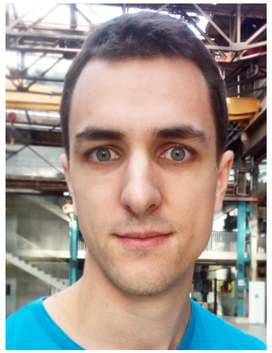
Generative Type-Aware Operator Mutation for Testing SMT Solvers

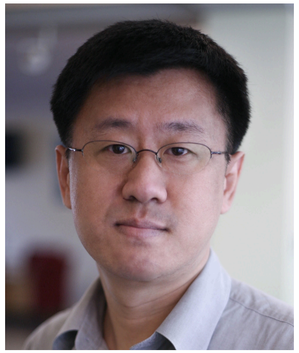
Jiwon Park*
École Polytechnique, France

Dominik Winterer*
ETH Zurich, Switzerland



Chengyu Zhang
East China Normal University, China

Zhendong Su
ETH Zurich, Switzerland



OOPSLA, 20 Oct, 2021, Online

SMT Problem

$$\varphi : x > 0 \wedge x < 0$$

SMT Problem

$$\varphi : x > 0 \wedge x < 0$$

UNSAT

SMT Problem

$$\varphi : x > 0 \wedge x < 1$$

SMT Problem

$$\varphi : x > 0 \wedge x < 1$$

SAT

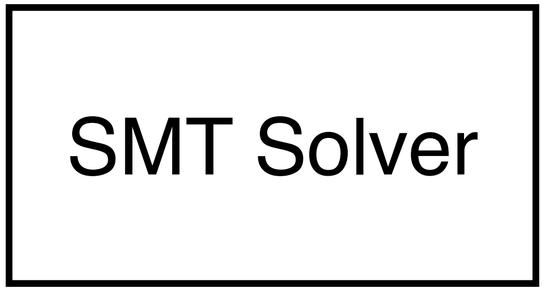
SMT Problem

$$x = 0.5$$

$$\varphi : x > 0 \wedge x < 1$$

SAT

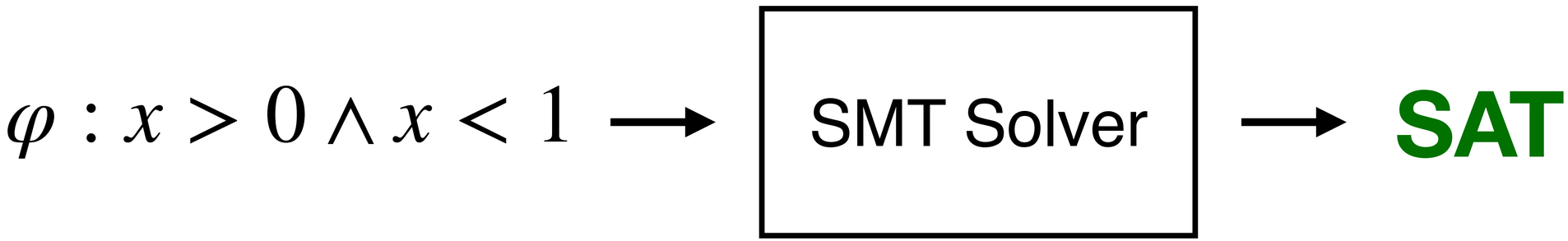
SMT Solver



