Diver: Oracle-Guided SMT Solver Testing with Unrestricted Random Mutations

Jongwook Kim Sunbeom So Hakjoo Oh

Korea University

ICSE'23 @ Melbourne, Australia

May 19, 2023

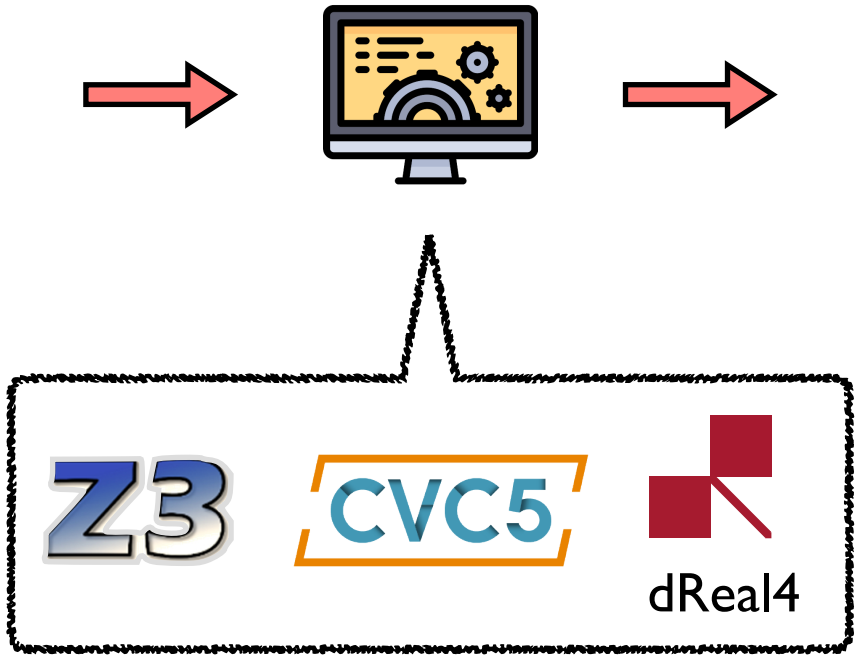
SMT Solver

- Software for deciding the satisfiability of FOL formula.

