Satisfiability Modulo Fuzzing

A Synergistic Combination of SMT Solving and Fuzzing

(Appeared in OOPSLA'22)

Sujit Kumar Muduli, Subhajit Roy

Indian Institute of Technology Kanpur

May 30, 2023

Introduction

$$(u = 6) \wedge (\mathtt{bar}(u, v) = 42)$$

Introduction

$$(\mathfrak{u} = 6) \wedge (\mathtt{bar}(\mathfrak{u}, \mathfrak{v}) = 42)$$

- External library call
- Web API call
- ML model

Introduction

$$(u = 6) \wedge (\mathtt{bar}(u, v) = 42)$$

Closed-box (CB) Function

Problem Statement

Test satisfiability of first-order logic constraints containing closed-box (CB) functions

Problem Statement

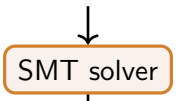
Test satisfiability of first-order logic constraints containing closed-box (CB) functions

Closed-Box (CB) Function

1. It must be functional
2. An input-output oracle interface is available

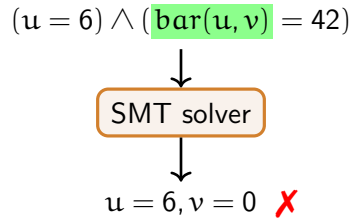
Constraint Solving with UF Theory

$$(u = 6) \wedge (\mathtt{bar}(u, v) = 42)$$



$$u = 6, v = 0$$

Constraint Solving with UF Theory



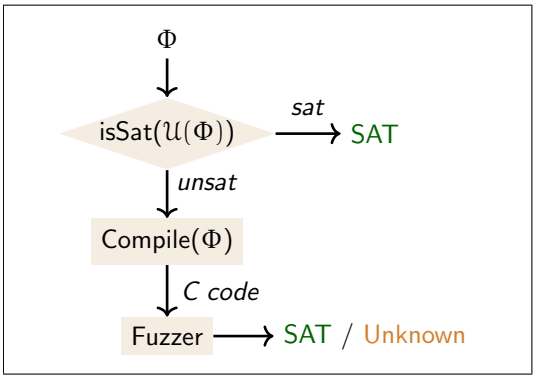

```
u32 bar(u32 u, u32 v) {
   return u * v;
}
```

Core Idea

1. **Reduce satisfiability of a formula to reachability in a program**
2. **Use fuzzing to solve the reachability problem**

Prior Works

COLOSSUS[1]

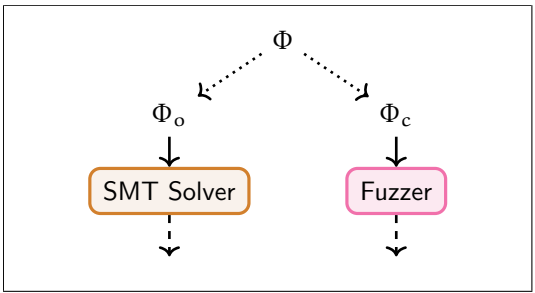


- Underapproximate the input formula and check for satisfiability
- If the underapproximated formula is UNSAT, Colossus uses fuzzing

[1] Awanish Pandey, Phani Raj Goutham Kotcharlakota, and Subhajit Roy. "Deferred Concretization in Symbolic Execution via Fuzzing". In: ISSTA 2019.