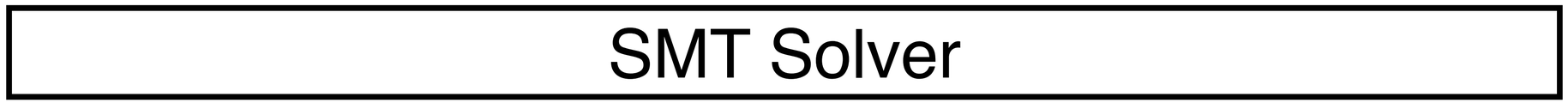
SMT Solver

$\varphi : x > 0 \wedge x < 1 \longrightarrow$ SMT Solver

SMT Solver

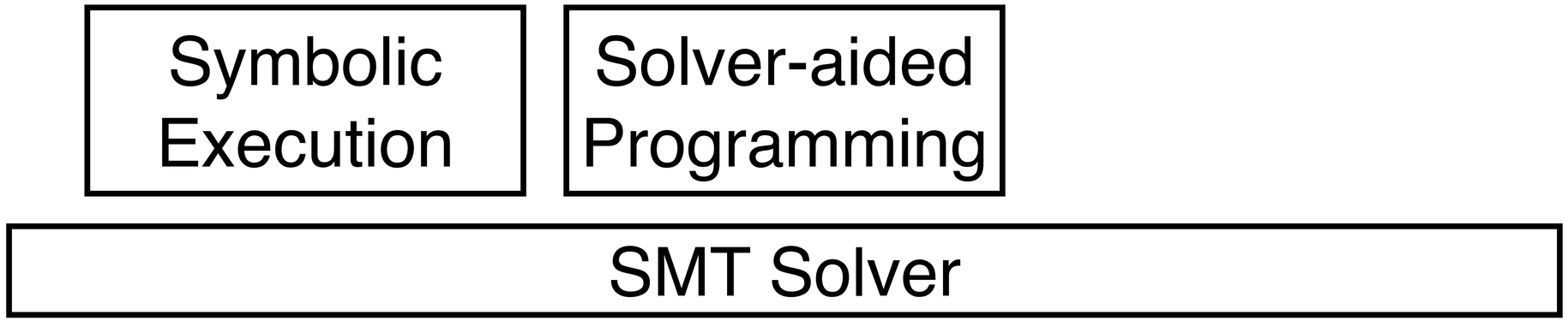


SMT Solver

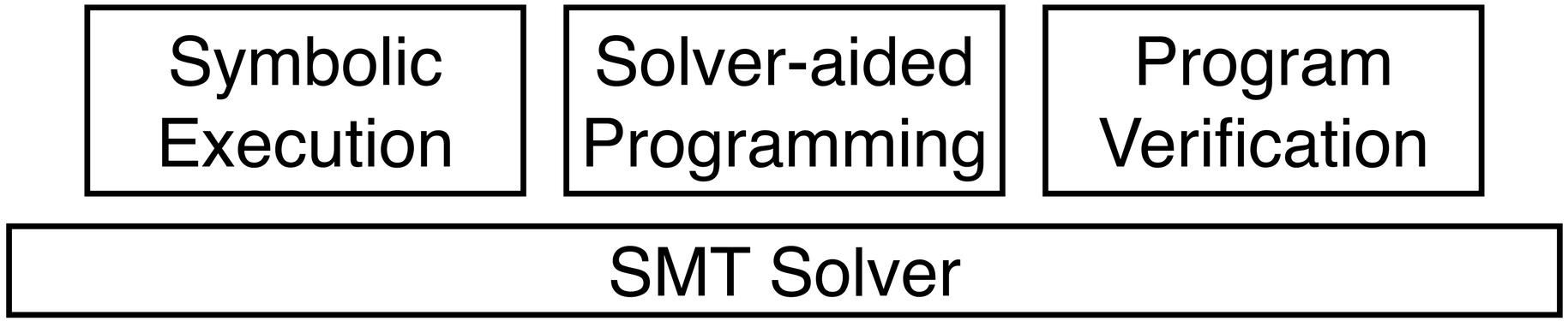


SMT Solver

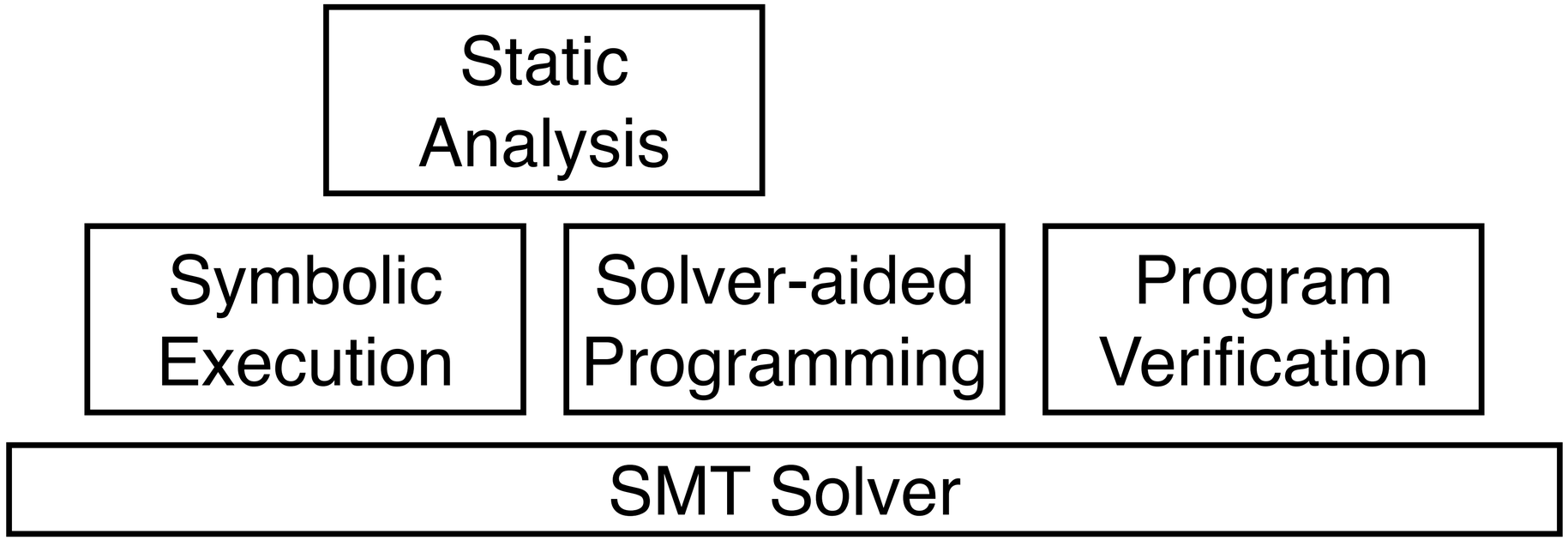
SMT Solver



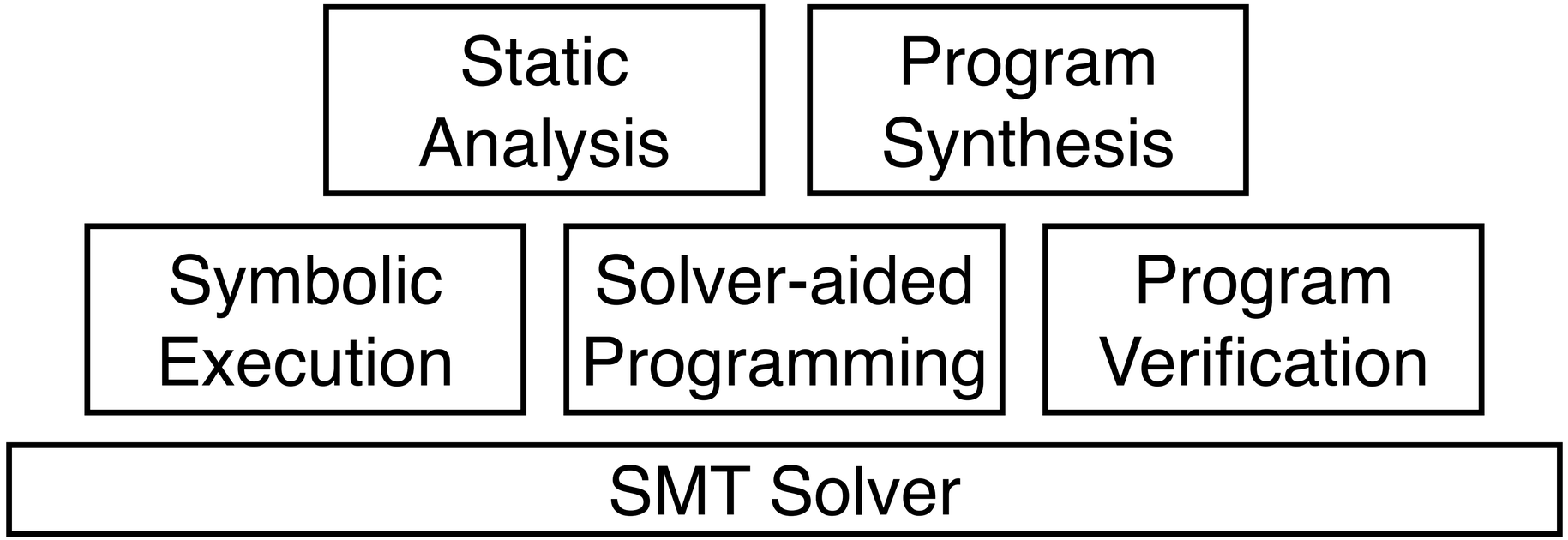
SMT Solver


Symbolic Execution
Solver-aided Programming
Program Verification
SMT Solver

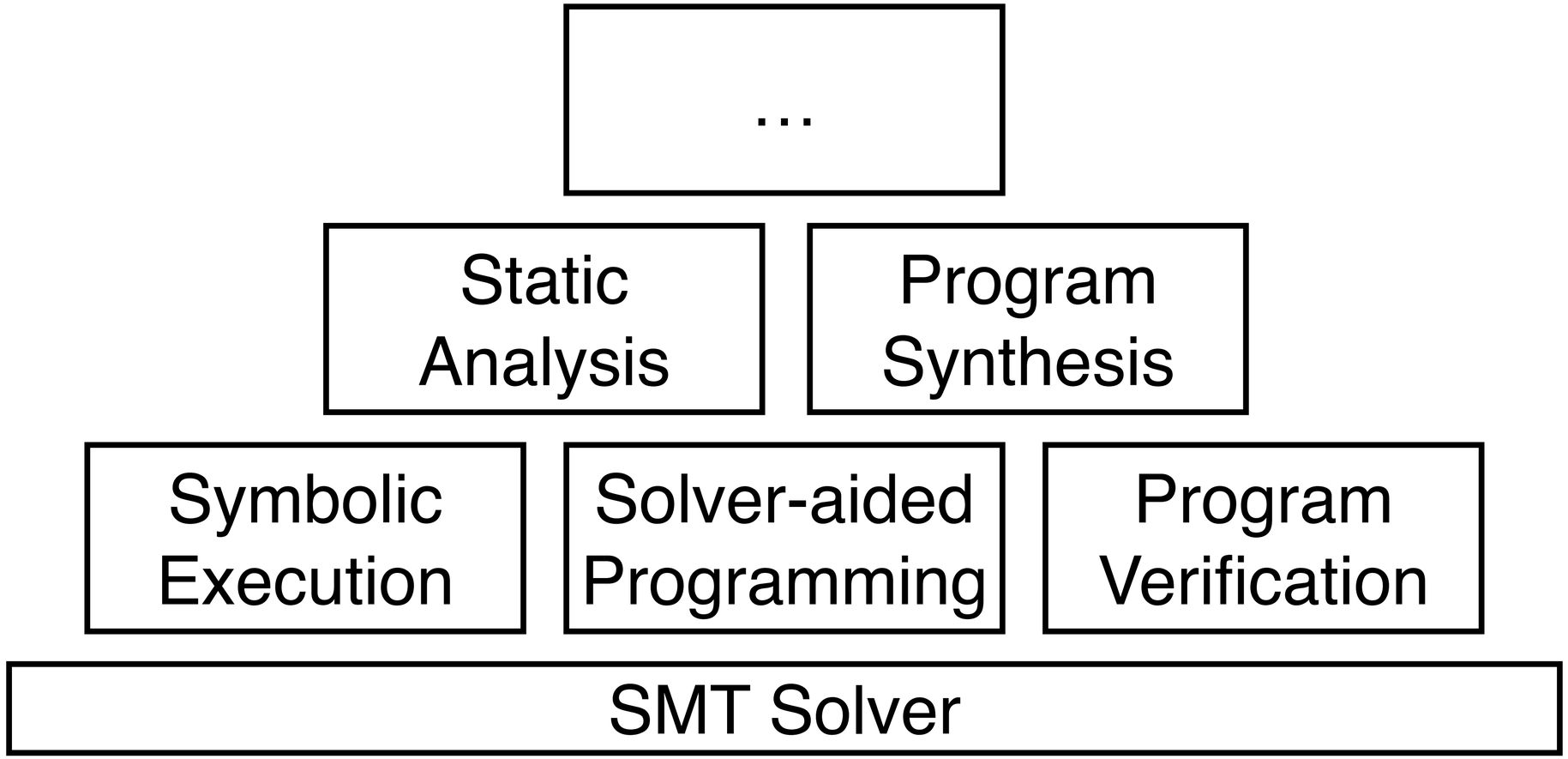
SMT Solver


Static
Analysis
Symbolic
Execution
Solver-aided
Programming
Program
Verification
SMT Solver

SMT Solver

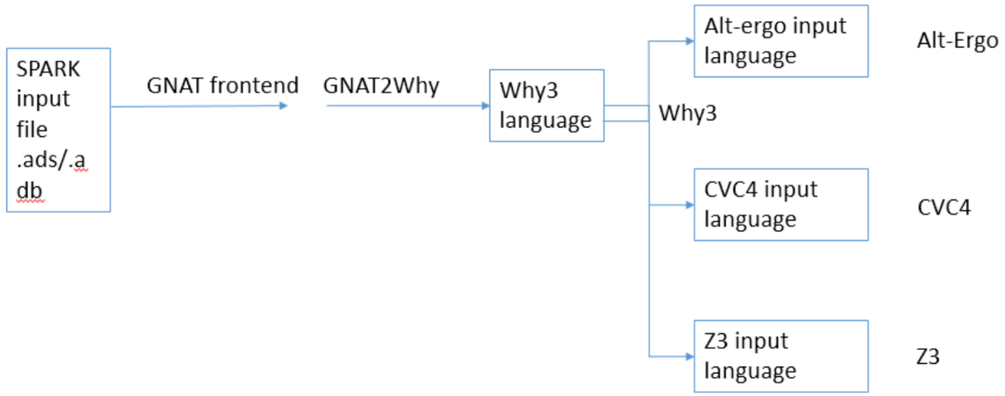

Static Analysis
Program Synthesis
Symbolic Execution
Solver-aided Programming
Program Verification
SMT Solver

SMT Solver


...
Static Analysis
Program Synthesis
Symbolic Execution
Solver-aided Programming
Program Verification
SMT Solver


```
{
    "Resource": “bucket1”
    "Prohibited": { "AWS": “3333"},
}
```

$(Resource = \text{“bucket1”}) \wedge (Prohibited \neq 3333)$



SMT-LIB language

$$\varphi : x > 0 \wedge x < 1$$

SMT-LIB language

$$\varphi : x > 0 \wedge x < 1 \longrightarrow$$

```
(declare-fun x () Real)
(assert (and (> x 0)(< x 1)))
(check-sat)
```

SMT-LIB language

$$\varphi : x > 0 \wedge x < 1 \longrightarrow$$

```
(declare-fun x () Real)
(assert (and (> x 0)(< x 1)))
(check-sat)
```

SMT-LIB language

$$\varphi : x > 0 \wedge x < 1 \longrightarrow$$

```
(declare-fun x () Real)
(assert (and (> x 0)(< x 1)))
(check-sat)
```

