INSIGHT: Automatic Generation of Explanations for Efficient Identification of Hardware Bugs and Underspecifications

Vincent Quentin Ulitzsch (MIT), Alessandro Bertani (Politecnico di Milano/MIT), Peter W. Deutsch (MIT), David Langus Rodriguez (MIT Lincoln Lab), Kelly Xu (MIT), Aarti Gupta (Princeton), Sharad Malik (Princeton), Mengjia Yan (MIT)

viniul@mit.edu

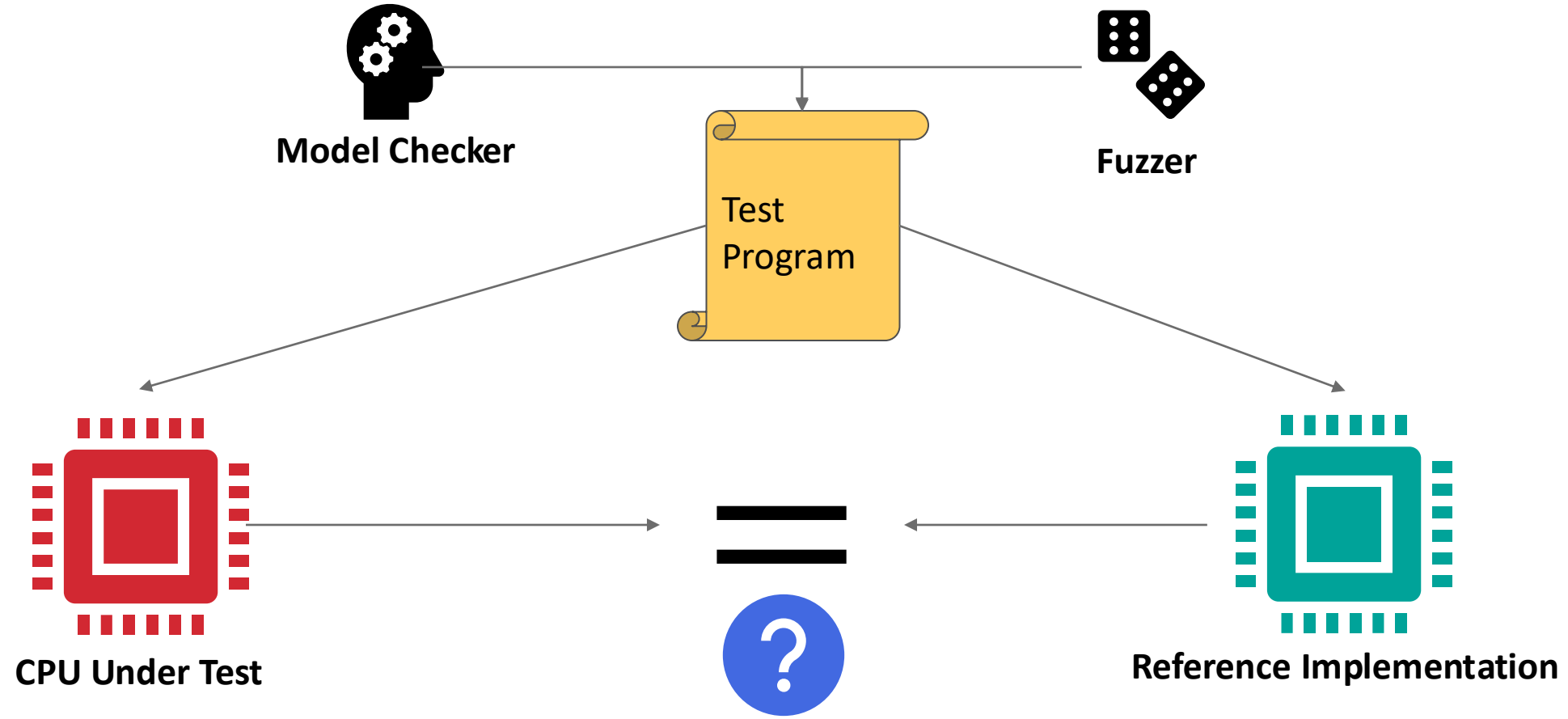
Methods to find bugs in CPUs do not give explanations