Formula

$(0 \leq y) \wedge (x = 2 + y)$

SMT Solver



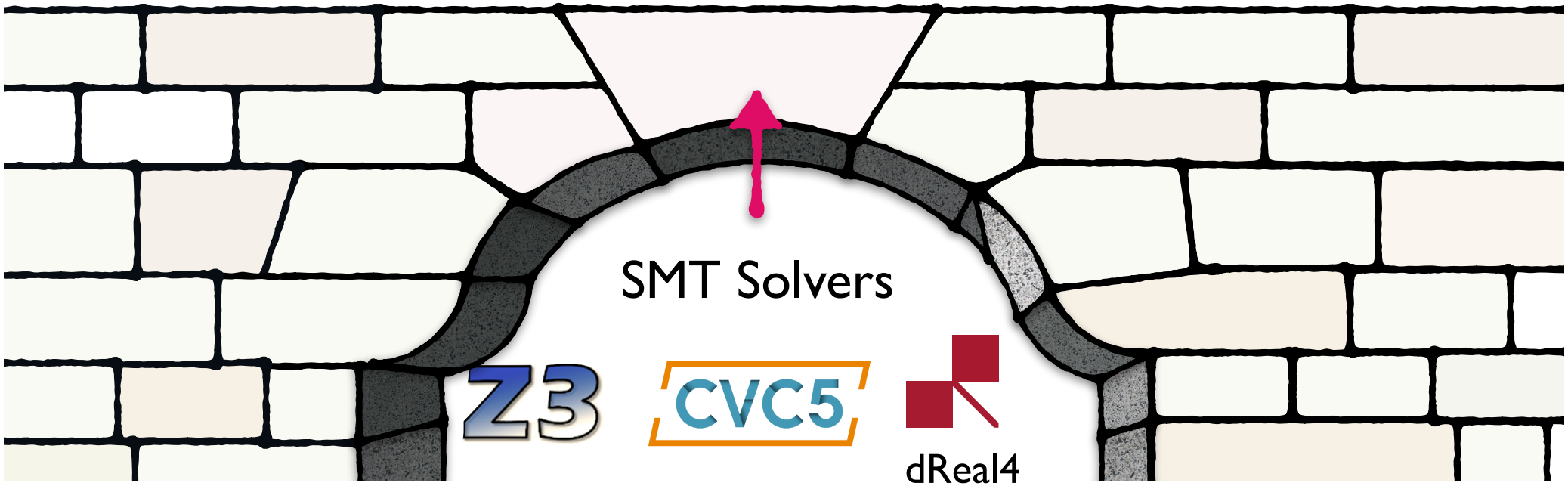
Result

SAT

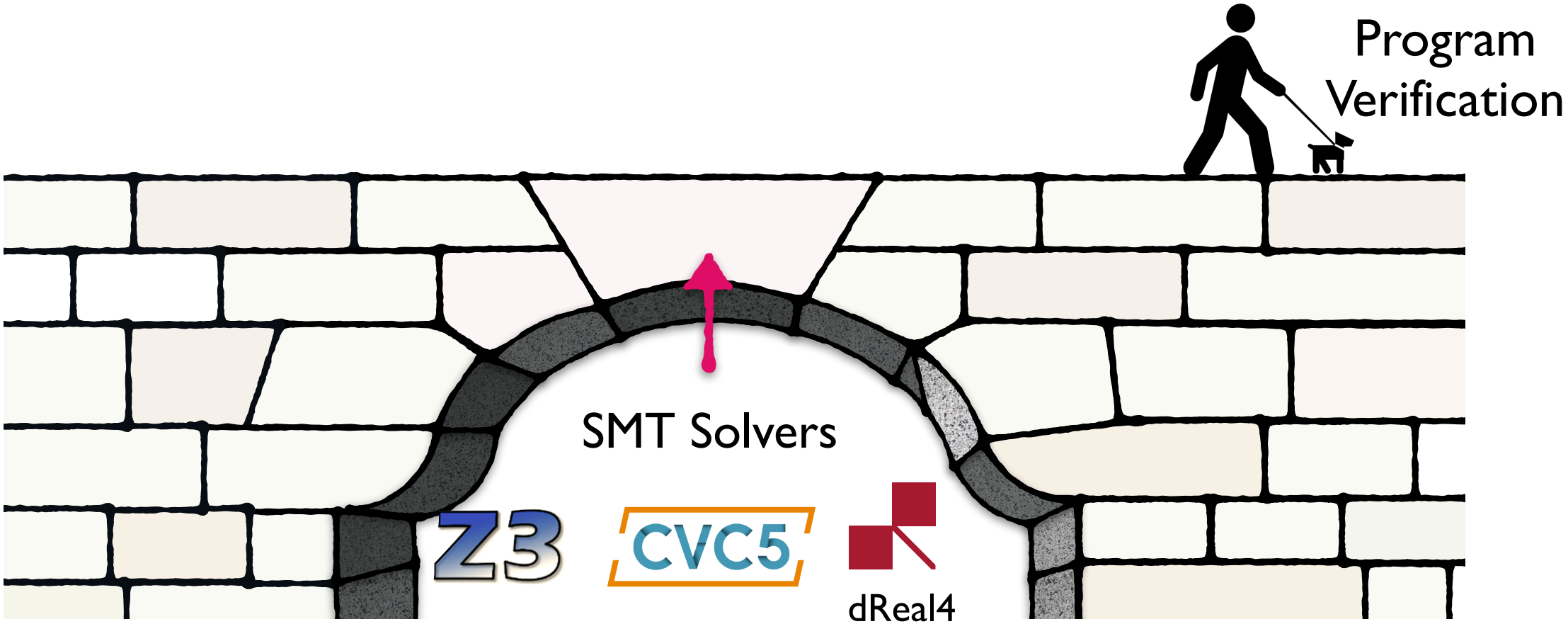
$\{x \mapsto 3,\ y \mapsto 1\}$

Model

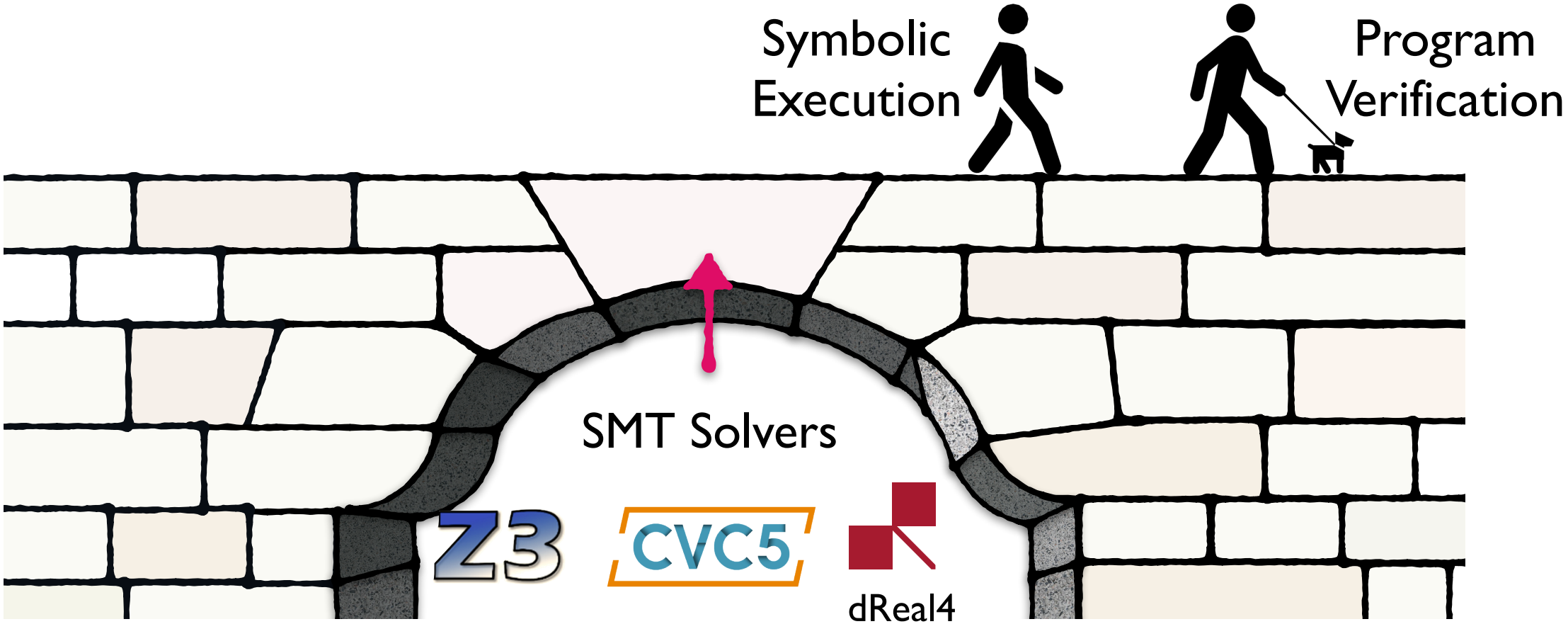
SMT Solver: Keystone of Many SE Tools



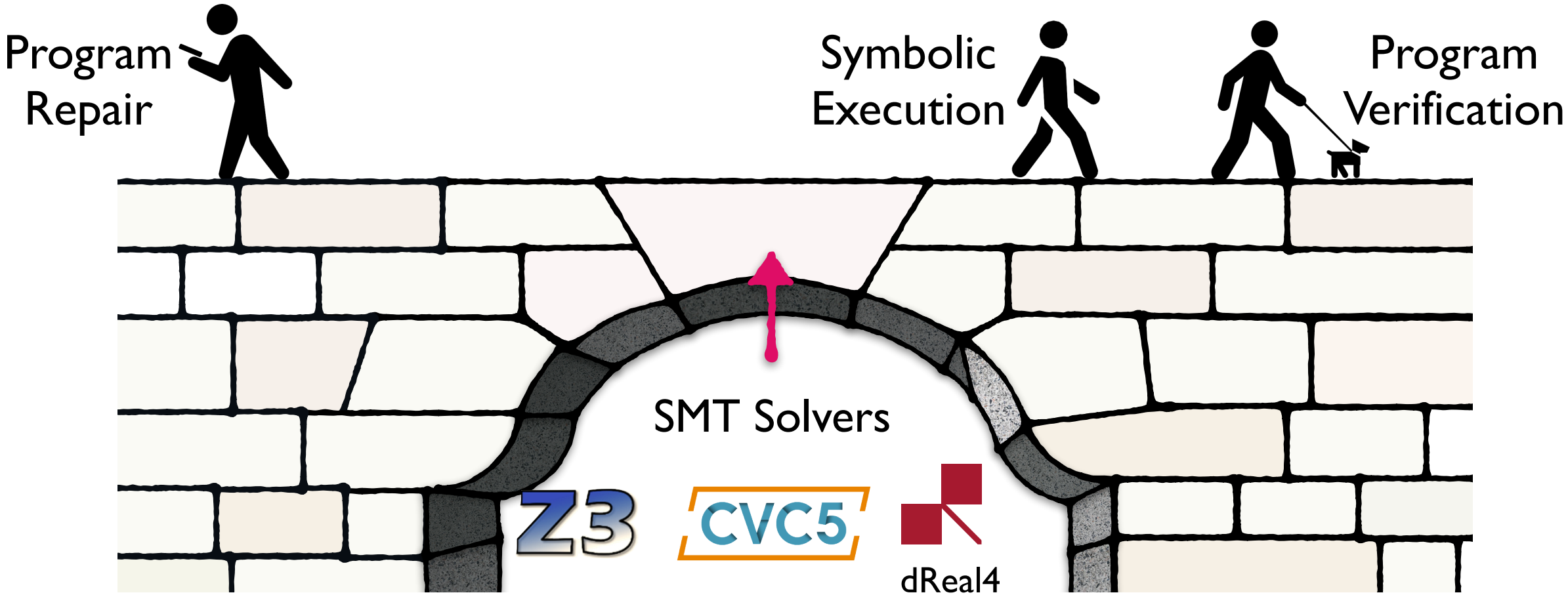
SMT Solver: Keystone of Many SE Tools