SMT-LIB language

$$\varphi : x > 0 \wedge x < 1 \longrightarrow$$

```
(declare-fun x () Real)
(assert (and (> x 0)(< x 1)))
(check-sat)
```

```
(= a "")          iff      (= (str.len a) 0)
```

```
(declare-fun a () String)
(assert (= a ""))
(check-sat)
(get-model)
```

```
(declare-fun a () String)
(assert (= 0 (str.len a)))
(check-sat)
(get-model)
```

```
$ cat formula.smt2
(declare-fun a () String)
(assert (= a ""))
(check-sat)
(get-model)

$ z3 model_validate=true formula.smt2
sat
```

```
$ cat formula.smt2
(declare-fun a () String)
(assert (= 0 (str.len a)))
(check-sat)
(get-model)

$ z3 model_validate=true formula.smt2
sat
(error an invalid model was generated)
```

Z3 Invalid Model Bug #5140

QF_S

```
$ cat formula.smt2
(declare-fun a () String)
(assert (= a ""))
(check-sat)
(get-model)

$ z3 model_validate=true formula.smt2
sat ✔
```

```
$ cat formula.smt2
(declare-fun a () String)
(assert (= 0 (str.len a)))
(check-sat)
(get-model)

$ z3 model_validate=true formula.smt2
sat ✘
(error an invalid model was generated)
```

https://github.com/Z3Prover/z3/issues/5140#issuecomment-812306248

Type-Aware Operator Mutation[1]

```
(declare-fun a () Int)
(declare-fun b () Int)
(assert (> a (+ b 5))
(check-sat)

(declare-fun a () Int)
(declare-fun b () Int)
(assert (<= a (+ b 5))
(check-sat)
```

FuzzChick's Mutators[2]

```
(declare-fun a () Int)
(Declare-fun b () Int)
(assert (> a (+ b 5))
(check-sat)

(declare-fun a () Int)
(Declare-fun b () Int)
(assert (< 5 (+ b 5))
(check-sat)
```

[1] Dominik Winterer, Chengyu Zhang, and Zhendong Su. 2020a. On the Unusal Effectiveness of Type-Aware Operator Mutation. OOPSLA '20.
[2] Leonidas Lampropoulos, Michael Hicks, and Benjamin C. Pierce. 2019. Coverage guided, property based testing. OOPSLA '19.

Type-Aware Operator Mutation [OOPSLA ’21] + **FuzzChick’s Mutators[2]**

Generative Type-Aware Mutation

[1] Dominik Winterer, Chengyu Zhang, and Zhendong Su. 2020a. On the Unusal Effectiveness of Type-Aware Operator Mutation. OOPSLA ’20.
[2] Leonidas Lampropoulos, Michael Hicks, and Benjamin C. Pierce. 2019. Coverage guided, property based testing. OOPSLA ’19.

Generative Type-Aware Mutation

```
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ x x))) 0))
(check-sat)
```

Generative Type-Aware Mutation

```
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ x x))) 0))
(check-sat)
```

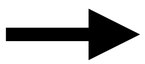
Bool: (> (- (str.to_int (str.++ x x))) 0))

Int : 0, (- (str.to_int (str.++ x x))) 0), …

String : (str.++ x x), x

Generative Type-Aware Mutation

```
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ x x))) 0))
(check-sat)
```

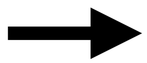


```
(declare-fun x () String)
(assert (> (- (str.to_int
          (str.++  x))) 0))
(check-sat)
```

Generative Type-Aware Mutation

```
(declare-fun x () String)
(assert (> (- (str.to_int
        (str.++ x x))) 0))
(check-sat)
```

String

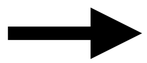


```
(declare-fun x () String)
(assert (> (- (str.to_int
        (str.++  x))) 0))
(check-sat)
```

Generative Type-Aware Mutation

```
(declare-fun x () String)
(assert (> (- (str.to_int
        (str.++ x x))) 0))
(check-sat)
```

String



```
(declare-fun x () String)
(assert (> (- (str.to_int
        (str.++  x))) 0))
(check-sat)
```

```
Bool: (> (- (str.to_int (str.++ x x))) 0))

Int : 0, (- (str.to_int (str.++ x x))) 0), …

String : (str.++ x x), x
```

Generative Type-Aware Mutation

```
(declare-fun x () String)
(assert (> (- (str.to_int
        (str.++ x x))) 0))
(check-sat)
```

String

```
(declare-fun x () String)
(assert (> (- (str.to_int
        (str.++  x))) 0))
(check-sat)
```

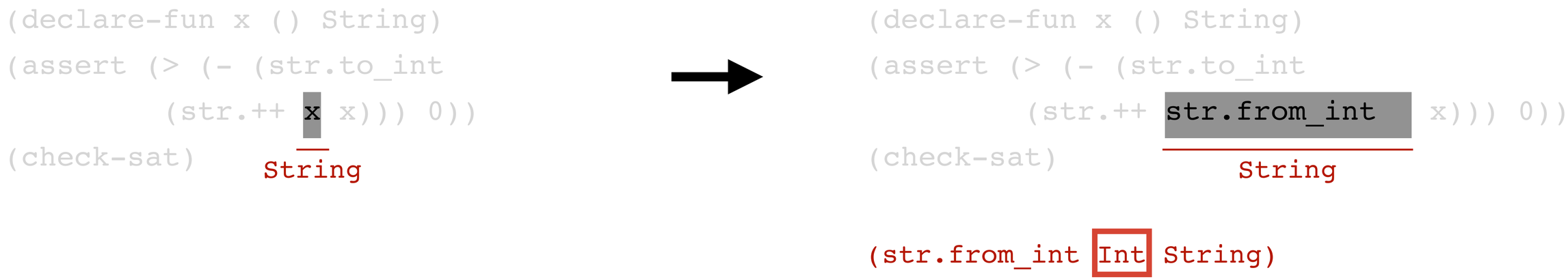
Operator candidates: str.from_int
str.++
str.replace
...

```
Bool: (> (- (str.to_int (str.++ x x))) 0))

Int : 0, (- (str.to_int (str.++ x x))) 0), …

String : (str.++ x x), x
```

Generative Type-Aware Mutation



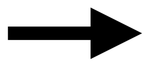

```
Bool: (> (- (str.to_int (str.++ x x))) 0))

Int : 0, (- (str.to_int (str.++ x x))) 0), …

String : (str.++ x x), x
```

Generative Type-Aware Mutation

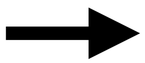
```
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ x x))) 0))
(check-sat)
```



```
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ (str.from_int 0) x))) 0))
(check-sat)
```