This creates manual overhead

Paper Contribution: **INSIGHT**

A framework to synthesize machine-readable explanations for CPU bugs

Background: Equivalence checking for bug finding



Counterexample programs (**CEX**) can point to bugs

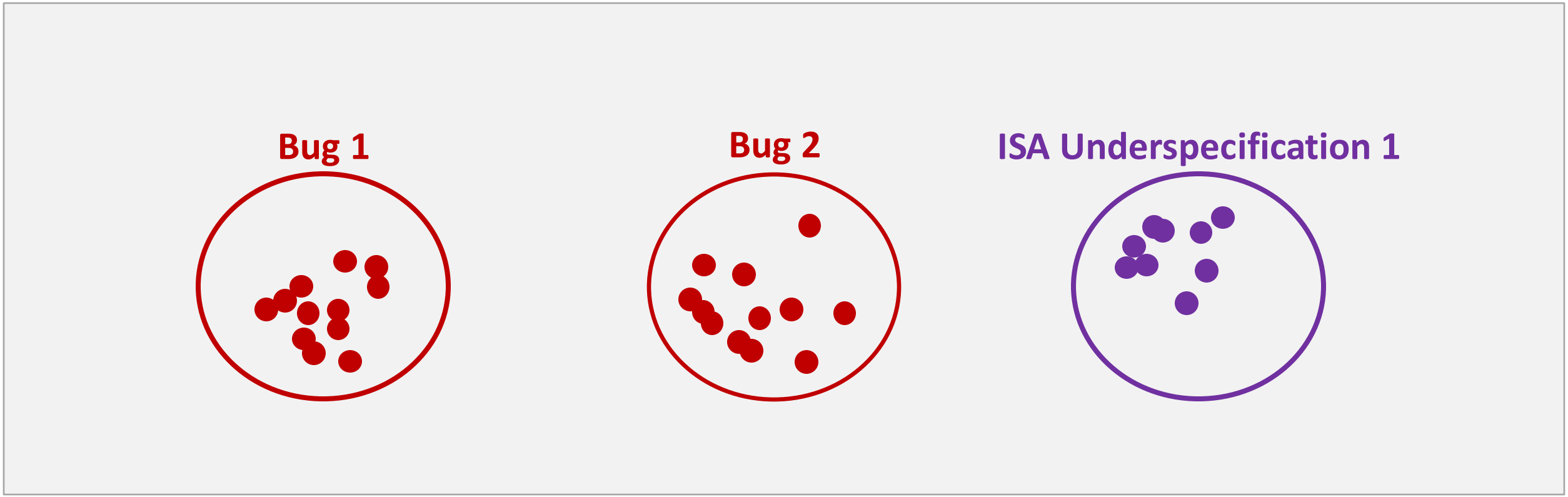
Challenge: Not every CEX is a new bug

New CEX != New Bug

1. There are many CEX per bug
2. ISA underspecification can cause spurious CEX

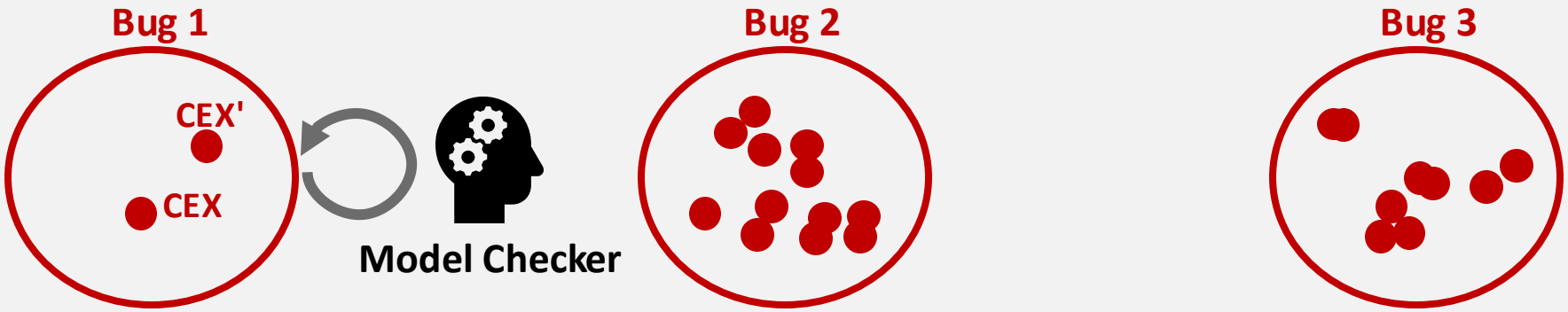
Typical Search Space: New CEX != New Bug

Search Space of Programs



Model checkers get stuck

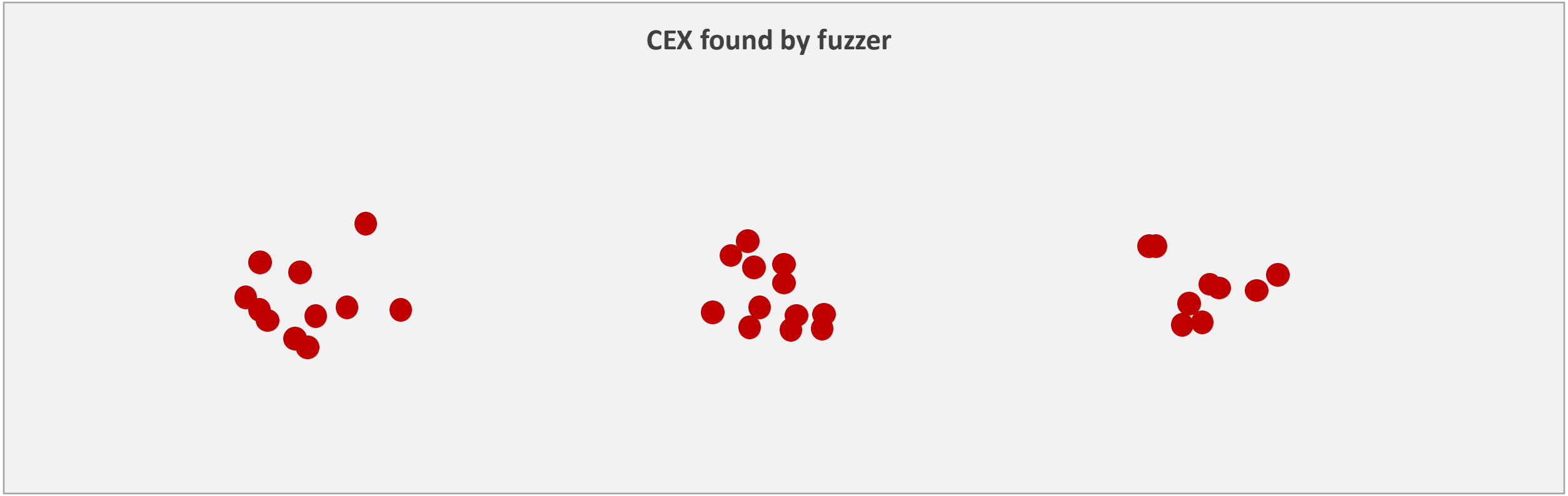
Search Space of Programs



Model-checkers rediscover the same CEX when repeatedly queried

Fuzzers need manual triage work

Search Space of Programs

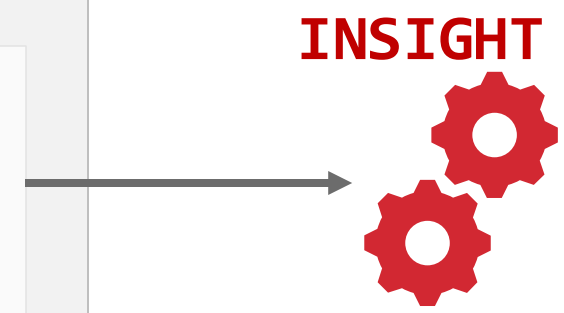


Fuzzer testcases need manually deduplication and debugging

INSIGHT: From program to explanation

Input: CEX program

```
addi x1, x0, 1
addi x1, x0, 2
add x2, x1, x0
```



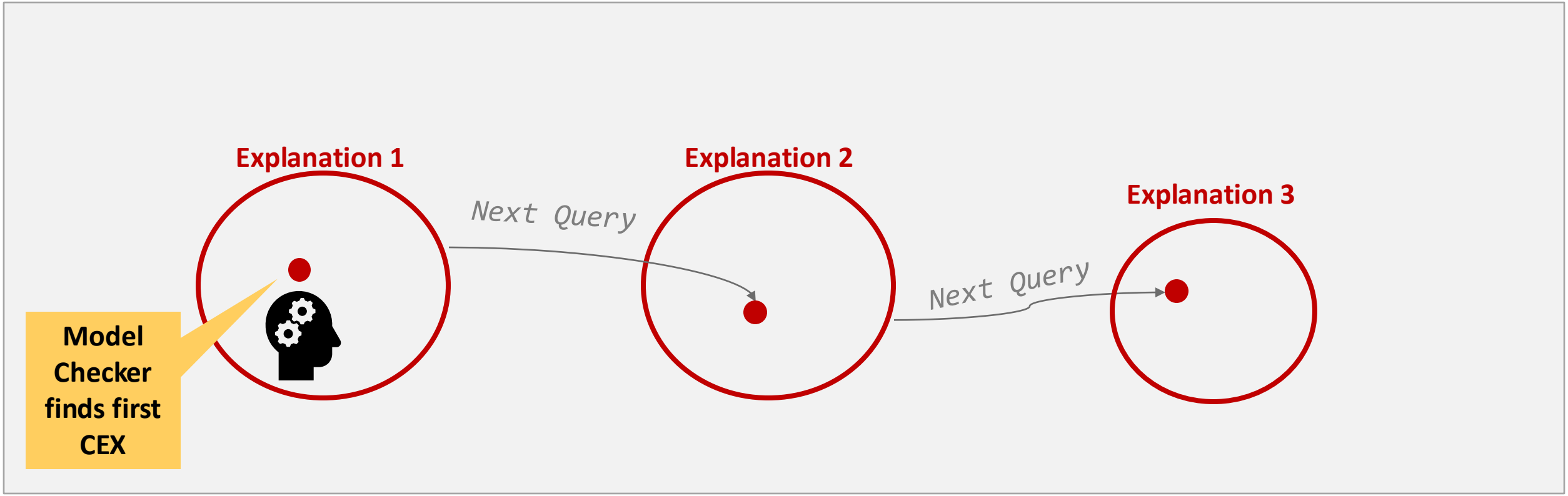
Output: Explanation as logic formula

Explanation: forwarding is active during a hazard

```
rs1_forward == 1 &&
exe_stage.target ==
dec_stage.source
```