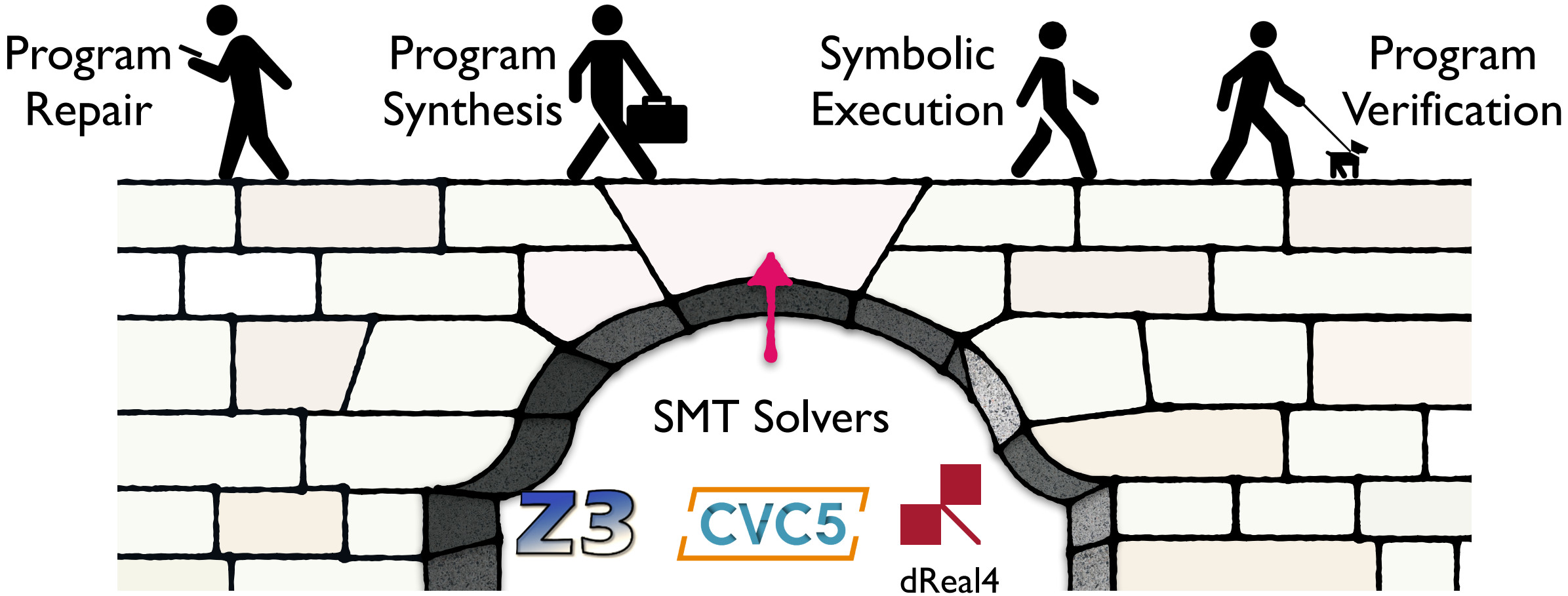
SMT Solver: Keystone of Many SE Tools



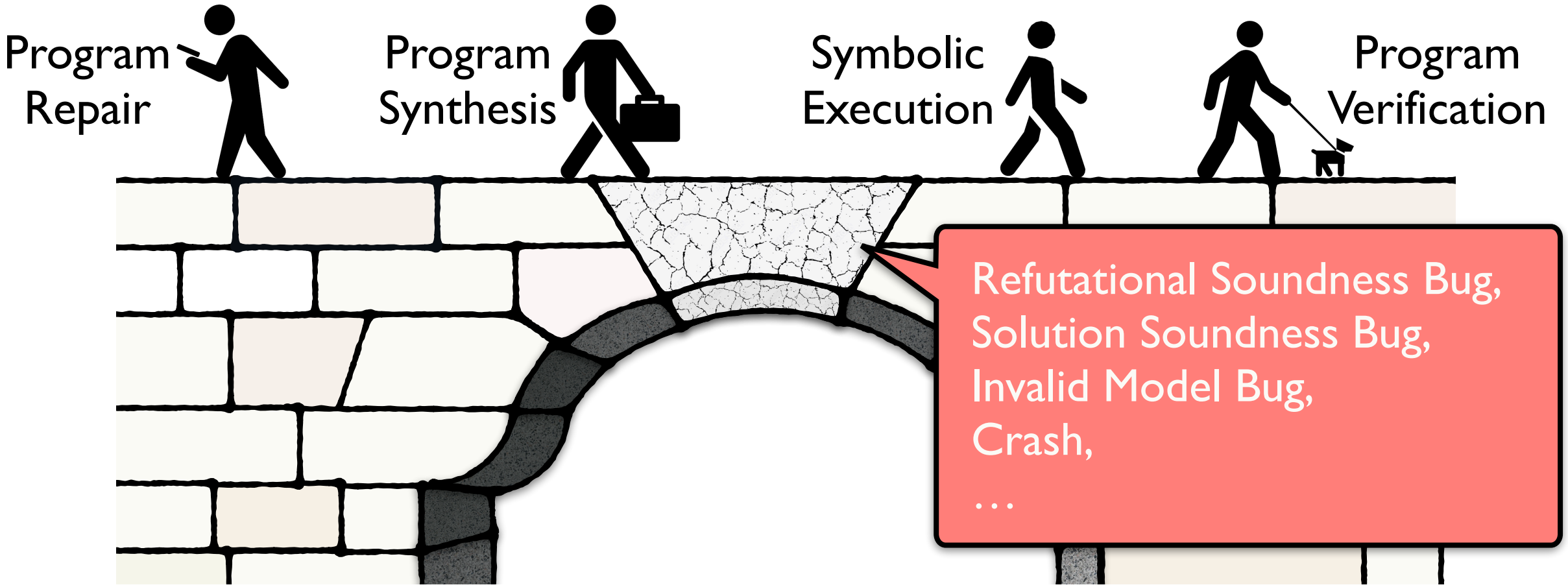
SMT Solver: Keystone of Many SE Tools



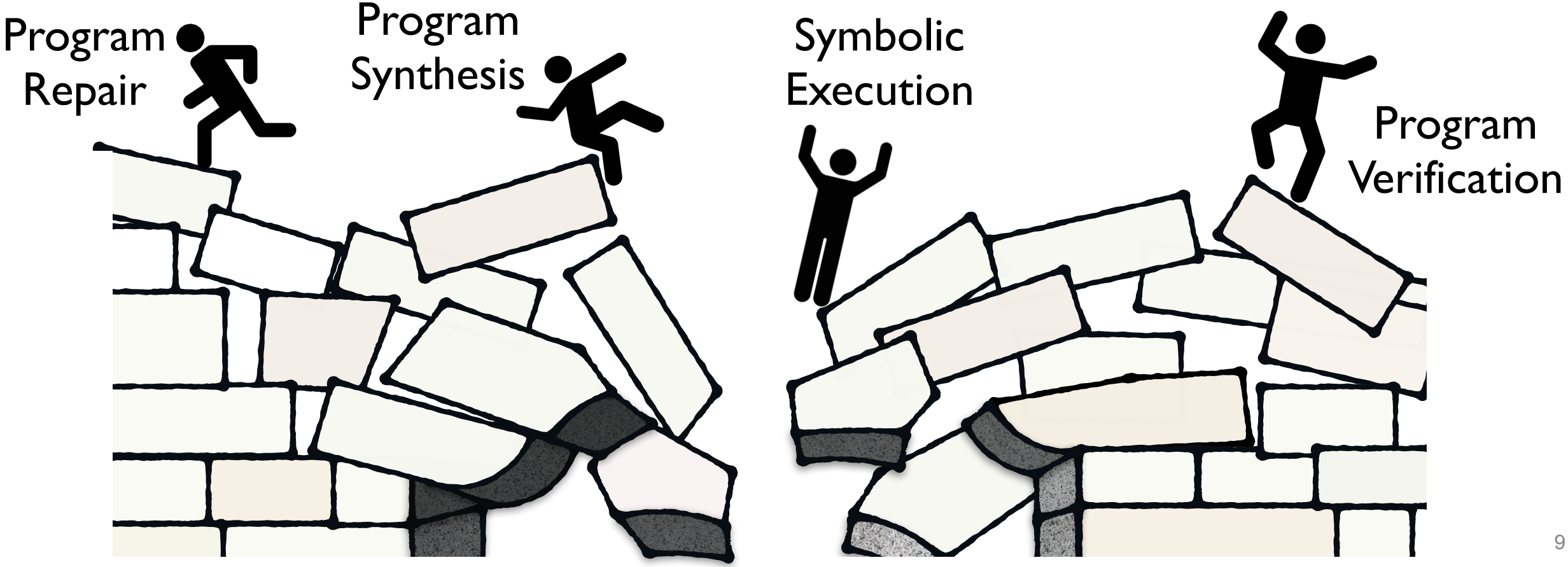
SMT Solver: Keystone of Many SE Tools



SMT Solver: Keystone of Many SE Tools



Correctness of SMT Solver is critical



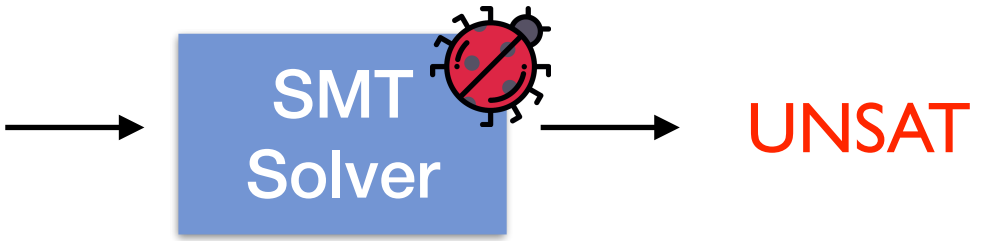