Prior Works

ACHAR[2]



- Create a disjunctive partition of open-box and closed-box components
- Use SMT for open-box and fuzzing for closed-box

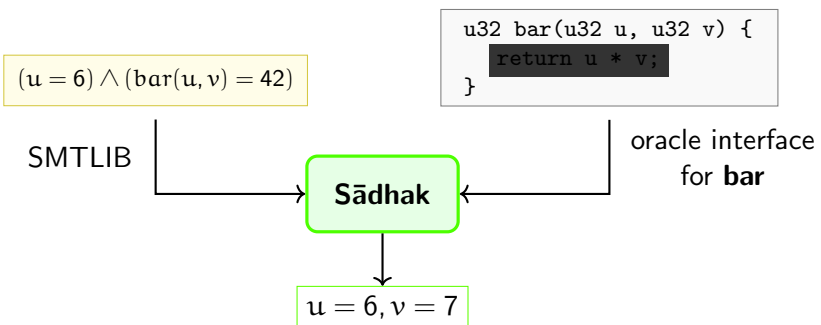
[2] Sumit Lahiri and Subhajit Roy. "Almost Correct Invariants: Synthesizing Inductive Invariants by Fuzzing Proofs". In: ISSTA 2022.

Contributions

1. Introduced CB theory to support closed-box functions in SMT solvers
2. A conflict-driven fuzz loop (CDFL) algorithm for solving CB constraints

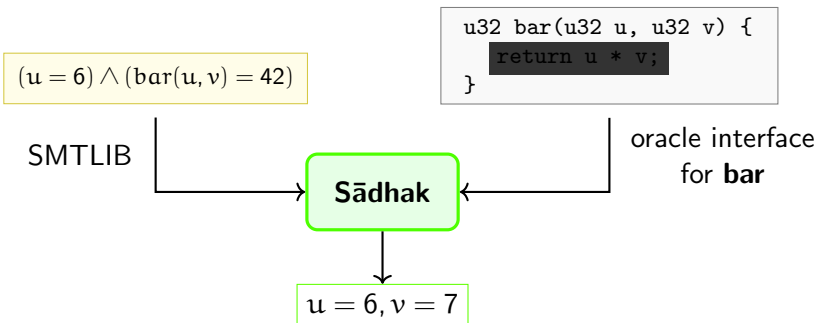
Synergistic Combination of SMT Solving and Fuzzing

In SĀDHAK SMT solver and fuzzer communicate with each other, exchanging information for efficient solving of CB constraints.



Synergistic Combination of SMT Solving and Fuzzing

In SĀDHAK SMT solver and fuzzer communicate with each other, exchanging information for efficient solving of CB constraints.



SĀDHAK is sound but not complete

SMTLIB Encoding

$$(u = 6) \wedge (bar(u, v) = 42)$$

(SMTLIB encoding of above constraint with closed-box function)

```
1 (declare-const u (_ BitVec 32))
2 (declare-const v (_ BitVec 32))
3 (declare-cb bar ((_ BitVec 32) (_ BitVec 32)) (_ BitVec 32))
4 (assert (= u (_ bv6 32)))
5 (assert (= (bar u v) (_ bv42 32)))
6 (check-sat)
7 (get-model)
```

CB Theory

$$\sum_{CB}\{T_1, T_2, \dots\} = \langle S, C, F, F^{CB}, B, R \rangle$$

- S : set of sorts in theories
- C : set of (sorted) constants in theories
- F : set of all (sorted) function symbols
- F^{CB} : set of (sorted) closed-box functions
- B : predicates from theories
- R : schema for translating sorts and expressions into a program

Example

closed-box function

$$\mathbf{f}(x, y) > 255 \wedge \mathbf{f}(x, y) < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r) \wedge (p \times q \times r = 64)$$

$$\begin{aligned}&\mathbf{f}(x,y) > 255 \wedge \mathbf{f}(x,y) < 65536\\ \wedge\ &(x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)\\ \wedge\ &\mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r) \wedge (p \times q \times r = 64)\end{aligned}$$