INSIGHT unblocks model checking

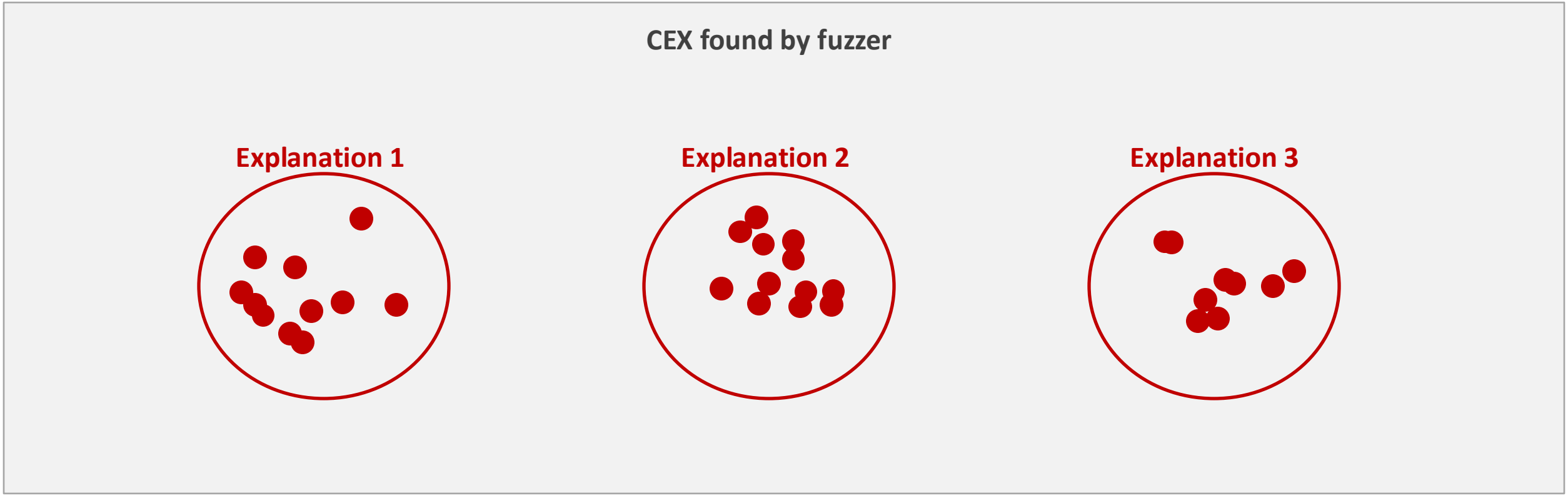
Search Space of Programs



Explanations push model checker into new mismatch classes

INSIGHT enables CEX deduplication

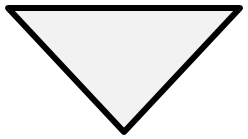
Search Space of Programs



INSIGHT can cluster large set of CEX by explanation

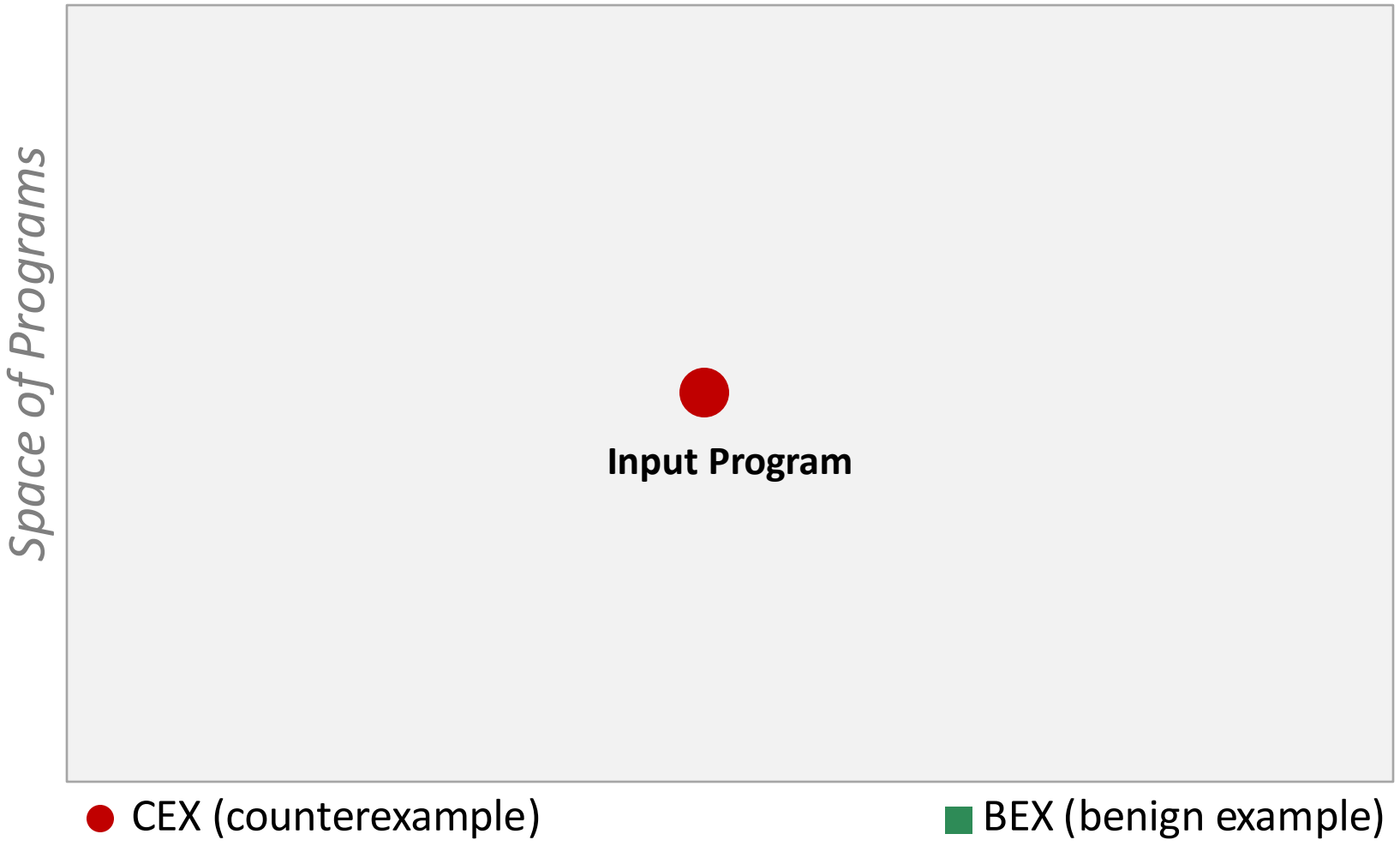
INSIGHT's explanation is a separating formula