Goal: Find Bugs in SMT Solvers

- Main target bugs

  - Refutational Soundness Bug: UNSAT instead of SAT.

Status: SAT

$(x \geq 10) \wedge (y \geq 5)$

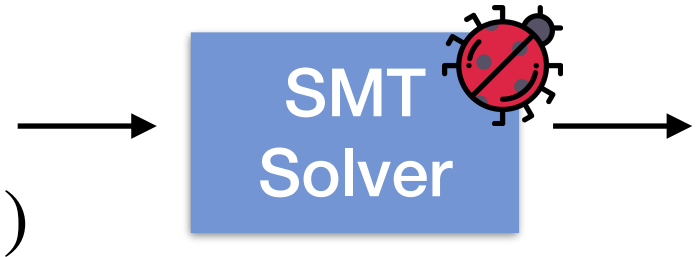


UNSAT

  - Invalid Model Bug: Produce an unsatisfying model.

Status: SAT

$F : (x \geq 10) \wedge (y \geq 5)$



SAT with Incorrect Model

$\{x \mapsto 10,\ y \mapsto 3\} \not\models F$

Goal: Find Bugs in SMT Solvers

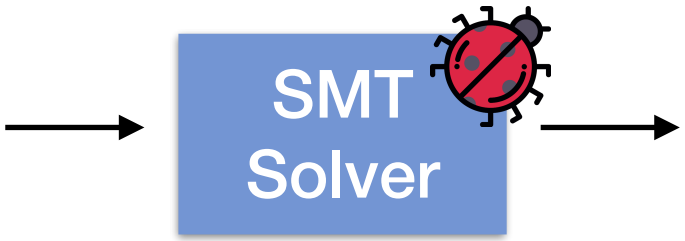
- Main target bugs

  - Refutational Soundness Bug: UNSAT instead of SAT.

Wrong verification results!

Status: SAT

$(x \geq 10) \wedge (y \geq 5)$



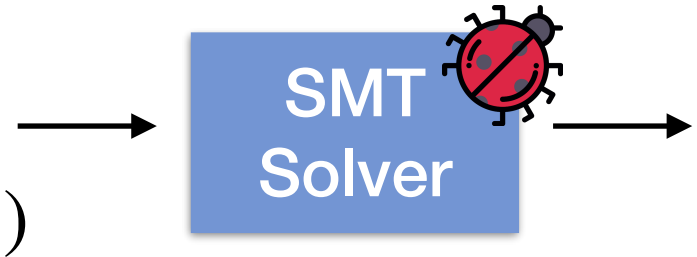
UNSAT

  - Invalid Model Bug: Produce an unsatisfying model.

Unintended Test-cases!