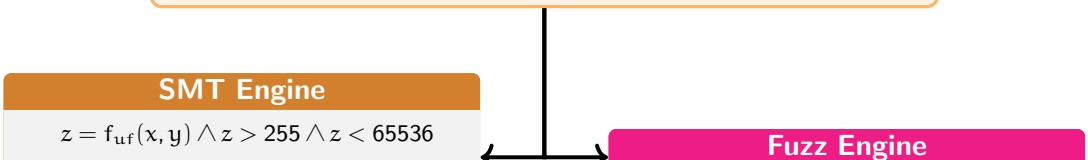
$$\begin{aligned}&\mathbf{z} = \mathbf{f}(x,y) \wedge \mathbf{z} > 255 \wedge \mathbf{z} < 65536\\ \wedge\ &(x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)\\ \wedge\ &\mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r) \wedge (p \times q \times r = 64)\end{aligned}$$

(Purification)

Segregation

Separation

$$z = f(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r) \wedge (p \times q \times r = 64)$$



(Separation)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 1

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$

Fuzz Engine

$$z = f_{cb}(x, y)$$

Conflict-driven Fuzz Loop (CDFL)

Iteration - 1

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ isPow2(p) \wedge isPow2(q) \wedge isPow2(r)$$
$$\wedge\ (p \times q \times r = 64)$$

Fuzz Engine

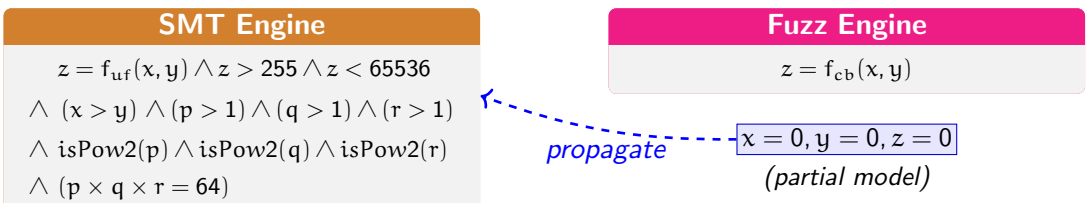
$$z = f_{cb}(x, y)$$

$x = 0, y = 0, z = 0$

(partial model)

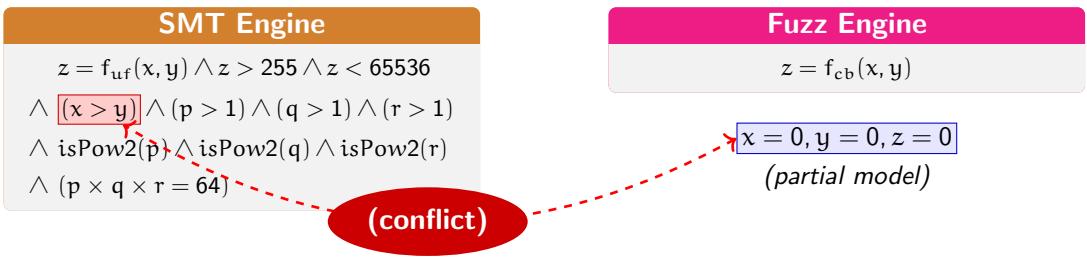
Conflict-driven Fuzz Loop (CDFL)

Iteration - 1



Conflict-driven Fuzz Loop (CDFL)

Iteration - 1



Conflict-driven Fuzz Loop (CDFL)