1 **Create small mutations of the input program**



2 **Find a formula that separates CEX from non-CEX mutations**

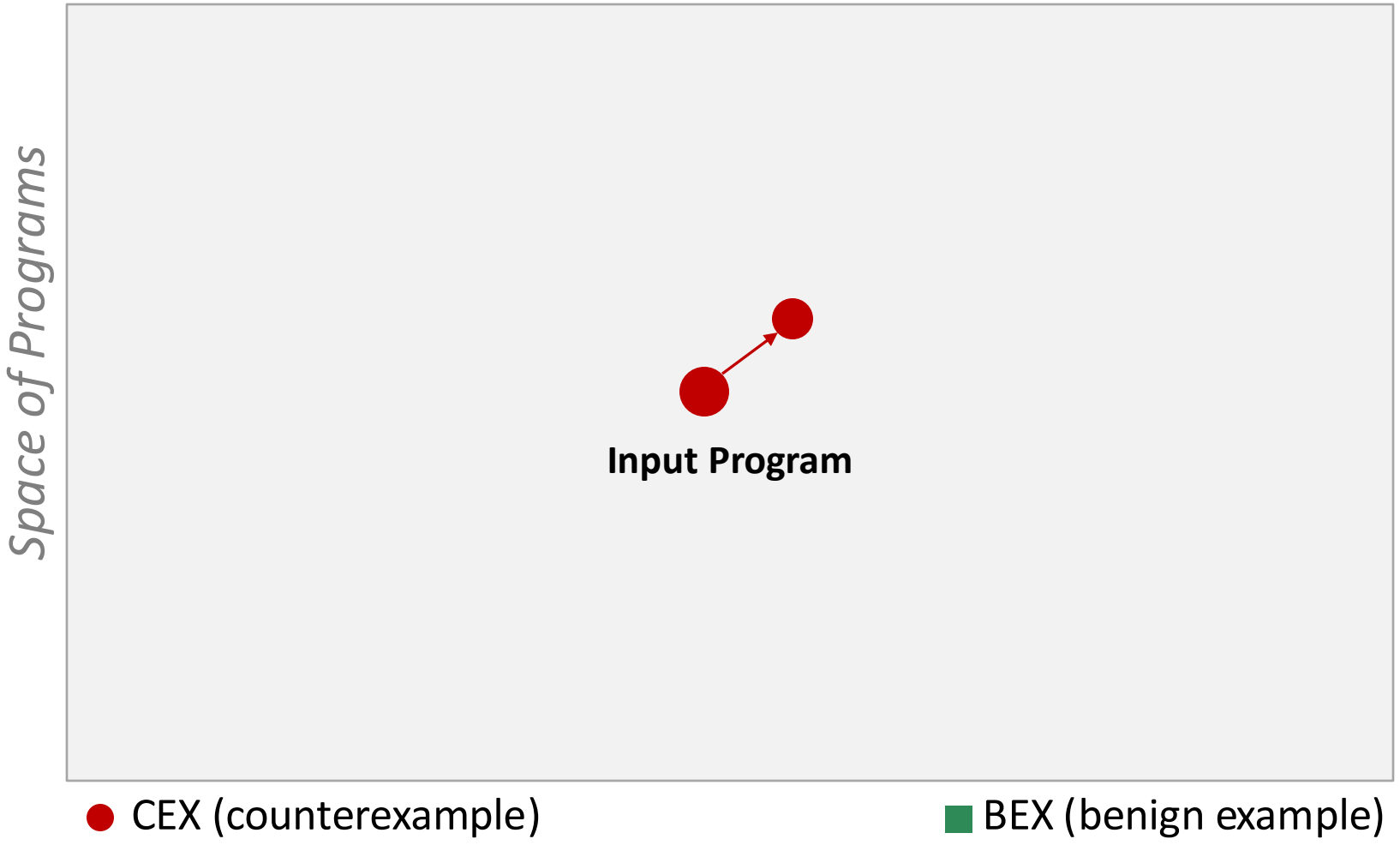
INSIGHT Step 1: Mutations of the input program



Input Program

```
addi x1, x0, 1
addi x1, x0, 2
add x2, x1, x0
```

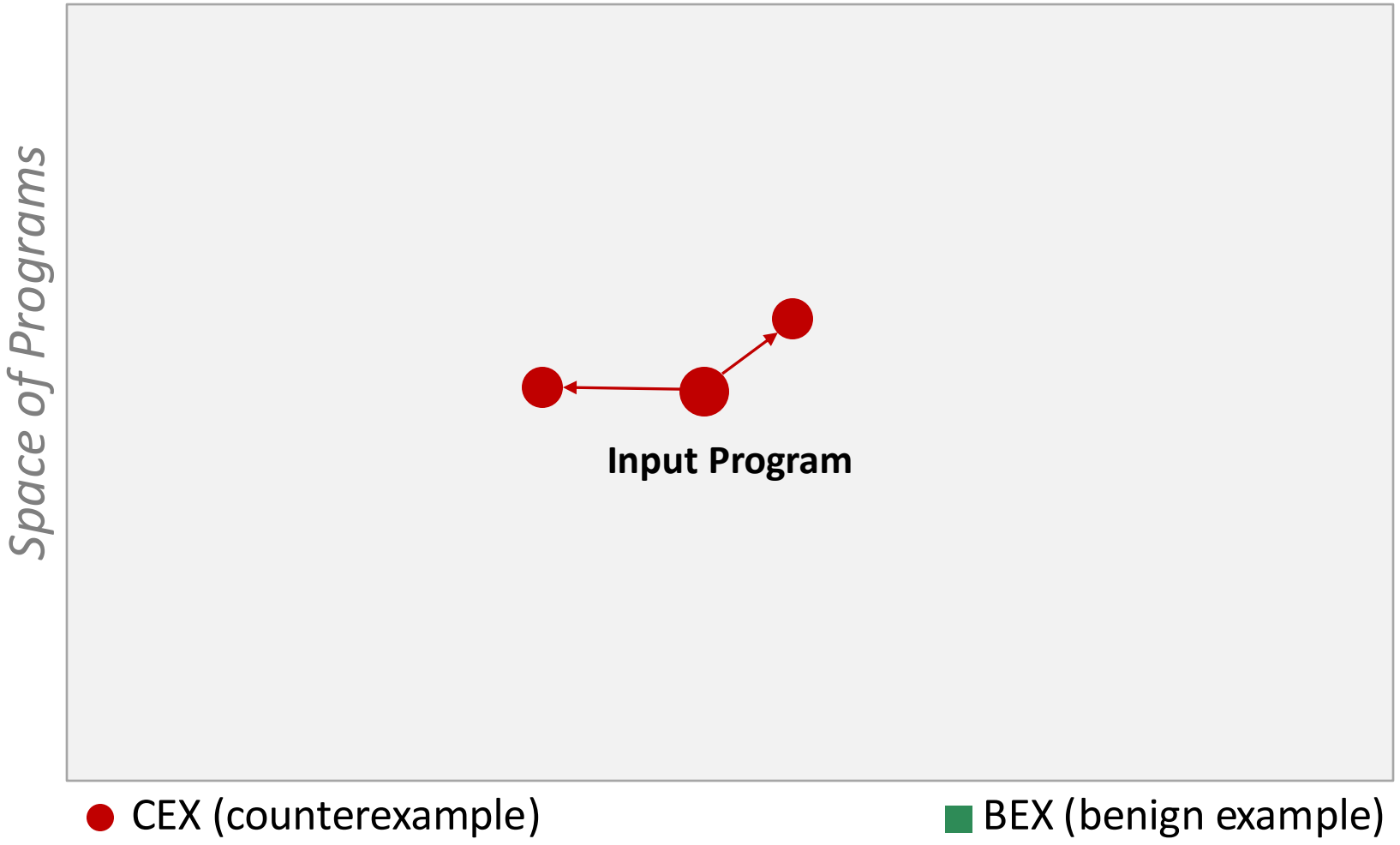
INSIGHT Step 1: Mutations of the input program