Status: SAT

$F : (x \geq 10) \wedge (y \geq 5)$



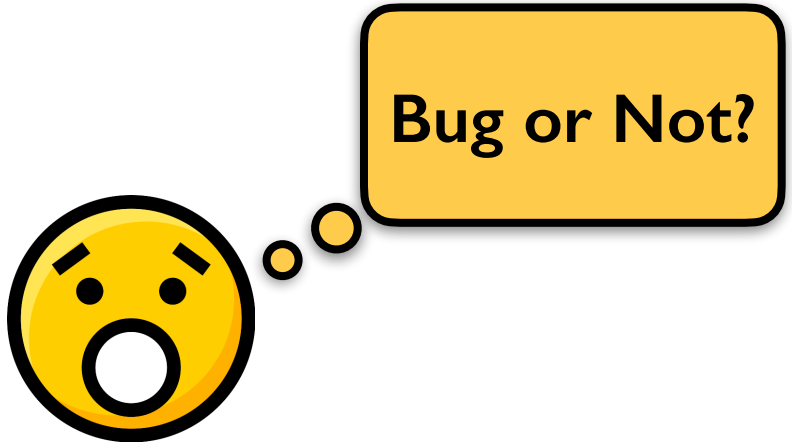
SAT with Incorrect Model

$\{x \mapsto 10, y \mapsto 3\} \not\models F$

Challenge: Test Oracle Problem

- These bugs can be identified when generated formula is known to be satisfiable.

```
$ cat test.smt2
(set-logic QF_SLIA)
(declare-fun t () String)
(assert (str.prefixof "-" (str.substr t 0 1)))
(assert (> (str.len (str.substr t 0 2)) 1))
(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
(assert (str.suffixof (str.replace t "-0" "-") "-"))
(check-sat)
```



Existing Approaches

- Differential Testing: Cross-check using multiple solvers.
  - TypeFuzz [OOPSLA'21], OpFuzz [OOPSLA'20]

- Oracle-Guided: Rely on "predefined", restricted mutation rules.
  - Sparrow [FSE'21], Storm [FSE'20], AutoString [ICSE'20], Fusion [PLDI'20]

Differential Testing:
Not Applicable when Cross-checking Impossible

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.substr t 0 1)))
4  (assert (> (str.len (str.substr t 0 2)) 1))
5  (assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
6  (assert (>= (+ 0 2) (str.len t)))
7  (check-sat)
```

Differential Testing:
Not Applicable when Cross-checking Impossible

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.substr t 0 1)))
4  (assert (> (str.len (str.substr t 0 2)) 1))
5  (-)(assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
5' (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6  (-)(assert (>= (+ 0 2) (str.len t)))
6' (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

This mutant formula is satisfiable when $t \mapsto$ "-0"

Differential Testing:
Not Applicable when Cross-checking Impossible

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.substr t 0 1)))
4  (assert (> (str.len (str.substr t 0 2)) 1))
5  (-)(assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
5' (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6  (-)(assert (>= (+ 0 2) (str.len t)))
6' (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

```
$ cvc5 mutant.smt2
unsat
```

Differential Testing:
Not Applicable when Cross-checking Impossible

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.substr t 0 1)))
4  (assert (> (str.len (str.substr t 0 2)) 1))
5  (-)(assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
5' (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6  (-)(assert (>= (+ 0 2) (str.len t)))
6' (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

```
$ cvc5 mutant.smt2
unsat
```

```
$ z3 mutant.smt2
Error:Unsupported Function
```

Differential Testing:
Not Applicable when Cross-checking Impossible

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.substr t 0 1)))
4  (assert (> (str.len (str.substr t 0 2)) 1))
5  (-)(assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
5' (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6  (-)(assert (>= (+ 0 2) (str.len t)))
6' (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

```
$ cvc5 mutant.smt2
unsat
```

```
$ z3 mutant.smt2
Error:Unsupported Function
```

Cannot compare the results!

Existing Oracle-guided: No Diverse Mutants

```
1   (set-logic QF_SLIA)
2   (declare-fun t () String)
3   (assert (str.prefixof "-" (str.substr t 0 1)))
4   (assert (> (str.len (str.substr t 0 2)) 1))
5   (-)(assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
5'  (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6   (-)(assert (>= (+ 0 2) (str.len t)))
6'  (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

Hard to devise
various SAT-preserving mutation rules

Existing Oracle-guided: No Diverse Mutants

```
1   (set-logic QF_SLIA)
2   (declare-fun t () String)
3   (assert (str.prefixof "-" (str.s
4   (assert (> (str.len (str.substr t 0 2)) 1))
5   (-)(assert (not (  =  (- 1) (str.to_int (str.substr t 1 1)))))
5'  (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6   (-)(assert (>= (+ 0 2) (str.len t)))
6'  (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

"=" term is replaced with "xor" term

Hard to devise
various SAT-preserving mutation rules

Existing Oracle-guided: No Diverse Mutants

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.s
4  (assert (> (str.len (str.substr t 0 2)) 1))
5  (-)(assert (not (  =  (- 1) (str.to_int (str.substr t 1 1)))))
5' (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6  (-)(assert (>= (+ 0 2) (str.len t)))
6' (+)(assert (str.suffixof (str.replace t "-0"    -") "-"))
7 (check-sat)
```

"=" term is replaced with "xor" term

New function is introduced

Hard to devise
various SAT-preserving mutation rules

Limitations of Existing Approaches

- Differential Testing: **Not applicable when cross-checking impossible.**
- Oracle-Guided Testing: **No diverse mutants.**

