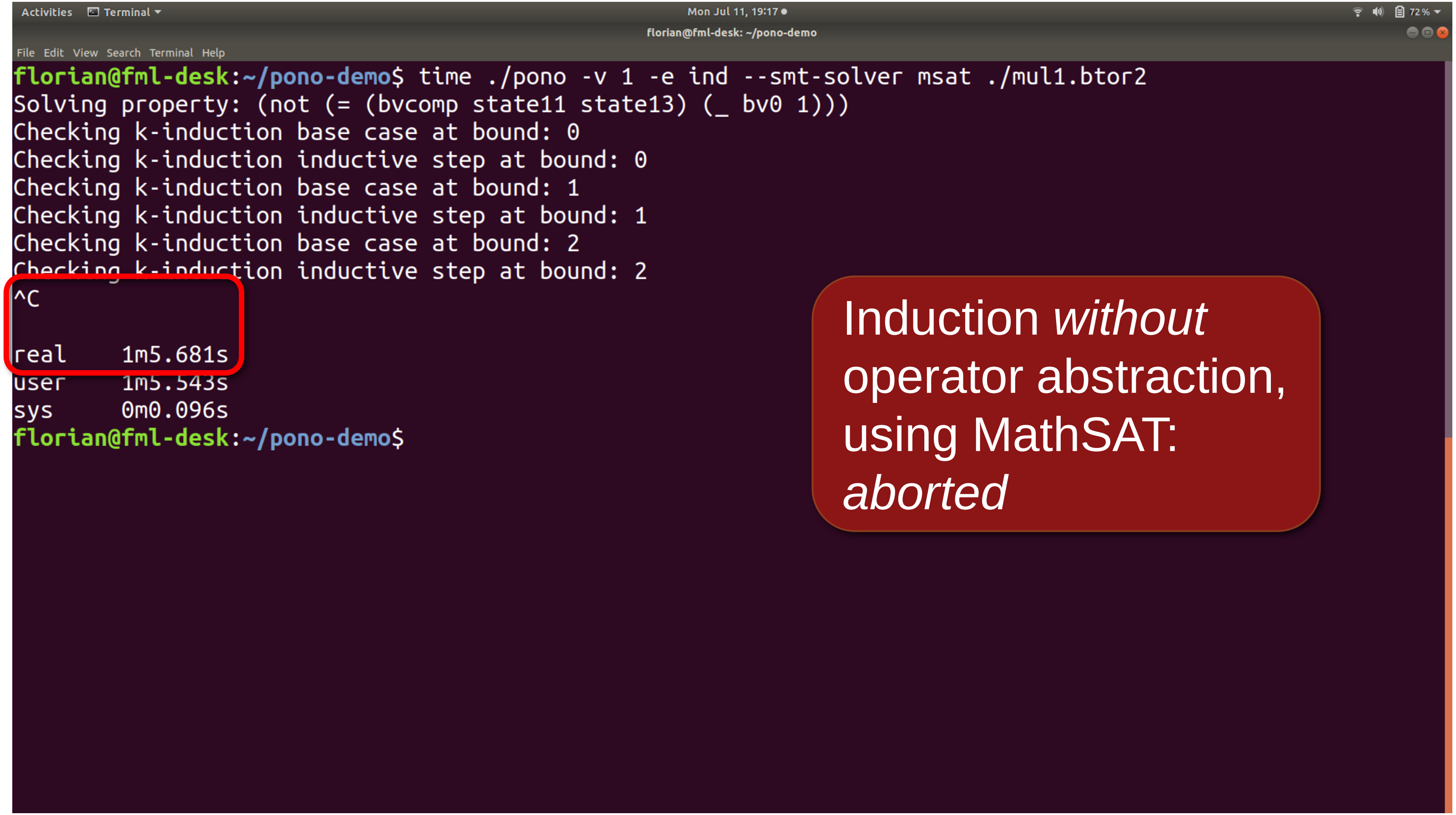
Example: Abstraction-Based Verification


Activities
Terminal
Mon Jul 11, 19:16
74 %
florian@fml-desk: ~/pono-demo
File Edit View Search Terminal Help

```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e ind --smt-solver msat ./mul1.btor2
```

Induction *without* operator abstraction, using MathSAT:

Example: Abstraction-Based Verification



Activities
Terminal
Mon Jul 11, 19:17
72 %
florian@fml-desk: ~/pono-demo
File Edit View Search Terminal Help

```
florian@fml-desk:~/pono-demo$ time ./pono -v 1 -e ind --smt-solver msat ./mul1.btor2
Solving property: (not (= (bvcomp state11 state13) (_ bv0 1)))
Checking k-induction base case at bound: 0
Checking k-induction inductive step at bound: 0
Checking k-induction base case at bound: 1
Checking k-induction inductive step at bound: 1
Checking k-induction base case at bound: 2
Checking k-induction inductive step at bound: 2
^C

real	1m5.681s
user	1m5.543s
sys	0m0.096s
florian@fml-desk:~/pono-demo$
```

Induction *without* operator abstraction, using MathSAT: *aborted*

Open-Source Model Checkers

Hardware Model Checking Competitions (HWMCC):

- SMT encodings of HW verification problems.
- Bitvectors + arrays to compactly model word-level problems.
- 639 HWMCC’20 benchmarks:

Tool	#Solved	#SAT	#UNSAT
AVR (HWMCC'20 winner)	547	66	481
Pono (Cosa2 successor)	386	58	328
Cosa2 (HWMCC'19 winner)	373	58	315
Pono-BMC (SAT only)	84	84	0

Open-Source Model Checkers

Hardware Model Checking Competitions (HWMCC):

- SMT encodings of HW verification problems.
- Bitvectors + arrays to compactly model word-level problems.
- 639 HWMCC’20 benchmarks:

Tool	#Solved	#SAT	#UNSAT
AVR (HWMCC'20 winner)	547	66	481
Pono (Cosa2 successor)	386	58	328
Cosa2 (HWMCC'19 winner)	373	58	315
Pono-BMC (SAT only)	84	84	0

Open-Source Model Checkers

Hardware Model Checking Competitions (HWMCC):

- SMT encodings of HW verification problems.
- Bitvectors + arrays to compactly model word-level problems.
- 639 HWMCC’20 benchmarks:

Tool	#Solved	#SAT	#UNSAT
AVR (HWMCC'20 winner)	547	66	481
Pono (Cosa2 successor)	386	58	328
Cosa2 (HWMCC'19 winner)	373	58	315
Pono-BMC (SAT only)	84	84	0

Open-Source Model Checkers

Hardware Model Checking Competitions (HWMCC):

- SMT encodings of HW verification problems.
- Bitvectors + arrays to compactly model word-level problems.
- 639 HWMCC’20 benchmarks:

Tool	#Solved	#SAT	#UNSAT
AVR (HWMCC'20 winner)	547	66	481
Pono (Cosa2 successor)	386	58	328
Cosa2 (HWMCC'19 winner)	373	58	315
Pono-BMC (SAT only)	84	84	0

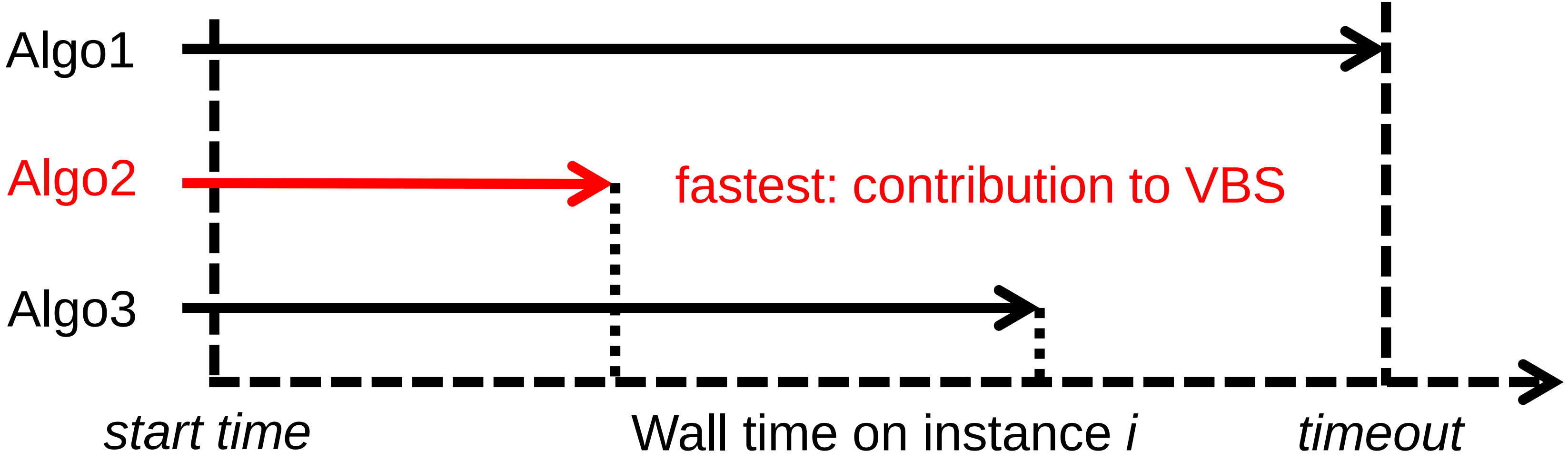
Open-Source Model Checkers

Hardware Model Checking Competitions (HWMCC):