Generative Type-Aware Mutation

```
$ cat formula.smt2
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ x x))) 0))
(check-sat)
$ cvc4 formula.smt2
sat
$ z3 formula.smt2
sat
```

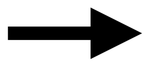


```
$ cat mutant.smt2
(declare-fun x () String)
(assert (> (- (str.to_int
          (str.++ (str.from_int 0) x))) 0))
(check-sat)
$ cvc4 formula.smt2
sat    ✓
$ z3 formula.smt2
unsat  ✗
```

Generative Type-Aware Mutation

QF_SLIA

```
$ cat formula.smt2
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ x x))) 0))
(check-sat)
$ cvc4 formula.smt2
sat
$ z3 formula.smt2
sat
```



```
$ cat mutant.smt2
(declare-fun x () String)
(assert (> (- (str.to_int
         (str.++ (str.from_int 0) x))) 0))
(check-sat)
$ cvc4 formula.smt2
sat    ✓
$ z3 formula.smt2
unsat  ✗
```

https://github.com/Z3Prover/z3/issues/5108

Empirical Evaluation

Empirical Evaluation

- Tool: TypeFuzz (based on yinyang)

Empirical Evaluation

- Tool: TypeFuzz (based on yinyang)
- Bug hunting: Feb 2021 - Sep 2021

Empirical Evaluation

- Tool: TypeFuzz (based on yinyang)
- Bug hunting: Feb 2021 - Sep 2021
- Testing targets: 

Empirical Evaluation

- Tool: TypeFuzz (based on yinyang)
- Bug hunting: Feb 2021 - Sep 2021
- Testing targets: 

- Seeds: SMT-LIB benchmarks (Strings, Arithmetic, BV)

Bug Findings

Status	Z3	CVC4	Total
Reported	177	60	237
Confirmed	135	54	189
Fixed	132	44	176
Duplicate	9	5	14

Bug status as of 30 Sep 2021

Bug Findings

Status	Z3	CVC4	Total
Reported	177	60	237
Confirmed	135	54	189
Fixed	132	44	176
Duplicate	9	5	14

Bug status as of 30 Sep 2021

Bug Findings

Status	Z3	CVC4	Total
Reported	177	60	237
Confirmed	135	54	189
Fixed	132	44	176
Duplicate	9	5	14

Bug status as of 30 Sep 2021

Bug Findings

Status	Z3	CVC4	Total
Soundness	49	24	73
Crash	47	20	67
Invalid model	39	10	49

Types of confirmed bugs

Bug Findings

Status	Z3	CVC4	Total
Soundness	49	24	73
Crash	47	20	67
Invalid model	39	10	49

Types of confirmed bugs

Bug Findings

Status	Z3	CVC4	Total
Soundness	49	24	73
Crash	47	20	67
Invalid model	39	10	49

Types of confirmed bugs

Bug Findings

Status	Z3	CVC4	Total
Soundness	49	24	73
Crash	47	20	67
Invalid model	39	10	49

Types of confirmed bugs

Bug Findings

#Options	Z3	CVC4	Total
Default	55		
1	12		
2	64	10	74
>= 3	5	0	5

63 out of 64 bugs on “new core” of Z3
(tactic.default_tactic=smt sat.euf=true)

Bug Findings

NikolajBjorner added a commit that referenced this issue on May 31

#5223 ✓ fb75dac

NikolajBjorner added a commit that referenced this issue on May 31

#5223 fe0727d

NikolajBjorner added a commit that referenced this issue on May 31

#5223 8d1dfb9

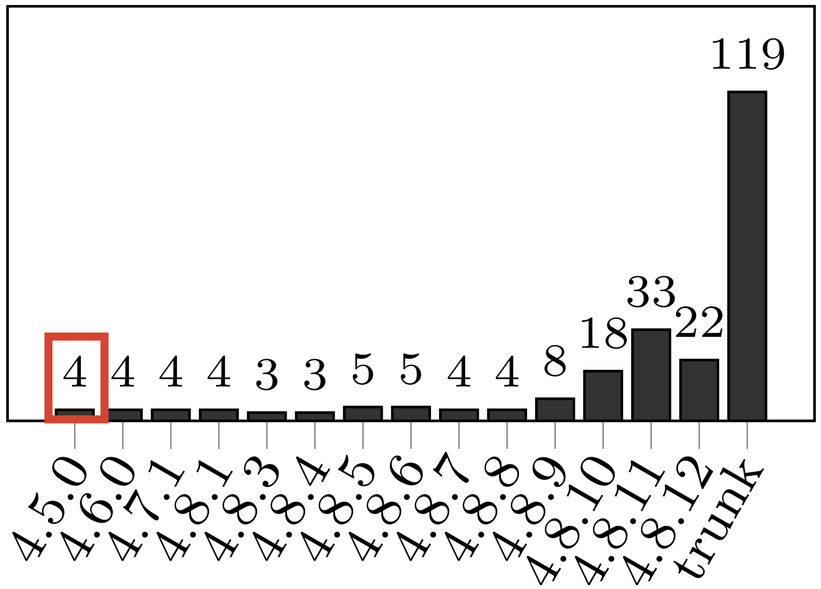
NikolajBjorner commented on May 31 Contributor

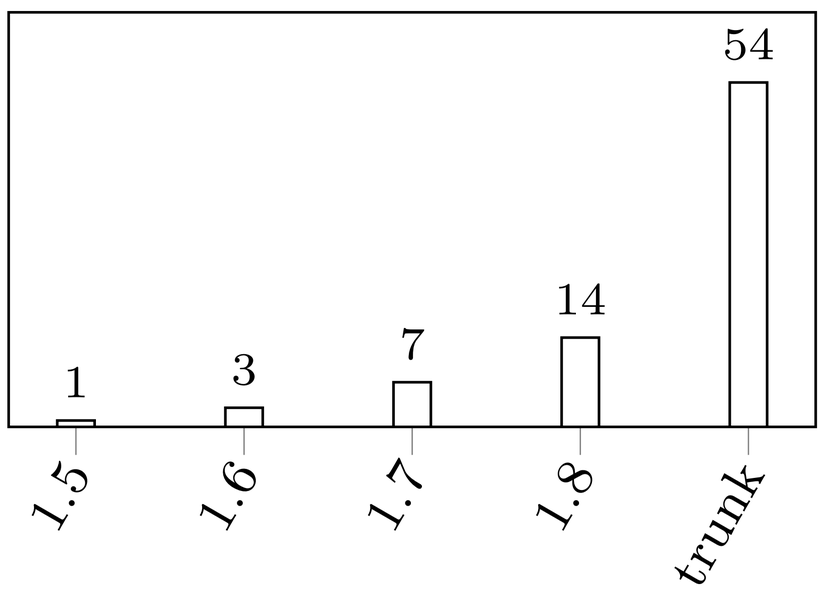
Thanks for targeting the new code.
It is a very good use of the fuzzing facilities and helps reaching
a more solid state for this so-far not exercised code.
All bugs reported in this thread have now been fixed.