Current Mutation

```
addi x3, x0, 1
addi x3, x0, 2
add x2, x3, x0
```

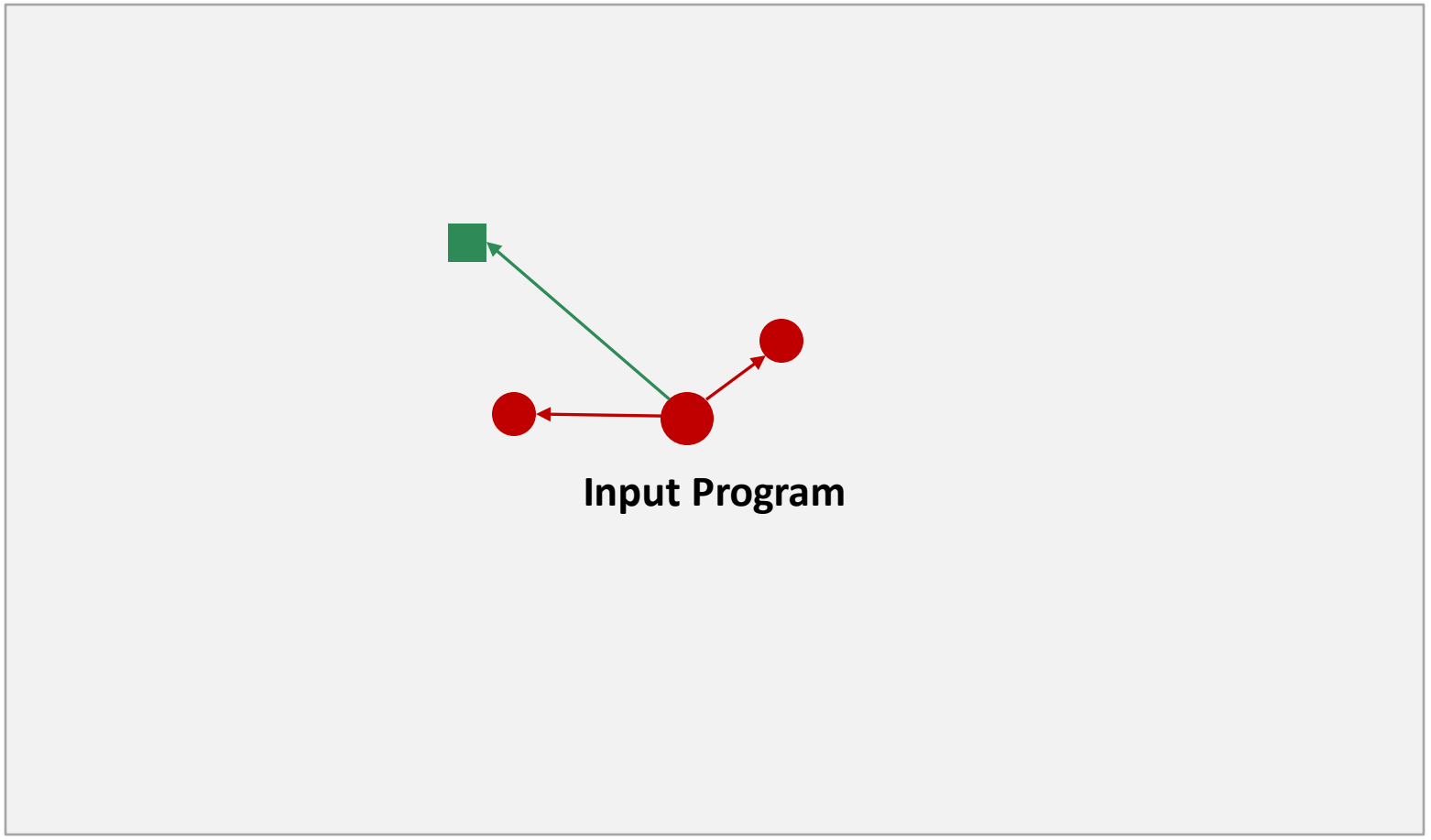
INSIGHT Step 1: Mutations of the input program



Current Mutation

```
addi x1, x0, 1
addi x1, x0, 2
sub x2, x1, x0
```

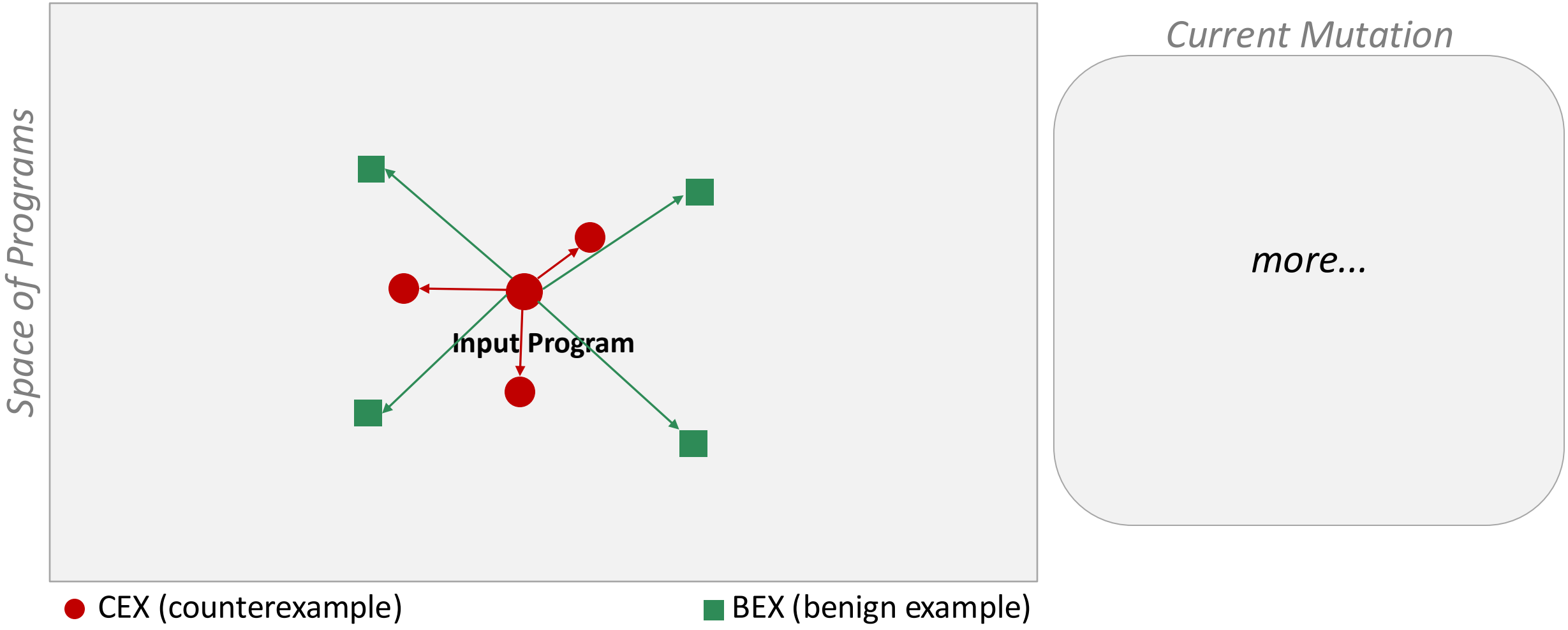
INSIGHT Step 1: Mutations of the input program