- SMT encodings of HW verification problems.
- Bitvectors + arrays to compactly model word-level problems.
- 639 HWMCC’20 benchmarks:

Parallel Portfolios	#Solved	#SAT	#UNSAT
AVR (HWMCC'20 winner)	547	66	481
Pono (Cosa2 successor)	386	58	328
Cosa2 (HWMCC'19 winner)	373	58	315
Pono-BMC (SAT only)	84	84	0

Virtual Best Solver (VBS) Analysis



- How to analyze performance of (parallel) algorithm portfolios?
- Run all algorithms sequentially on all instances.
- Fastest algorithm on an instance *i* contributes to VBS.

Open-Source Tools vs. Commercial Tool

- VBS of algorithm portfolios consisting of AVR and Pono.
- 13 resp. 16 (variants of) algorithms in Pono and AVR.
- Commercial model checker “CMC”: VBS of 21 algorithms.
- 639 instances, 1h wall time, 32 GB, failures count as timeouts:

VBS	#Solved	#SAT	#UNSAT	Time (wall sec.)
CMC	582	87	495	233,019
AVR-Pono	576	87	489	274,728
AVR	553	66	487	368,350
Pono	508	86	422	585,708

Open-Source Tools vs. Commercial Tool

- VBS of algorithm portfolios consisting of AVR and Pono.
- 13 resp. 16 (variants of) algorithms in Pono and AVR.
- Commercial model checker “CMC”: VBS of 21 algorithms.
- 639 instances, 1h wall time, 32 GB, failures count as timeouts:

VBS	#Solved	#SAT	#UNSAT	Time (wall sec.)
CMC	582	87	495	233,019
AVR-Pono	576	87	489	274,728
AVR	553	66	487	368,350
Pono	508	86	422	585,708

1.59X

Open-Source Tools vs. Commercial Tool

- VBS of algorithm portfolios consisting of AVR and Pono.
- 13 resp. 16 (variants of) algorithms in Pono and AVR.
- Commercial model checker “CMC”: VBS of 21 algorithms.
- 639 instances, 1h wall time, 32 GB, failures count as timeouts:

VBS	#Solved	#SAT	#UNSAT	Time (wall sec.)
CMC	582	87	495	233,019
AVR-Pono	576	87	489	274,728
AVR	553	66	487	368,350
Pono	508	86	422	585,708

1.34X

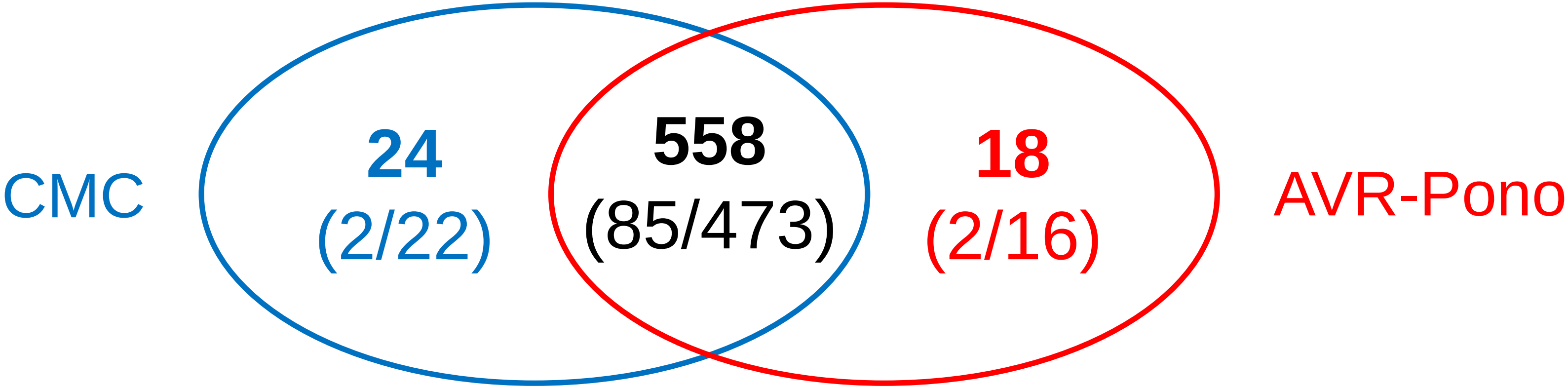
Open-Source Tools vs. Commercial Tool

- VBS of algorithm portfolios consisting of AVR and Pono.
- 13 resp. 16 (variants of) algorithms in Pono and AVR.
- Commercial model checker “CMC”: VBS of 21 algorithms.
- 639 instances, 1h wall time, 32 GB, failures count as timeouts:

VBS	#Solved	#SAT	#UNSAT	Time (wall sec.)
CMC	582	87	495	233,019
AVR-Pono	576	87	489	274,728
AVR	553	66	487	368,350
Pono	508	86	422	585,708

1.18X

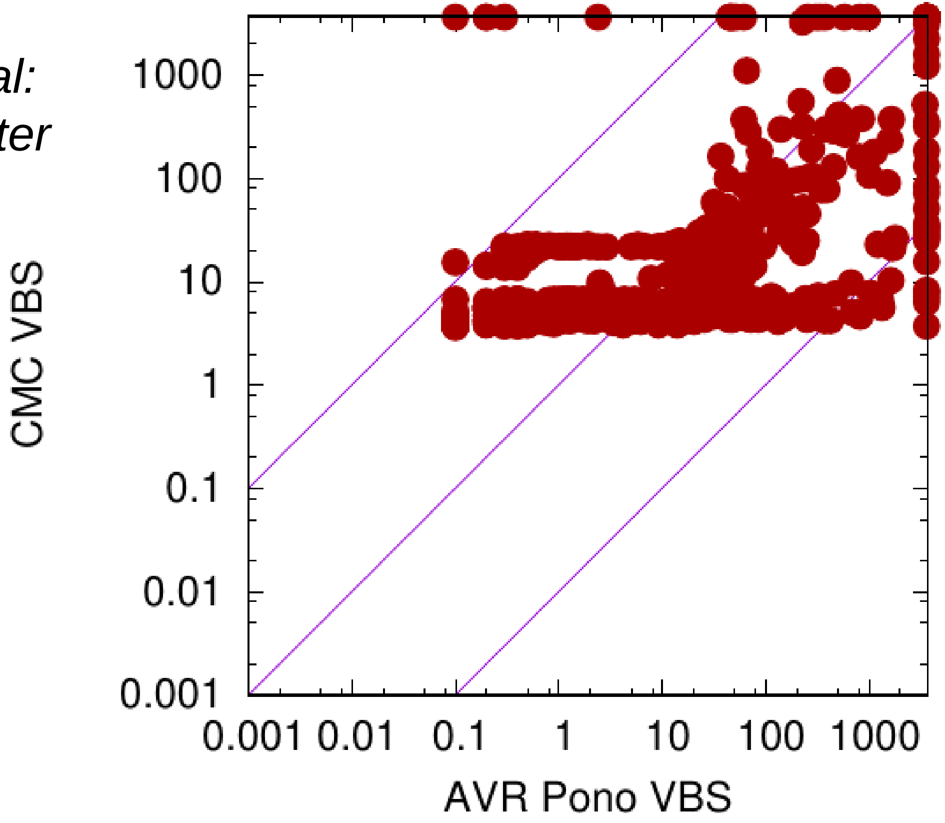
Solved Instances (SAT/UNSAT)



VBS	#Solved	#SAT	#UNSAT	Time (wall sec.)
CMC	582	87	495	233,019
AVR-Pono	576	87	489	274,728