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.substr t 0 1)))
4  (assert (> (str.len (str.substr t 0 2)) 1))
5  (-)(assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
5' (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6  (-)(assert (>= (+ 0 2) (str.len t)))
6' (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

```
$ cvc5 mutant.smt2
unsat
```

```
$ z3 mutant.smt2
Error:Unsupported Function
```

Cannot compare the results!

```
1  (set-logic QF_SLIA)
2  (declare-fun t () String)
3  (assert (str.prefixof "-" (str.
4  (assert (> (str.len (str.su
5  (-)(assert (not (  =  (- 1) (str.to_int (str.substr t 1 1)))))
5' (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6  (-)(assert (>= (+ 0 2) (str.len t)))
6' (+)(assert (str.suffixof (str.replace t "-0"  -") "-"))
7 (check-sat)
```

"=" term is replaced with "xor" term

New function is introduced

Hard to devise
various SAT-preserving mutation rules

Diver's Key Feature:
Oracle-Guided yet Unrestricted Mutations

```
1   (set-logic QF_SLIA)
2   (declare-fun t () String)
3   (assert (str.prefixof "-" (str.substr t 0 1)))
4   (assert (> (str.len (str.substr t 0 2)) 1))
5   (-)(assert (not (= (- 1) (str.to_int (str.substr t 1 1)))))
5'  (+)(assert (not (xor (str.< (str.update "-0" 0 t) "-0") false)))
6   (-)(assert (>= (+ 0 2) (str.len t)))
6'  (+)(assert (str.suffixof (str.replace t "-0" "-") "-"))
7 (check-sat)
```

Mutant Generated by Diver

- **Oracle-Guided**: no need to rely on cross-checking
- **Unrestricted**: can generate diverse mutants

Diver's Idea: Using Model as Oracle

- Find a mutant that satisfies the original seed's model.

$$(M \vDash \varphi) \Rightarrow (M \vDash \varphi')$$

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Goal:
Find mutants satisfying ***M***

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Mutant Formula

$$M \vDash (0 \leq y) \wedge (x = \boxed{\text{pow}(2, y)} + y)$$

Goal:
Find mutants satisfying *M*

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Mutant Formula

$$M \vDash (0 \leq y) \wedge (x = \boxed{\text{pow}(2, y)} + y)$$

true! true!

Goal:
Find mutants satisfying *M*

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Mutant Formula

$$M \models (0 \leq y) \wedge (x = \boxed{\text{pow}(2, y)} + y)$$

true! true!

$$M \not\models (\boxed{x + y} \leq y) \wedge (x = 2 + y)$$

Goal:
Find mutants satisfying *M*

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Mutant Formula

$$M \models (0 \leq y) \wedge (x = \boxed{\text{pow}(2, y)} + y)$$

true! true!

$$M \not\models (\boxed{x + y} \leq y) \wedge (x = 2 + y)$$

false! ($4 \leq 1$) true!

Goal:
Find mutants satisfying *M*

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Goal:
Find mutants satisfying M

Mutant Formula

$$M \models (0 \leq y) \wedge (x = \boxed{\text{pow}(2, y)} + y)$$

true! true! ✓

$$M \not\models (\boxed{x + y} \leq y) \wedge (x = 2 + y)$$

false! ($4 \leq 1$) true! ✗

$$M \models (0 \leq \boxed{\frac{y - 1}{x}}) \wedge (x = 2 + y)$$

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Goal:
Find mutants satisfying M

Mutant Formula

$$M \models (0 \leq y) \wedge (x = \boxed{\text{pow}(2, y)} + y)$$

true! true! ✓

$$M \not\models (\boxed{x + y} \leq y) \wedge (x = 2 + y)$$

false! ($4 \leq 1$) true! ✗

$$M \models (0 \leq \boxed{\frac{y - 1}{x}}) \wedge (x = 2 + y)$$

true! true! ✓

Diver's Idea: Using Model as Oracle

Seed Formula

M: $\{ x \mapsto 3, y \mapsto 1 \}$

$$(0 \leq y) \wedge (x = 2 + y)$$

Goal:
Find mutants satisfying M

Mutant Formula

$$M \models (0 \leq y) \wedge (x = \boxed{\text{pow}(2, y)} + y)$$

true! true! ✓

$$M \not\models (\boxed{x + y} \leq y) \wedge (x = 2 + y)$$

false! ($4 \leq 1$) true! ✗

$$M \models (0 \leq \boxed{\frac{y - 1}{x}}) \wedge (x = 2 + y)$$

true! true! ✓

Can generate diverse mutants without differential testing.

Evaluation: Setup

- We tested the recent version of three SMT solvers during 4-5 months.
  - Z3: https://github.com/Z3Prover/z3
  - CVC5: https://github.com/cvc5/cvc5
  - dReal4: https://github.com/dreal/dreal4
- We focused on the theories for String, Real, and Integer.
  - QF_SLIA, QF_S, QF_NRA, …

RQ1: Bug-Finding in Z3, CVC5, dReal

"Statistics on Bugs found by Diver"

Status	Z3	CVC5	dReal	Total
Reported	12	15	2	29
Duplicate	4	0	0	0
New	8	15	2	25
- Confirmed	4	15	2	21
- Fixed	0	14	0	14