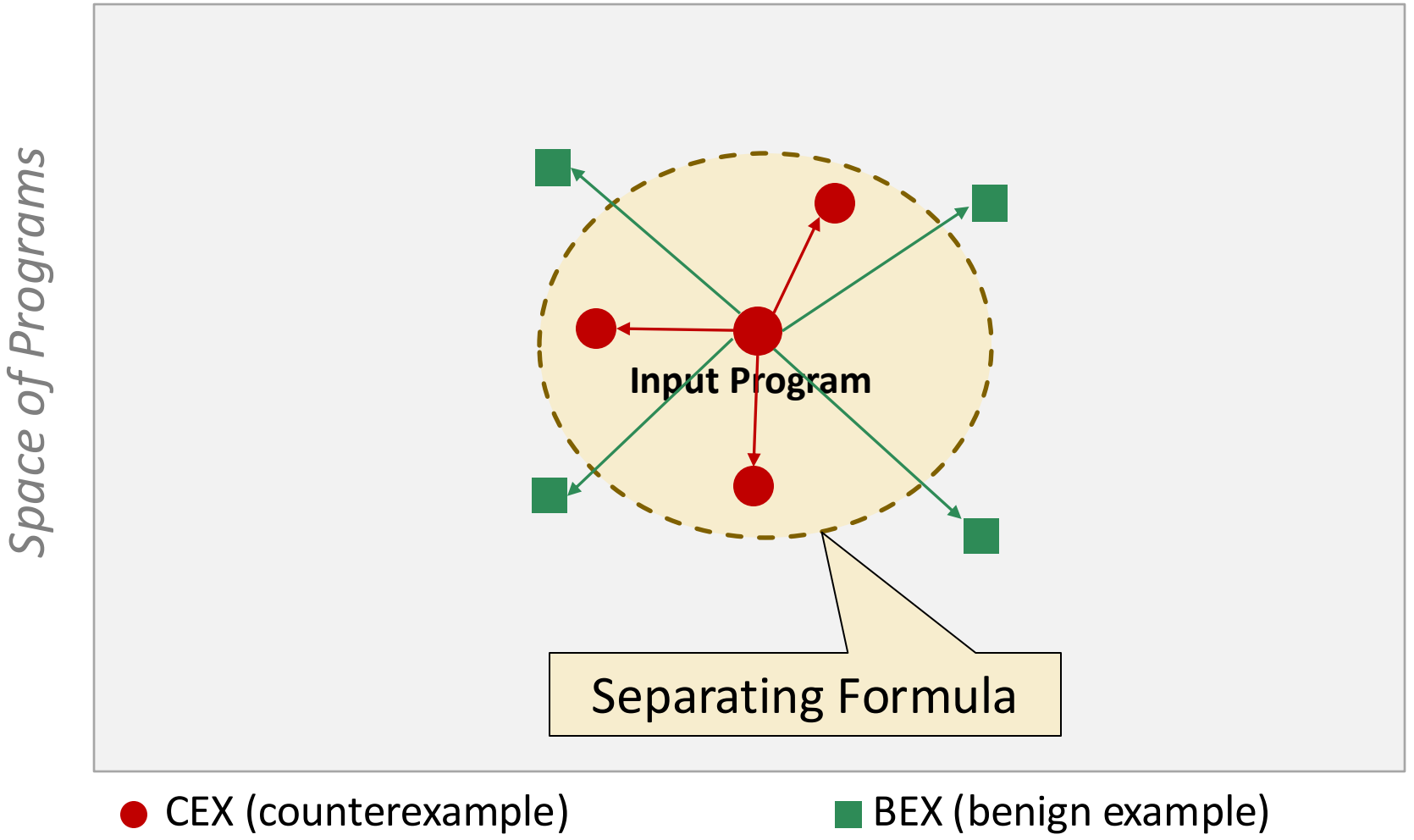
Current Mutation

```
addi x1, x0, 1
addi x1, x0, 2
add x2, x3, x0
```

INSIGHT Step 1: Mutations of the input program



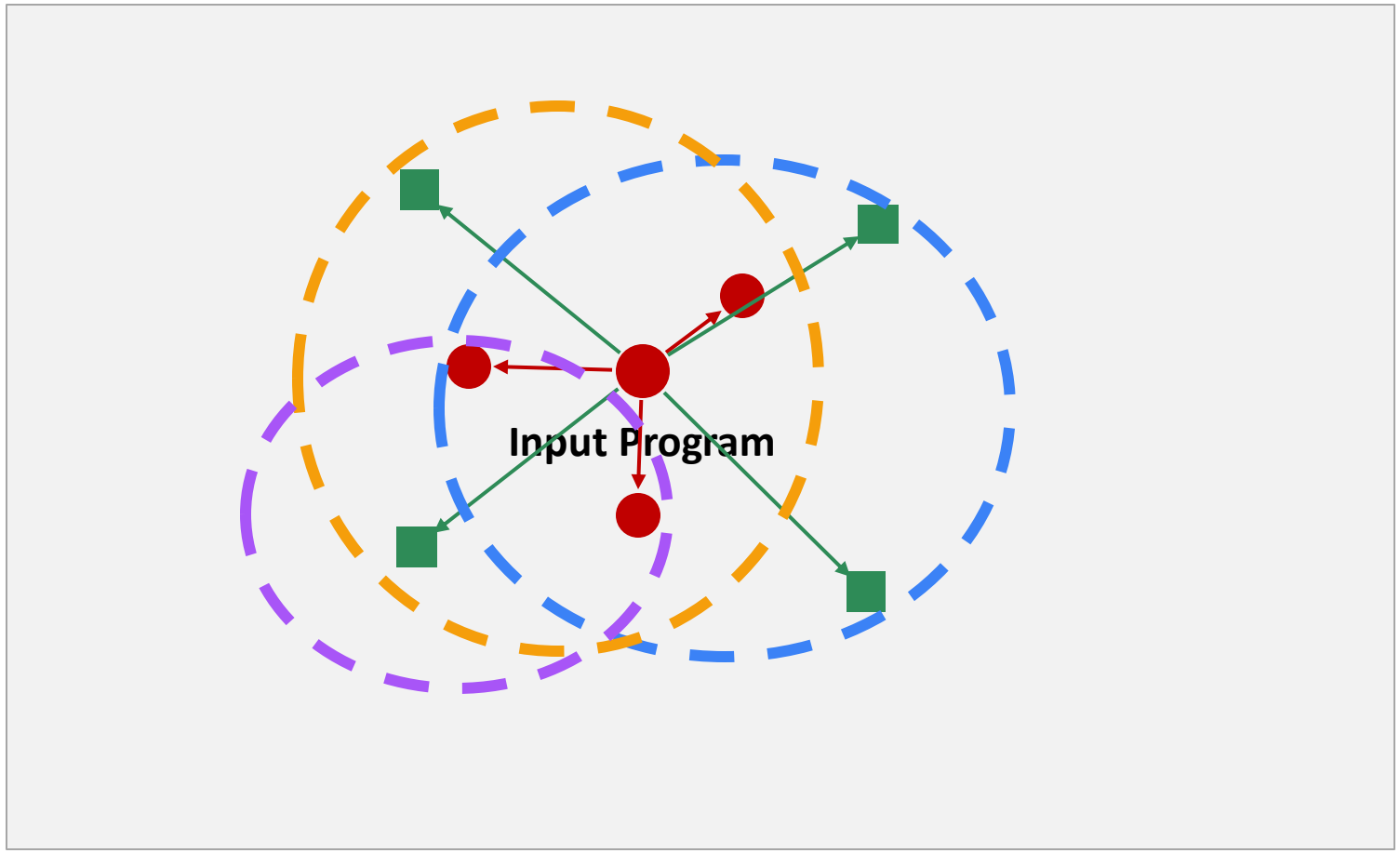
INSIGHT's Explanations: Separating Formulas