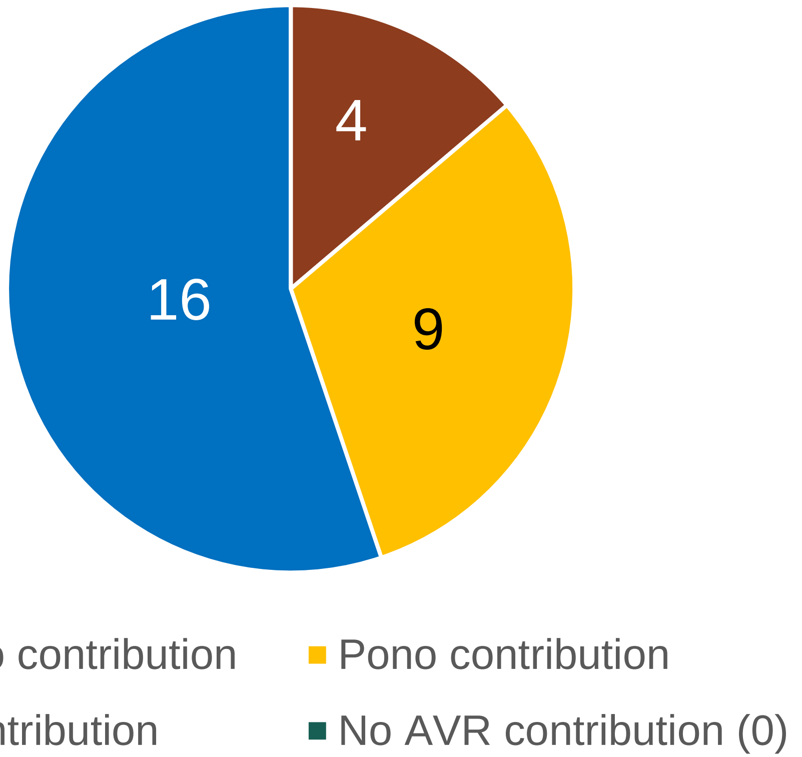
Scatter Plot: CMC vs. AVR-Pono

Above diagonal:
AVR-Pono faster



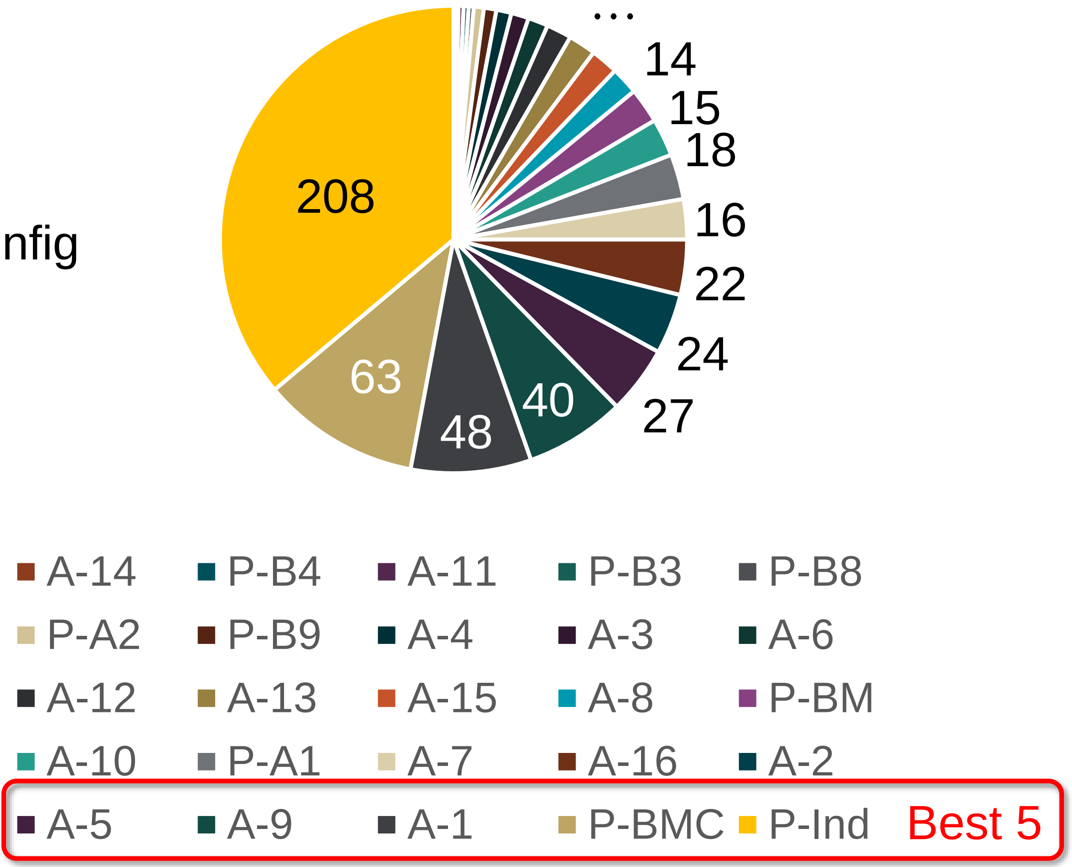
Below diagonal:
CMC faster