Iteration - 1

SMT Engine

$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$

$\wedge\ (x > y) \wedge (p \geq 1) \wedge (q > 1) \wedge (r > 1)$

$\wedge\ isPow2(p) \wedge isPow2(q) \wedge isPow2(r)$

$\wedge\ (p \times q \times r = 64)$

lemma

Fuzz Engine

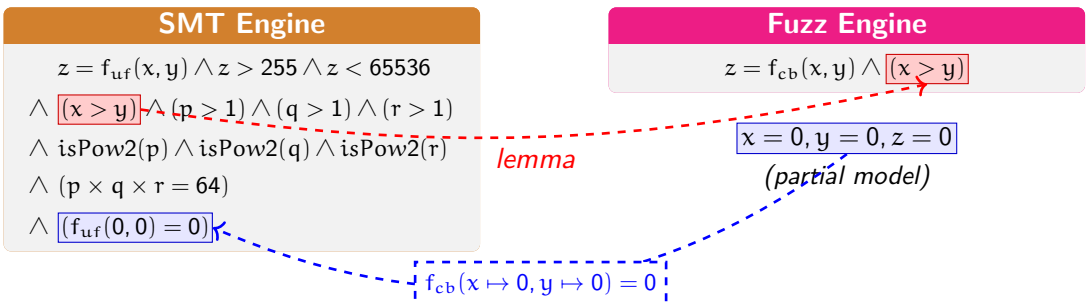
$z = f_{cb}(x, y) \wedge (x > y)$

$x = 0, y = 0, z = 0$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 1



Conflict-driven Fuzz Loop (CDFL)

Iteration - 2

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y)$$

Conflict-driven Fuzz Loop (CDFL)

Iteration - 2

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ isPow2(p) \wedge isPow2(q) \wedge isPow2(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y)$$

$\boxed{x = 1, y = 0, z = 0}$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 2

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y)$$

propagate

$x = 1, y = 0, z = 0$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 2

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \text{isPow2}(p) \wedge \text{isPow2}(q) \wedge \text{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$

(conflict)

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y)$$

$x = 1, y = 0, z = 0$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 2

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$

lemma

Fuzz Engine

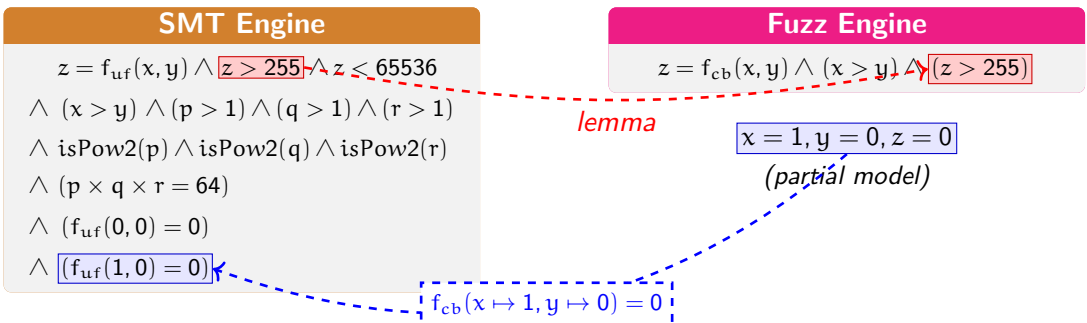
$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$

$x = 1, y = 0, z = 0$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 2



Conflict-driven Fuzz Loop (CDFL)

Iteration - 3

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$

Conflict-driven Fuzz Loop (CDFL)

Iteration - 3

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$

$$\boxed{x = 65536, y = 1, z = 65536}$$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 3

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$

propagate

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$

$x = 65536, y = 1, z = 65536$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 3

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$

(conflict)

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$

$x = 65536, y = 1, z = 65536$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 3

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge \boxed{z < 65536}$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$

lemma

Fuzz Engine

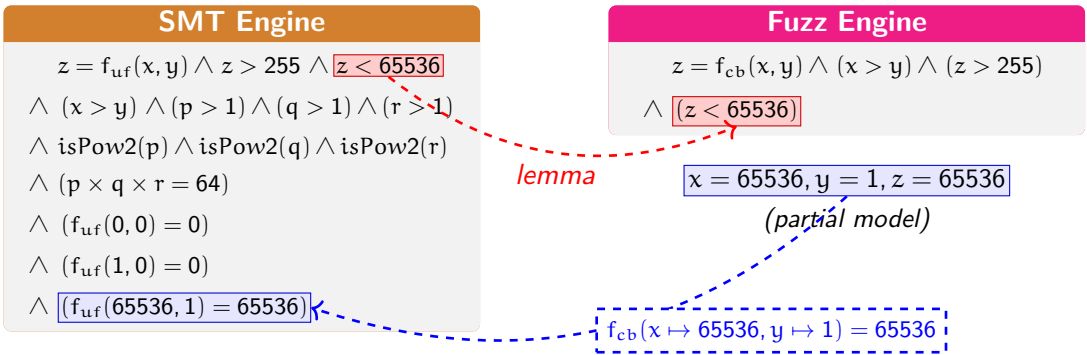
$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$
$$\wedge\ \boxed{(z < 65536)}$$

$$\boxed{x = 65536, y = 1, z = 65536}$$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 3