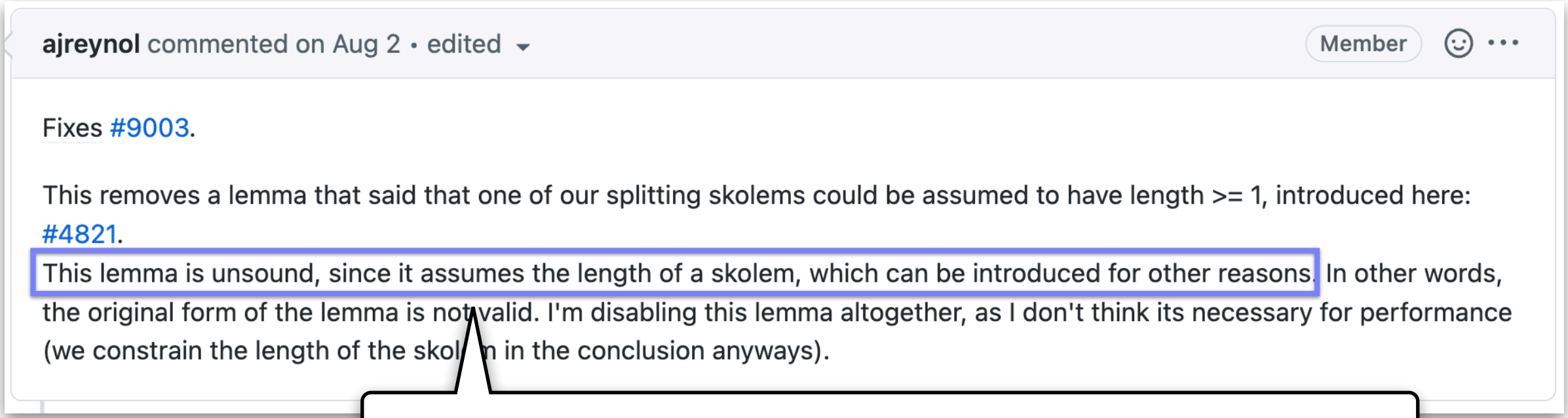
"Statistics on Types of 25 new bugs"

Bug Type	Z3	CVC5	dReal	Total
Soundness	6	4	2	12
Invalid-Model	2	7	0	9
Crash	0	4	0	4

- Diver found 25 new bugs.
- 21 out of 25 new bugs are critical bugs.

RQ1: Bug-Finding in Z3, CVC5, dReal

- Positive response from the CVC5 developer.



The lemma, which are used in CVC5's string solver, is unsound.

- This lemma had been used for 2 years (Aug 2, 2020).

RQ2: Comparison with Existing Tools

- Compared with 5 Existing Tools
  - 2 differential testing-based tools: TypeFuzz[OOPSLA'21], OpFuzz[OOPSLA'20]
  - 3 oracle-guided tools: Storm[FSE'20], AutoString[ICSE'20], Fusion[PLDI'20]

- Settings
  - Benchmarks: 25 bugs found by Diver
  - Time budget: 1 hours each a benchmark (Repeated 30 times)

RQ2: Comparison with Existing Tools

Bug Type	No	Solver	Theory	Confirmed	Specific	Oracle-Guided			Differential Testing	
						Storm	AutoString	Fusion	TypeFuzz	OpFuzz
Soundness	1	CVC5 (v 1.0.1)	QF_SLIA	✔	✔	✗	✗	✗	✗	✗
	2	CVC5 (v 1.0.1)	QF_S	✔	✗	✗	✗	✗	✗	✗
	3	CVC5 (v 1.0.0)	QF_SLIA	✔	✔	✗	✗	✗	✗	✗
	4	CVC5 (v 1.0.0)	QF_NRA	✔	✗	✗	N/A	✗	✗	✗
	5	Z3 (v 4.11.0)	QF_SLIA	✔	✗	✗	✗	✗	✗	✗
	6	Z3 (v 4.11.0)	QF_SLIA	✔	✗	✗	✗	✗	✗	✗
	7	Z3 (v 4.11.0)	QF_SLIA	✔	✗	✗	✗	✗	✗	✗
Invalid-Mode	18	CVC5 (0bf059f)	QF_SLIA	✔	✗	✗	✗	✗	✔	✔
	19	CVC5 (v 1.0.0)	QF_LIA	✔	✔	✗	N/A	✗	✗	✗
	20	Z3 (v 4.11.0)	QF_S	✔	✗	✗	✗	✗	✔	✗
	21	Z3 (v 4.11.0)	QF_NIA	✗	✗	✗	N/A	✗	✔	✗
Crash	22	CVC5 (v 1.0.1)	QF_LIA	✔	✗	✗	N/A	✗	✗	✗
	23	CVC5 (v 1.0.1)	QF_SLIA	✔	✗	✔	✗	✔	✗	✗
	24	CVC5 (v 1.0.1)	QF_S	✔	✔	✗	✗	✗	✗	✗
	25	CVC5 (0bf059f)	QF_SLIA	✔	✗	✗	✗	✗	✗	✗
Total				✔: 21 ✗ : 4	✔: 7 ✗ : 18	✔: 2 ✗ : 23	✔: 0 ✗ : 16	✔: 3 ✗ : 22	✔: 4 ✗ : 21	✔: 1 ✗ : 24

The existing tools collectively detected only 7 out of 25.

- 3 oracle-guided tools found 3 out of 25.
- 2 differential testing tools found 4 out of 25.

Summary

- **Diver: New Oracle-Guided Fuzzer with Unrestricted Mutations**
  - Existing approaches could not generate diverse mutants or struggled with test local problem.
- **Diver found 25 new bugs in CVC5, Z3, dReal.**
  - Existing tools managed to find 7 out of 25 found by Diver.
- In the paper: "weighted sampling" for improving performance.

Thank you