VBS: 29 Algorithms/Configurations



VBS: Contributions to 576 Solved Instances

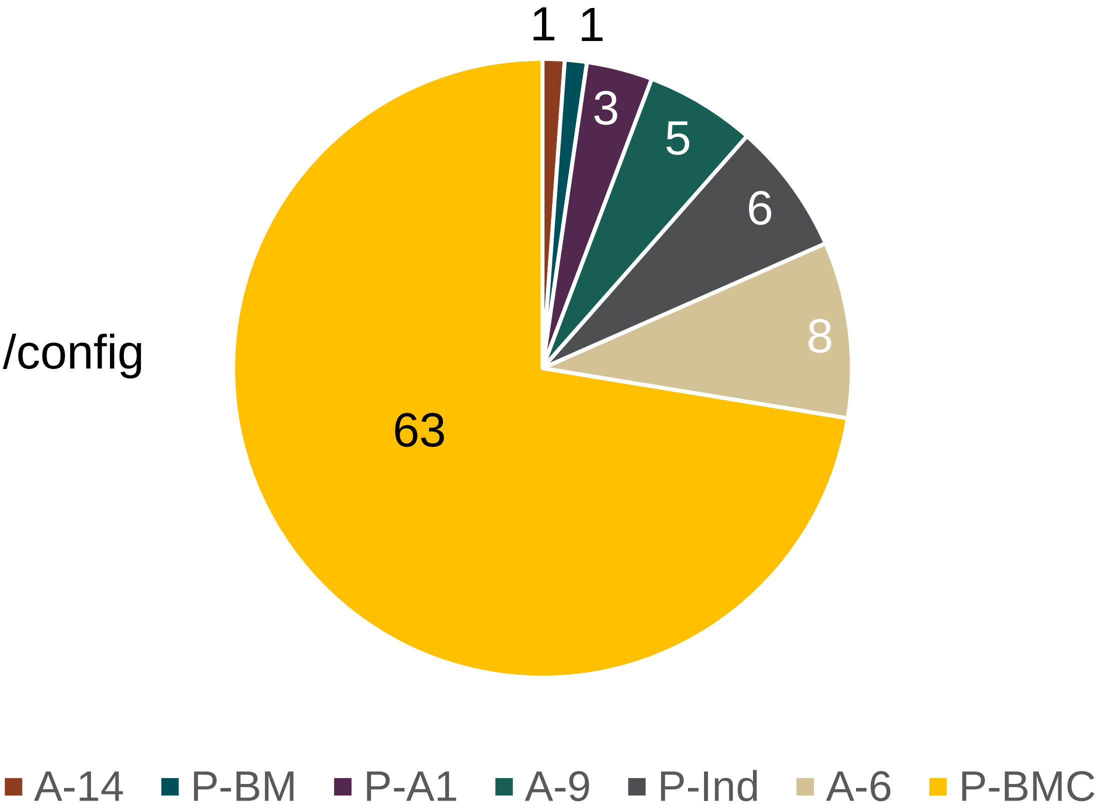
slice ≈ algorithm/config



Best 5