Conflict-driven Fuzz Loop (CDFL)

Iteration - 4

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$
$$\wedge\ (f_{uf}(65536, 1) = 65536)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$
$$\wedge\ (z < 65536)$$

Conflict-driven Fuzz Loop (CDFL)

Iteration - 4

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$
$$\wedge\ (f_{uf}(65536, 1) = 65536)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$
$$\wedge\ (z < 65536)$$

$x = 256, y = 1, z = 256$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 4

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$
$$\wedge\ (f_{uf}(65536, 1) = 65536)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$
$$\wedge\ (z < 65536)$$

propagate

$x = 256, y = 1, z = 256$

(partial model)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 4

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \mathrm{isPow2}(p) \wedge \mathrm{isPow2}(q) \wedge \mathrm{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$
$$\wedge\ (f_{uf}(65536, 1) = 65536)$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$
$$\wedge\ (z < 65536)$$

$x = 256, y = 1, z = 256$

(partial model)

(consistent)

Conflict-driven Fuzz Loop (CDFL)

Iteration - 4

SMT Engine

$$z = f_{uf}(x, y) \wedge z > 255 \wedge z < 65536$$
$$\wedge\ (x > y) \wedge (p > 1) \wedge (q > 1) \wedge (r > 1)$$
$$\wedge\ \text{isPow2}(p) \wedge \text{isPow2}(q) \wedge \text{isPow2}(r)$$
$$\wedge\ (p \times q \times r = 64)$$
$$\wedge\ (f_{uf}(0, 0) = 0)$$
$$\wedge\ (f_{uf}(1, 0) = 0)$$
$$\wedge\ (f_{uf}(65536, 1) = 65536)$$

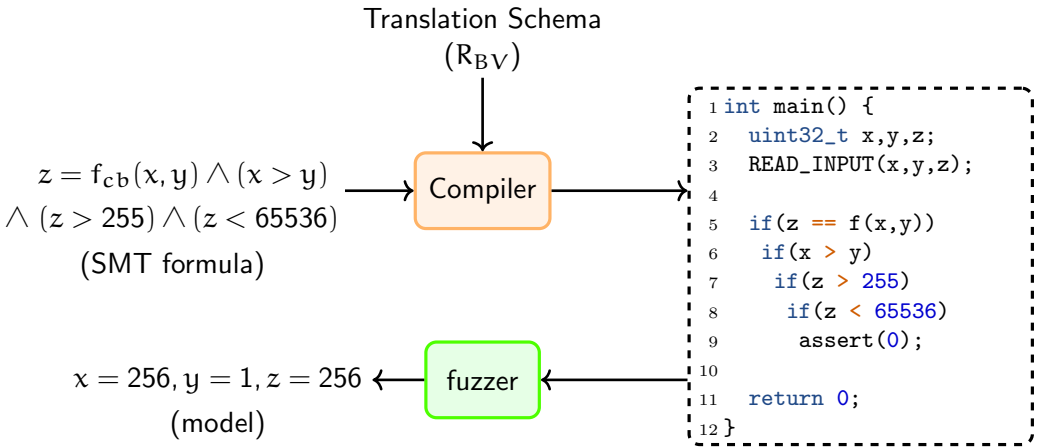
$$x = 256, y = 1, z = 256,$$
$$p = 2, q = 8, r = 4$$