👍 1

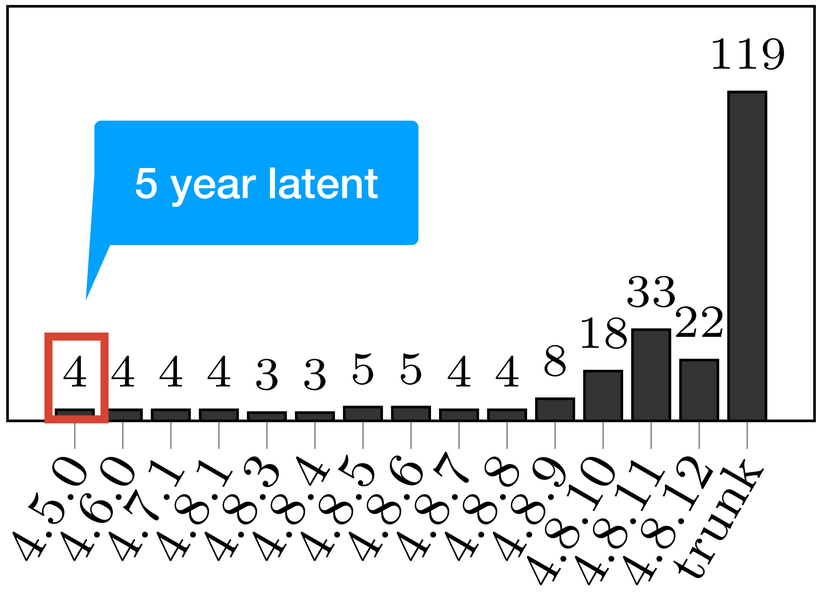
https://github.com/Z3Prover/z3/issues/5223