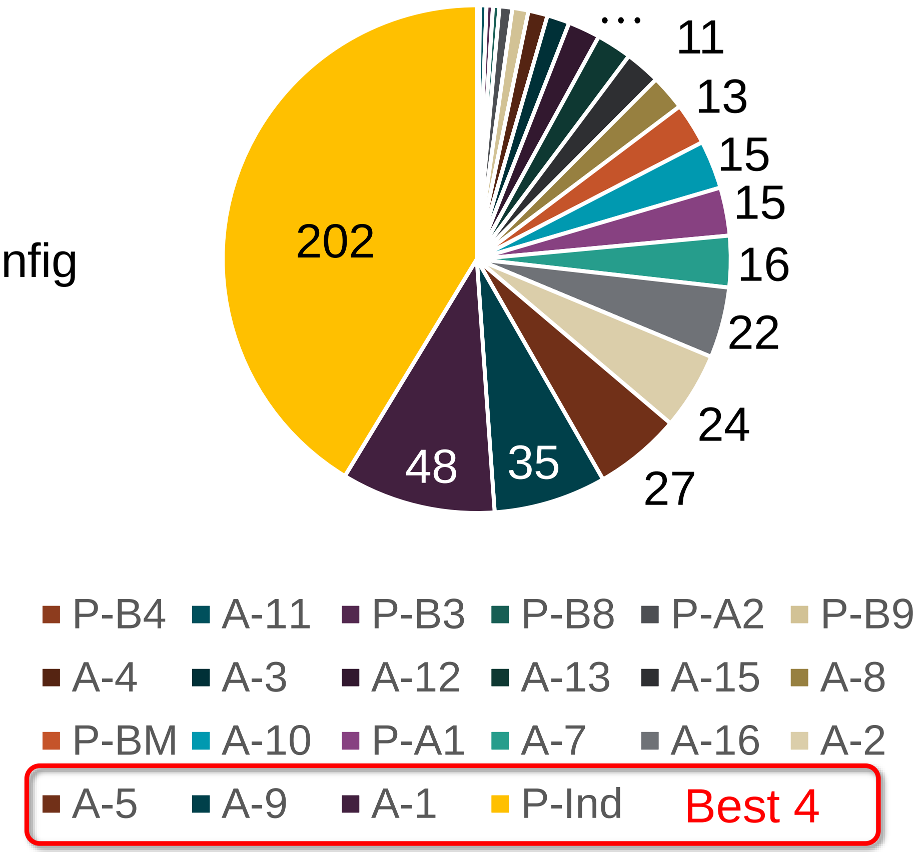
VBS: Contributions to 87 SAT Instances

slice ≈ algorithm/config



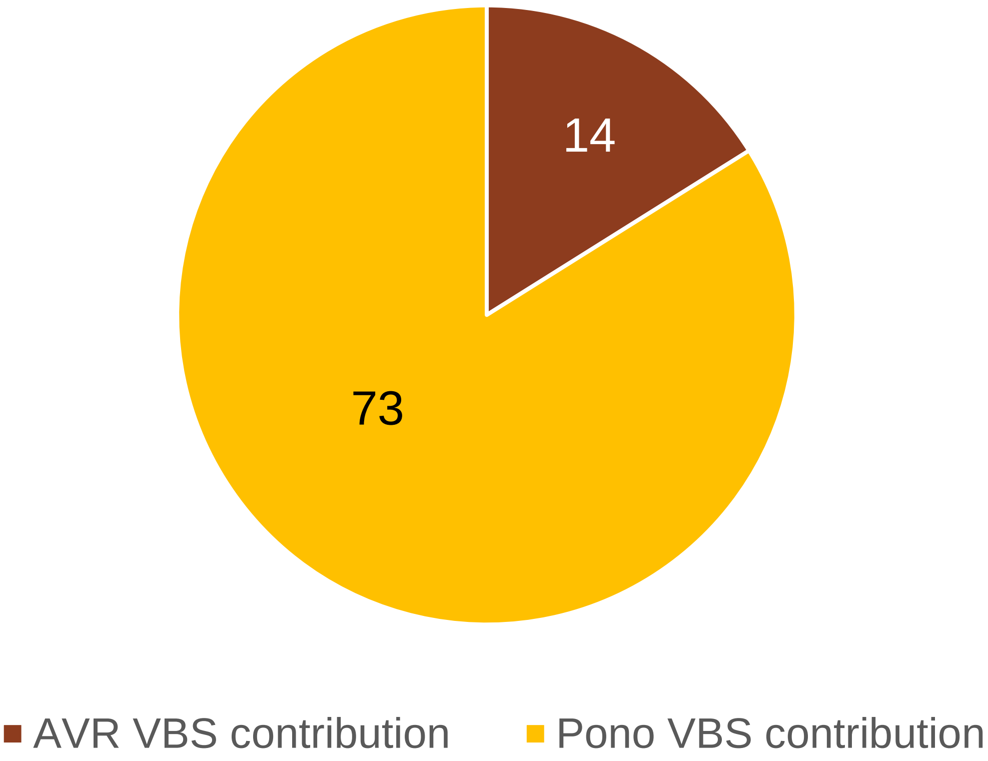
VBS: Contributions to 489 UNSAT Instances