Fuzz Engine

$$z = f_{cb}(x, y) \wedge (x > y) \wedge (z > 255)$$
$$\wedge\ (z < 65536)$$

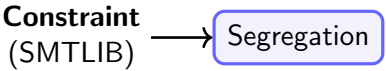
```
u32 f(u32 x, u32 y) {
  return x * y;
}
```

Fuzz Engine

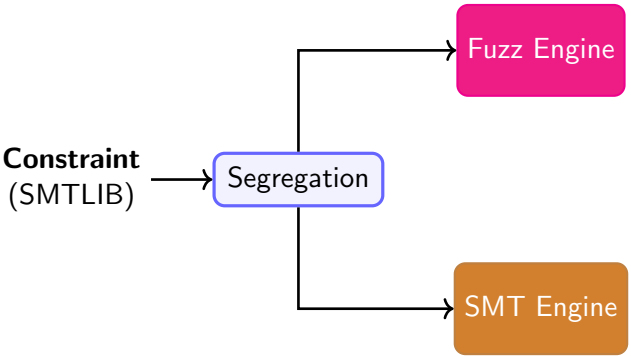

Translation Schema
(R_{BV})
$z = f_{cb}(x, y) \wedge (x > y)$
$\wedge\ (z > 255) \wedge (z < 65536)$
(SMT formula)
Compiler
1 int main() {
2 uint32_t x,y,z;
3 READ_INPUT(x,y,z);
4
5 if(z == f(x,y))
6 if(x > y)
7 if(z > 255)
8 if(z < 65536)
9 assert(0);
10
11 return 0;
12 }
fuzzer
$x = 256, y = 1, z = 256$
(model)

Constraint
(SMTLIB)

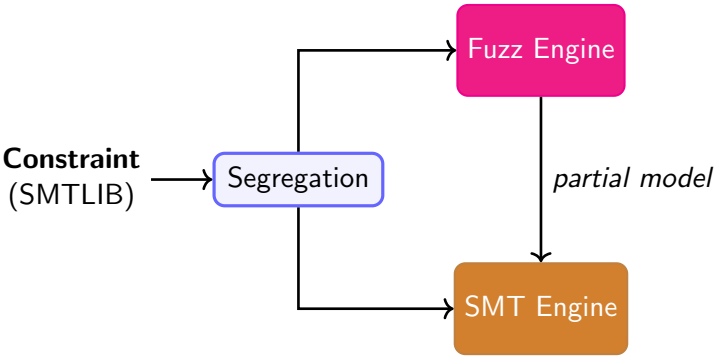
SĀDHAK: An SMT Solver for CB Constraints



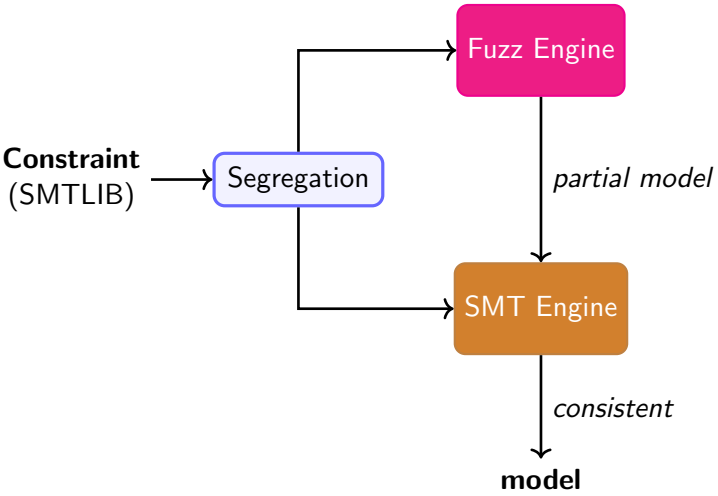
SĀDHAK: An SMT Solver for CB Constraints