How to find a separating formula? Predicate-based search

1. Generate candidate predicates using internal core signals
2. Find a conjunction of predicates that leads to optimal separation

INSIGHT Step 2: Finding a separating formula

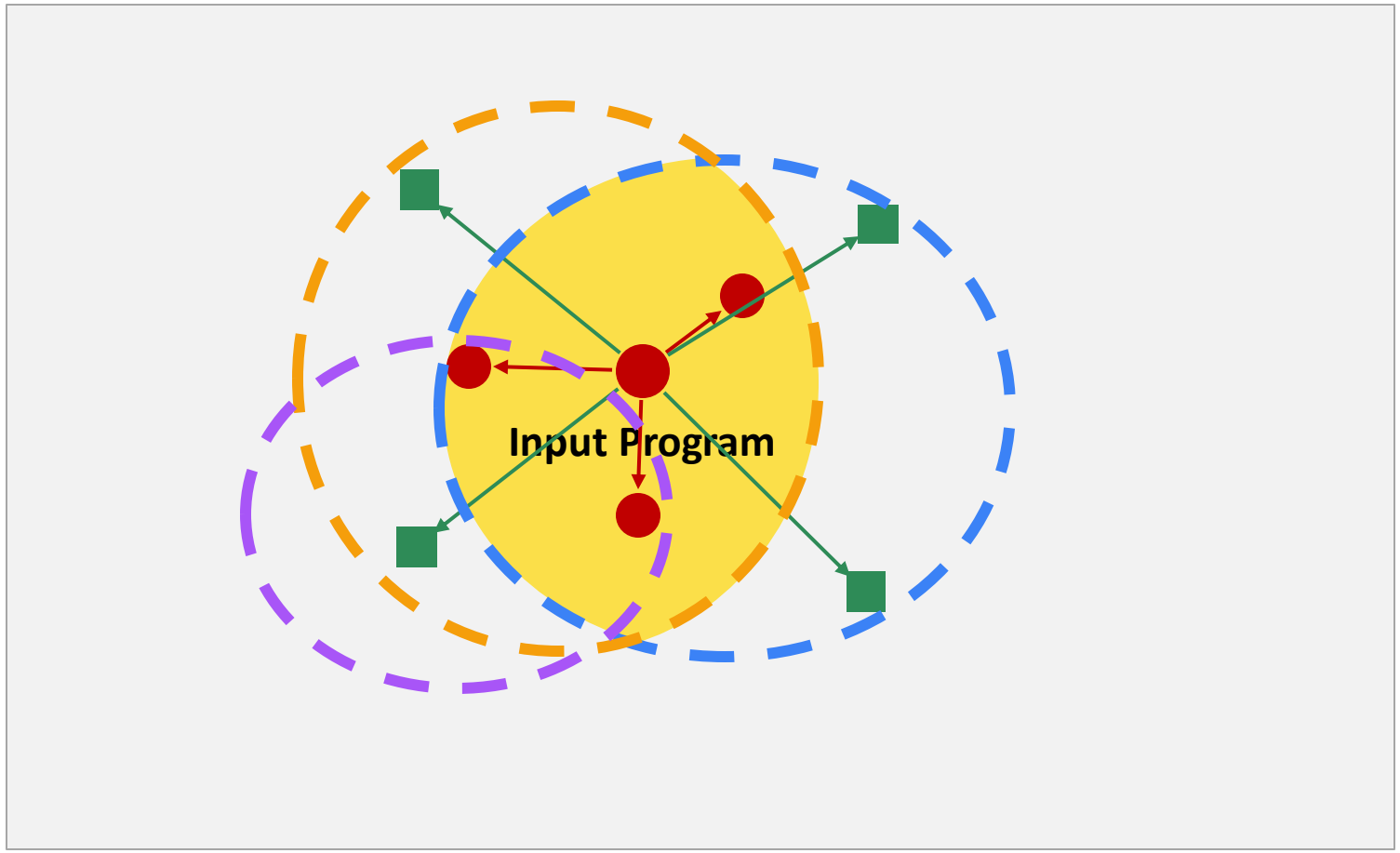


Predicates

- `p1: rs1_fwd == 1`
- `p2: exe.target== dec.source`
- `p3: dec.op == add`

For each predicate p compute cover-set: the programs for which p holds true

INSIGHT Step 2: Finding a separating formula

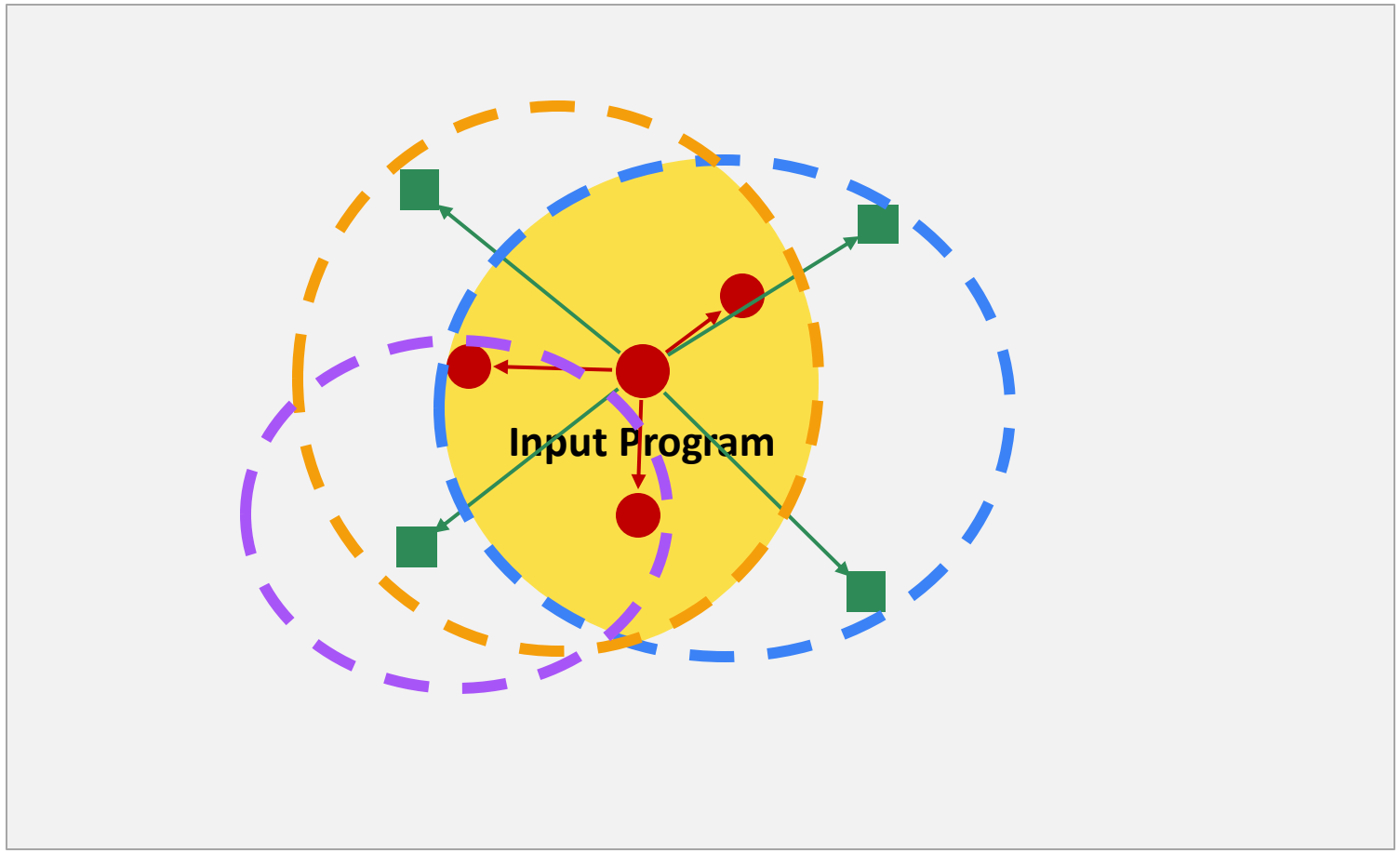


Predicates

- `p1: rs1_fwd == 1`
- `p2: exe.target== dec.source`
- `p3: dec.op == add`
- `p1 ∧ p2 (selected)`

Find combination of predicates such that intersection of their cover sets maximizes separation

INSIGHT Step 2: Finding a separating formula



Predicates

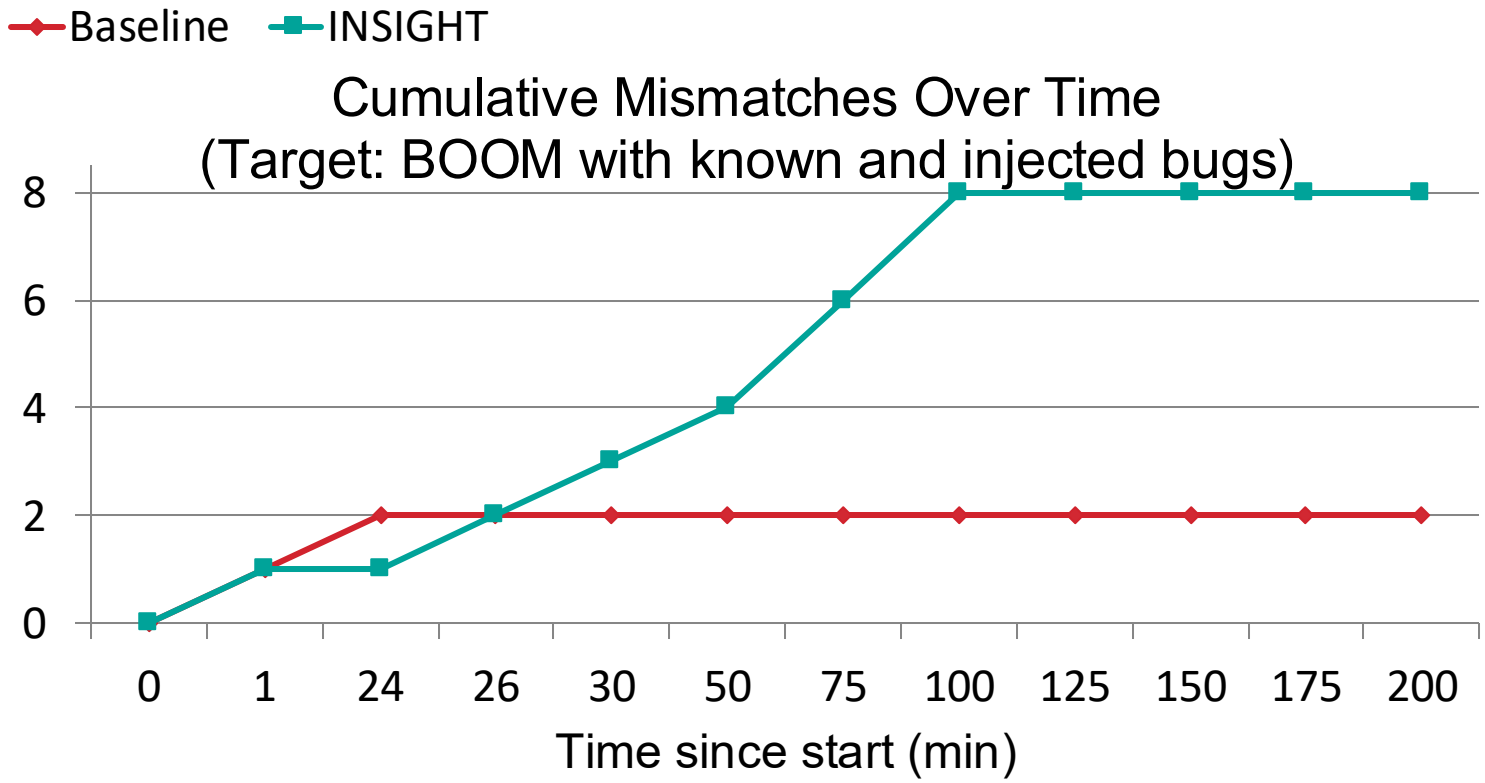
- `p1: rs1_fwd == 1`
- `p2: exe.target== dec.source`
- `p3: dec.op == add`
- `p1 ∧ p2 (selected)`

Final selection: $p_1 \wedge p_2$

```
rs1_forward == 1 && exe_stage.target == dec_stage.source
```

Insight Evaluation

Can INSIGHT help unblock model checkers? Yes, it finds more bugs



Unguided Run: **2** mismatches in **24 hours**

Insight-Guided Runs: **8** mismatches in **2 hours**

Evaluation: Unblocked model checkers and automated deduplication

INSIGHT unblocks model checking (BOOM):

Unguided: **2** mismatches/**24 hours**

Insight-Guided: **8 mismatches/2 hours**

INSIGHT deduplicates testcases (Kronos):

9000 programs reported by a fuzzer **into 10 distinct clusters**

Conclusion

1

INSIGHT can automatically synthesize machine-readable explanations for mismatch-triggering programs

2

INSIGHT unblocks model checking and allows for fuzzer testcase deduplication

Questions/Comments?
viniul@mit.edu

https://github.com/MATCHA-MIT/insight