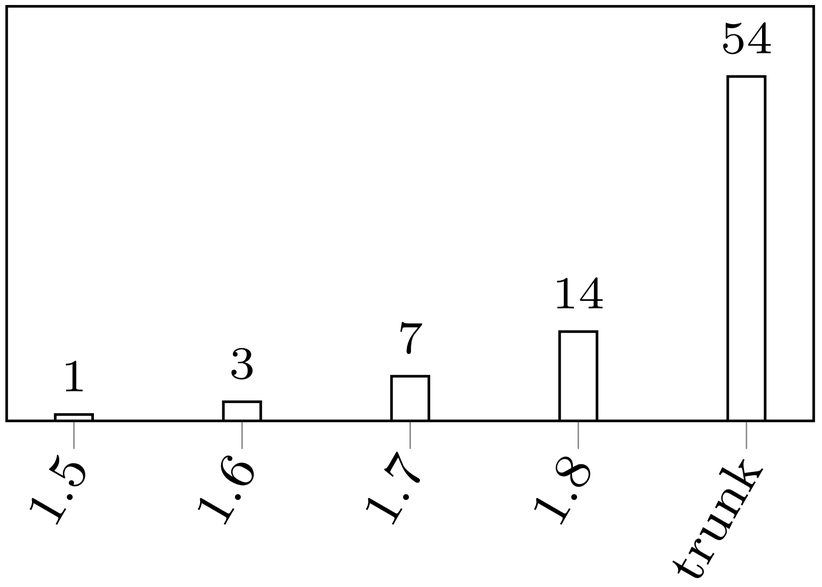
Bugs Affecting Historic Releases



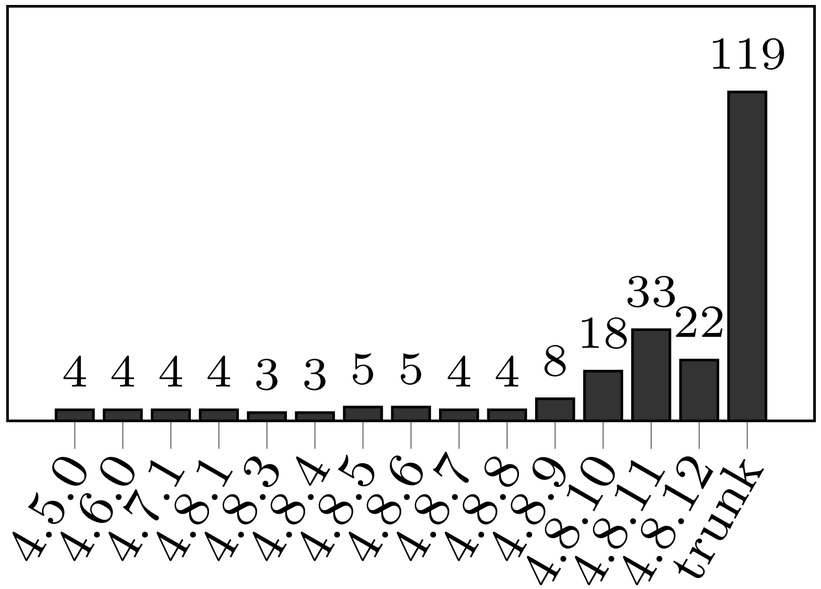


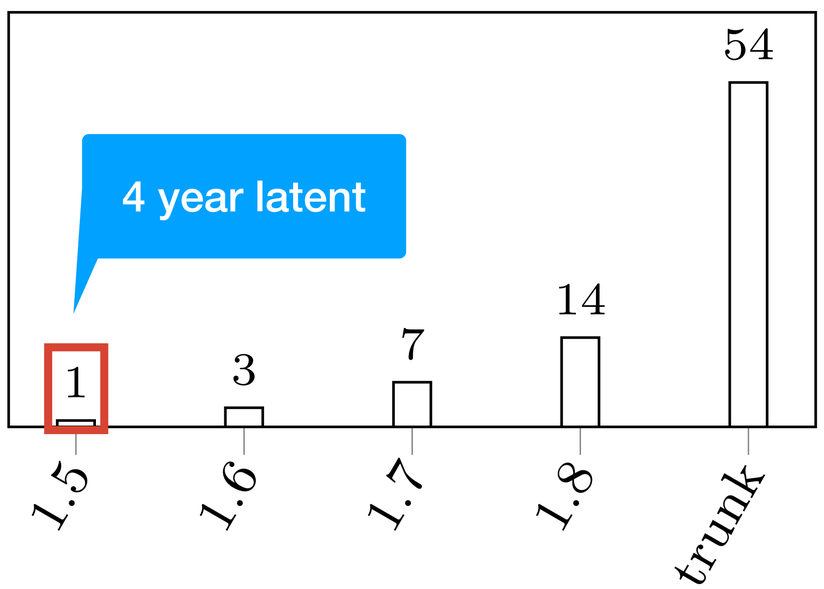
Bugs Affecting Historic Releases



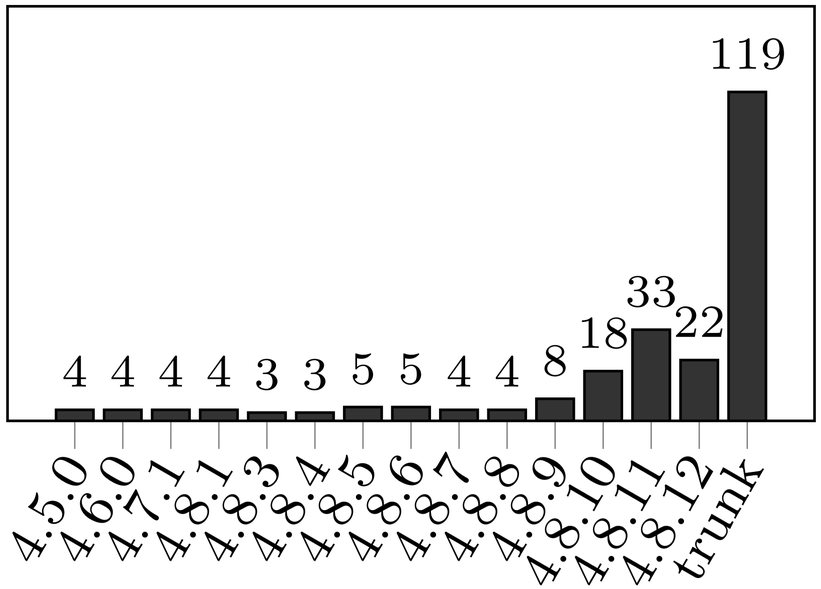


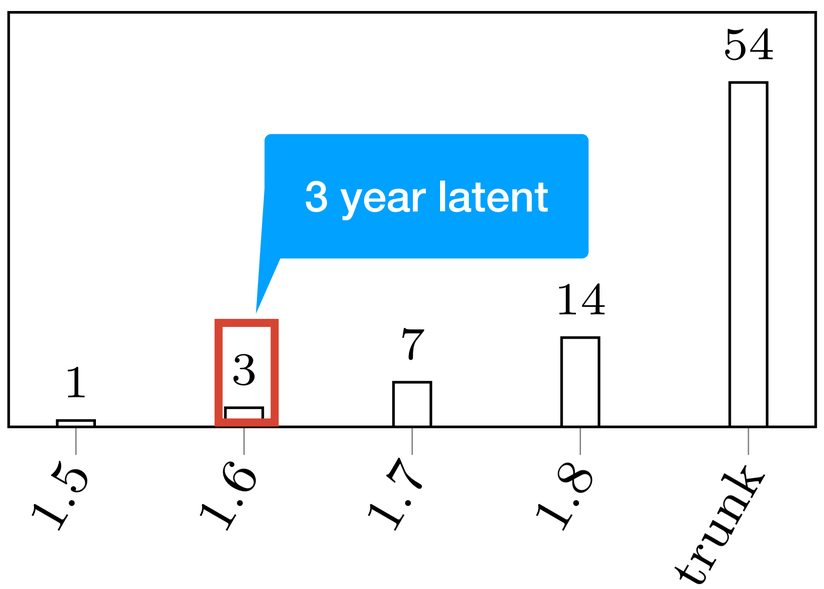
Bugs Affecting Historic Releases





Bugs Affecting Historic Releases





CVC5 Soundness Bug #6075

QF_S

```
$ cat bug.smt2
(declare-fun x () String)
(declare-fun y () String)
(assert (str.< x (str.replace "" (str.++ (str.replace "B" x "")
        (str.replace "B" (str.replace "B" x "") "")) y)))
(check-sat)


$ z3 bug.smt2
unsat ✔

$ cvc5 bug.smt2
sat   ✘
```

https://github.com/CVC5/CVC5/issues/6075

CVC5 Soundness Bug #6075

QF_S

```
$ cat bug.smt2
(declare-fun x () String)
(declare-fun y () String)
(assert (str.< x (str.replace "" (str.++ (str.replace "B" x "")
        (str.replace "B" (str.replace "B" x "") "")) y)))
(check-sat)


$ z3 bug.smt2
unsat ✔

$ cvc5 bug.smt2
sat   ✘
```

https://github.com/CVC5/CVC5/issues/6075

Z3 Soundness Bug #5140

QF_S

```
[619] % z3release small.smt2
unsat
[620] % cvc4 -q small.smt2
sat
[621] % cat small.smt2
(assert (str.contains (str.++ "\uAA" "AA") "AAA"))
(check-sat)
[622] %
```

https://github.com/Z3Prover/z3/issues/5140#issuecomment-846386635

Z3 Soundness Bug #5429

QF_BV

```
$ cat bug.smt2
(declare-fun a () (_ BitVec 64))
(assert (not (fp.geq ((_ to_fp 11 53) a) ((_ to_fp 11 53)
(_ bv0 64)))))
(check-sat)

$ z3release bug.smt2
sat  ✔

$ z3release tactic.default_tactic=smt sat.euf=true bug.smt2
unsat ✘
```

https://github.com/Z3Prover/z3/issues/5429#issuecomment-885793775

CVC4 Soundness Bug #6142

QF_S

```
$ cat bug.smt2
(declare-fun x () String)
(declare-fun y () String)
(assert (= (str.++ "A" y) (str.replace x (str.++ "A" y)
(str.replace (str.++ "A" (str.replace y "" "A")) x "A"))))
(check-sat)
(get-model)

$ z3 bug.smt2
sat   ✔

$ cvc4-1.5 bug.smt2
unsat ✘

$ cvc4-1.6 bug.smt2
unsat ✘
```

https://github.com/cvc5/cvc5/issues/6142

Z3 Soundness Bug #5429

QF_BV

```
$ cat bug.smt2
(declare-fun a () (_ BitVec 64))
(assert (not (fp.geq ((_ to_fp 11 53) a) ((_ to_fp 11 53)
(_ bv0 64)))))
(check-sat)

$ z3release bug.smt2
sat  ✔

$ z3release tactic.default_tactic=smt sat.euf=true bug.smt2
unsat ✘
```