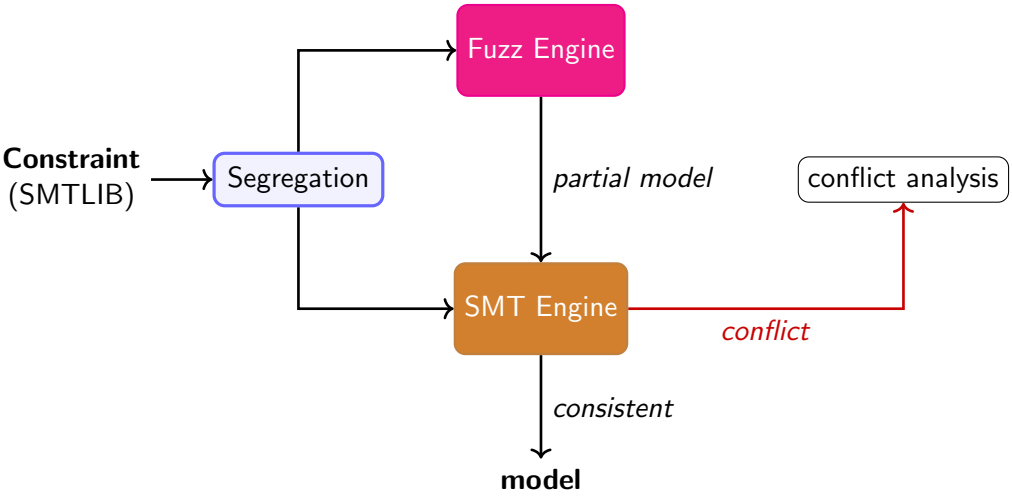
SĀDHAK: An SMT Solver for CB Constraints



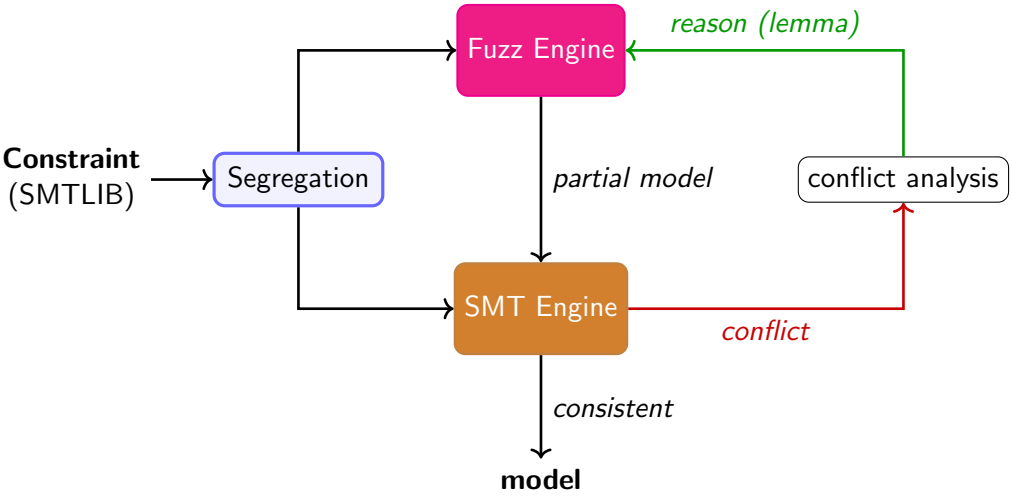
SĀDHAK: An SMT Solver for CB Constraints