slice ≈ algorithm/config

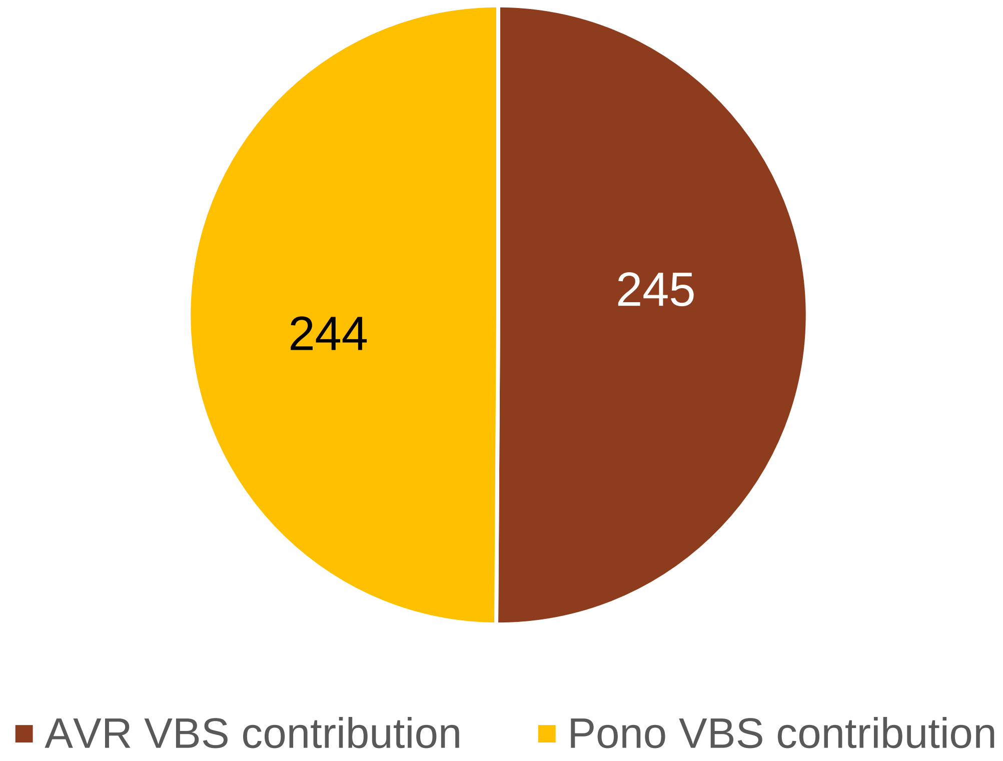


Best 4

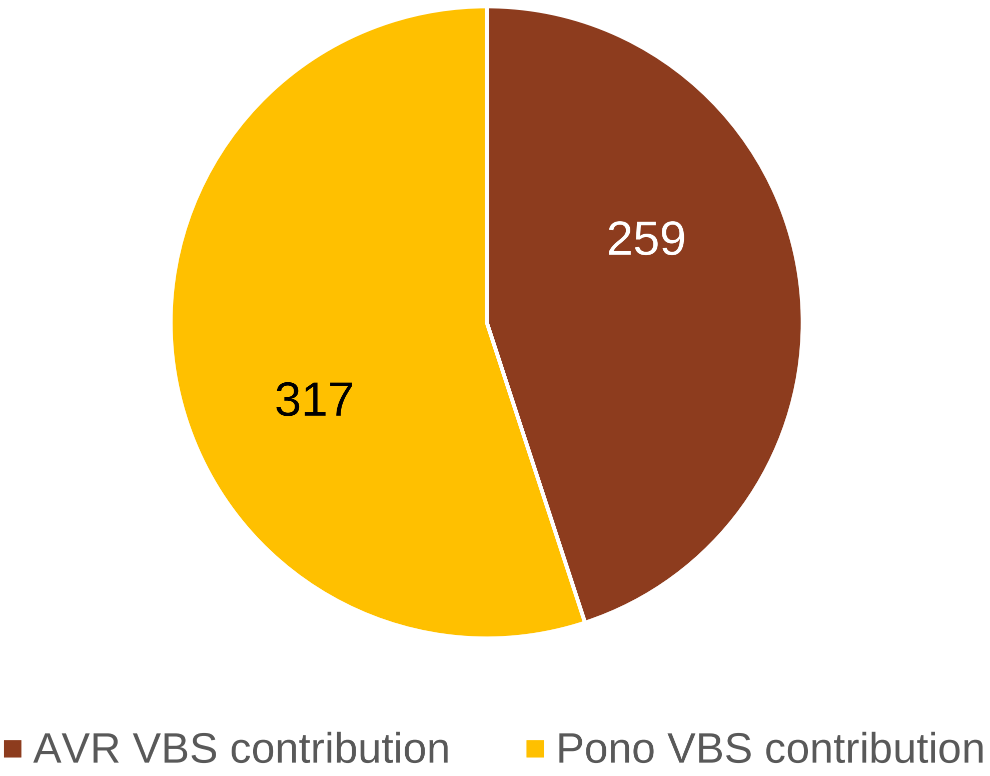
87 VBS SAT Contributions: Pono vs. AVR



489 VBS UNSAT Contributions: Pono vs. AVR



576 VBS Total Contributions: Pono vs. AVR



Summary: Pono, an SMT-based Model Checker

- Word-level model checking by state-of-the-art SMT solving.
- Performance diversity: algorithm portfolios, VBS analysis.
- Open-source vs. commercial tools:
  - Criteria: usability, robustness, performance.

- Pono on GitHub: *https://github.com/upscale-project/pono*
- CAV'21: *Pono: A Flexible and Extensible SMT-Based Model Checker*
- SAT'21: *Smt-Switch: A Solver-Agnostic C++ API for SMT Solving*