Z3's new core

https://github.com/Z3Prover/z3/issues/5429#issuecomment-885793775

Z3 Soundness Bug #5429

LIA

```
$ cat bug.smt2
(assert (forall ((a Real)) (exists ((b Real)) (> b (* b a)))))
(check-sat)
```

```
$ z3release bug.smt2
unsat ✔
```

Z3's new core

```
$ z3release tactic.default_tactic=smt sat.euf=true bug.smt2
sat   ✘
```

https://github.com/Z3Prover/z3/issues/5429#issuecomment-885803038

Future Work

- Extend TypeFuzz to more logics
- Generalize Generative Type-Aware Mutation to other applications
- Integration into CI of the SMT solvers

yinyang —a fuzzing framework for SMT Solvers

testsmt/yinyang

TypeFuzz [OOPSLA ’21]

OpFuzz [OOPSLA ’20]

YinYang [PLDI ’20]

Project Yin-Yang for Testing SMT Solvers

[Summary: **1,560** (total) / **1,061** (fixed)]

[Z3 bugs: **1,147** (total) / **779** (fixed)]
[CVC4 bugs: **413** (total) / **282** (fixed)]

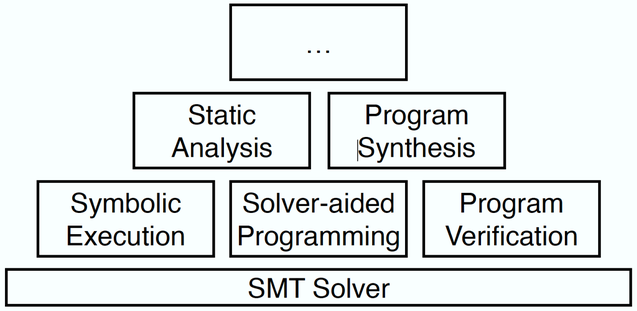
[Bugs in default mode (Z3): **680** (total) / **479** (fixed)]
[Bugs in default mode (CVC4): **204** (total) / **147** (fixed)]

[Soundness bugs (Z3): **375** (total) / **228** (fixed)]
[Soundness bugs (CVC4): **71** (total) / **60** (fixed)]

Summary

testsmt.github.io @testsmt

SMT Solver



Type-aware operator mutation

```
$ cat > formula.smt2
(assert (forall ((a Int))
        (exist  ((b Int))
        (distinct (* 2 b) a))))
(check-sat)

$ cvc4 formula.smt2
sat

$ z3 formula.smt2
sat
```

→

```
$ cat > bug.smt2
(assert (forall ((a Int))
        (exist  ((b Int))
        (= (* 2 b) a))))
(check-sat)

$ cvc4 bug.smt2
unsat ✔

$ z3 bug.smt2
sat   ✘
```

https://github.com/Z3Prover/z3/issues/3973

Bug Findings

Status	Z3	CVC4	Total
Reported	811	281	1,092
Confirmed	578	241	819
Fixed	521	164	106
Duplicate	85	18	79

Bug status as of 30 Oct 2020

Z3 Soundness Bug #2832

```
$ cat bug.smt2
(declare-const a (_ BitVec 8))
(declare-const b (_ BitVec 8))
(declare-const c (_ BitVec 8))
(assert (= (bvxnor a b c)
           (bvxnor (bvxnor a b) c)))
(check-sat)

$ cvc4 bug.smt2
sat    ✔

$ z3 bug.smt2
unsat  ✘
```

https://github.com/Z3Prover/z3/issues/2832

CVC4 Soundness Bug #3497

```
$ cat bug.smt2
(declare-fun x () String)
(declare-fun y () String)
(assert (= (str.indexof x y 1)
           (str.len x)))
(assert (str.contains x y))
(check-sat)

$ z3 bug.smt2
sat    ✔

$ cvc4 bug.smt2
unsat  ✘
```

(str.indexof x y 1) < (str.len x)

Root Cause: Missed boundary condition

https://github.com/CVC4/CVC4/issues/3497

Z3 Soundness Bug #2830

```
$ cat bug.smt2
(declare-fun a () Int)
(declare-fun b (Int) Bool)
(assert (b 0))
(push)
(assert (distinct true (= a 0) (not (b 0))))
(check-sat)

$ cvc4 bug.smt2
unsat  ✔

$ z3 bug.smt2
sat    ✘
```

Root Cause: Last argument omitted in distinct operator

https://github.com/Z3Prover/z3/issues/2830

CVC4 Soundness Bug #4469

```
$ cat bug.smt2
(set-logic QF_AUFBVLIA)
(declare-fun a () Int)
(declare-fun b (Int) Int)
(assert (distinct (b a)
   (b (b a))))
(check-sat)

$ z3 bug.smt2
sat    ✔

$ cvc4 bug.smt2
unsat  ✘
```

Labels: bug, major

https://github.com/CVC4/CVC4/issues/4469

CVC5 Soundness Bug #6834

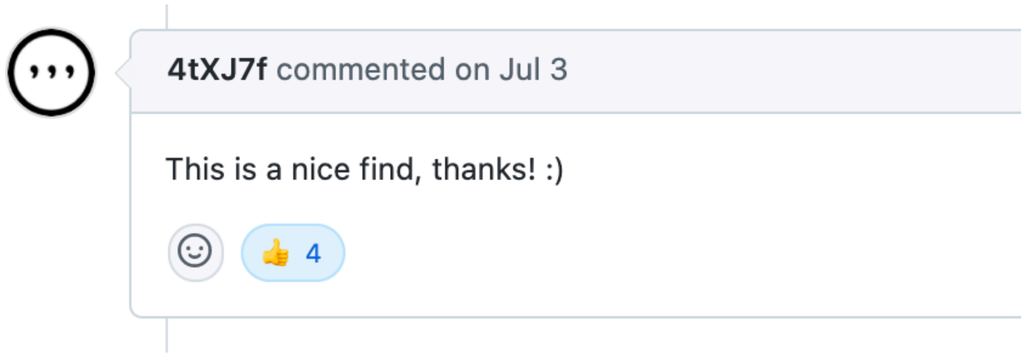
QF_SLIA

major

```
$ cat bug.smt2
(declare-fun a () Int)
(assert (= (str.++ (str.substr "A" 0 a) “B"
(str.substr "A" 0 a)) "B"))
(check-sat)

$ z3 bug.smt2
sat    ✔

$ cvc5 bug.smt2
unsat  ✘
```



https://github.com/CVC5/CVC5/issues/6834