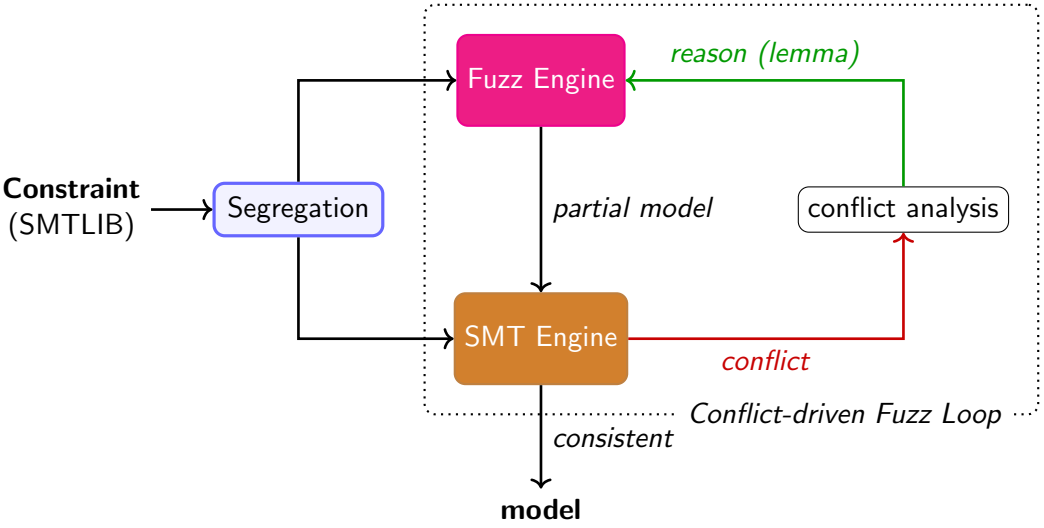
SĀDHAK: An SMT Solver for CB Constraints



SĀDHAK: An SMT Solver for CB Constraints

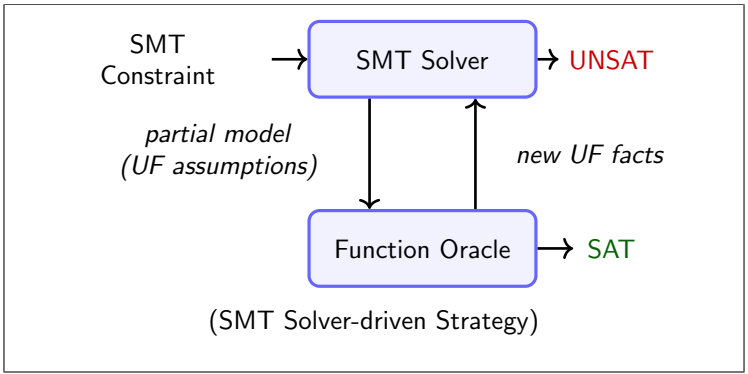


SĀDHAK: An SMT Solver for CB Constraints



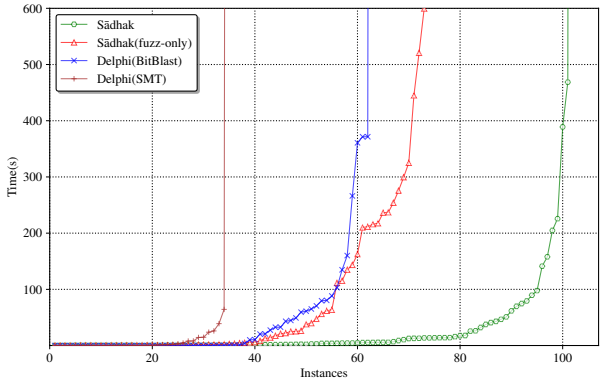
Related Work

DELPHI[3]



[3] Elizabeth Polgreen, Andrew Reynolds, and Sanjit A. Seshia. "Satisfiability and Synthesis Modulo Oracles". In: VMCAI 2022.

Evaluation



	Sādhak		Delphi	
	Fuzz only	CDFL	SMT	BitBlast
# solved	73	**101**	34	62

Conclusion

- Introduced closed-box function theory (CB theory)
- Conflict-driven fuzz loop (CDFL) for using fuzzing in synergy with the SMT solving
- Sādhak is built on top of CVC4 SMT solver.
- A set of 95 new benchmarks (SMTLIB queries with closed-box constraints)

(paper and artifact)