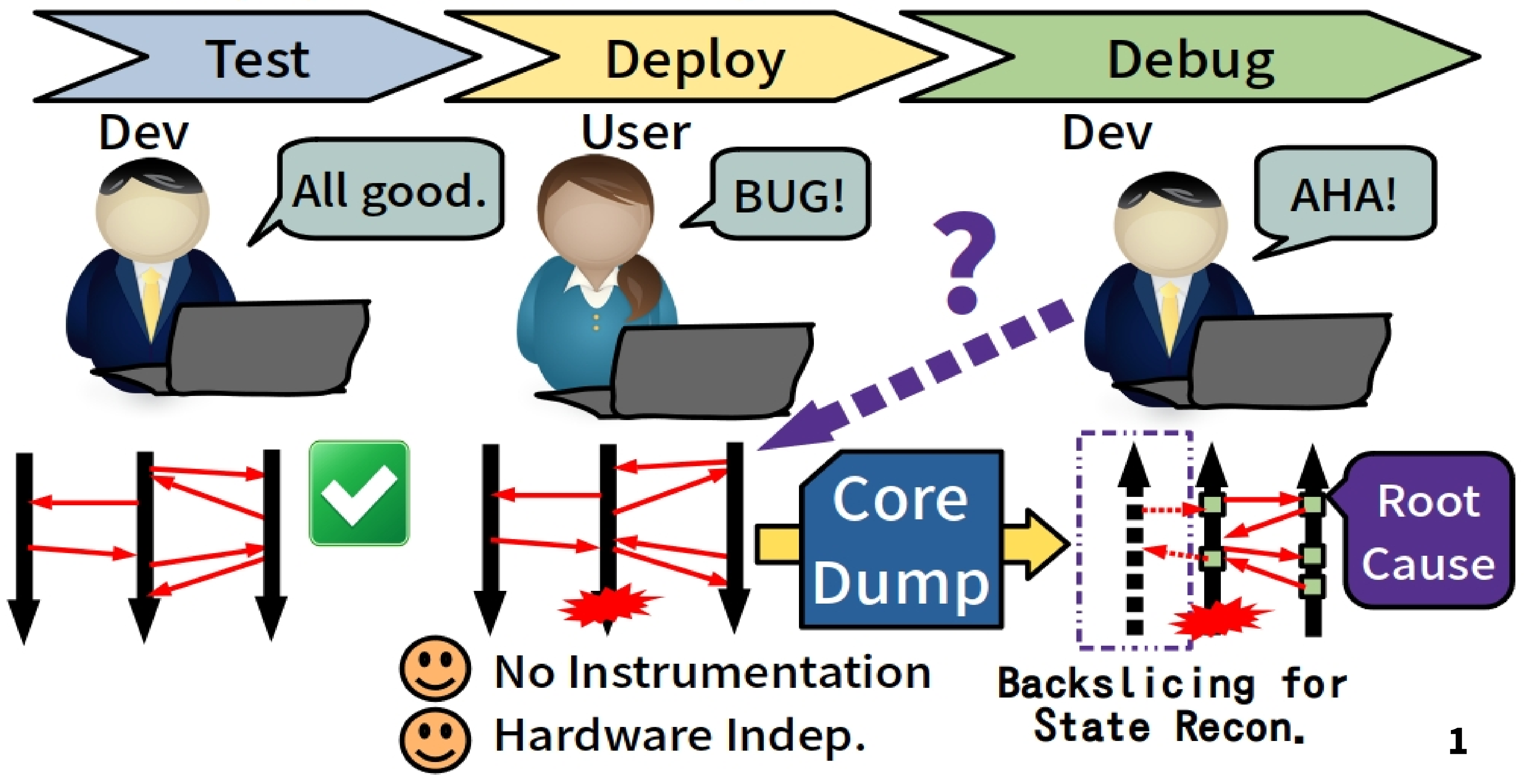
Static Analysis for Efficient Reverse-Debugging of Concurrency Bugs

P18

Shinji Hoshino, Yoshitaka Arahori, Katsuhiko Gondow (Tokyo Institute of Technology)

Overarching Goal

- Efficient, Software-Only Reverse-Debugging of Deployed Concurrent Programs



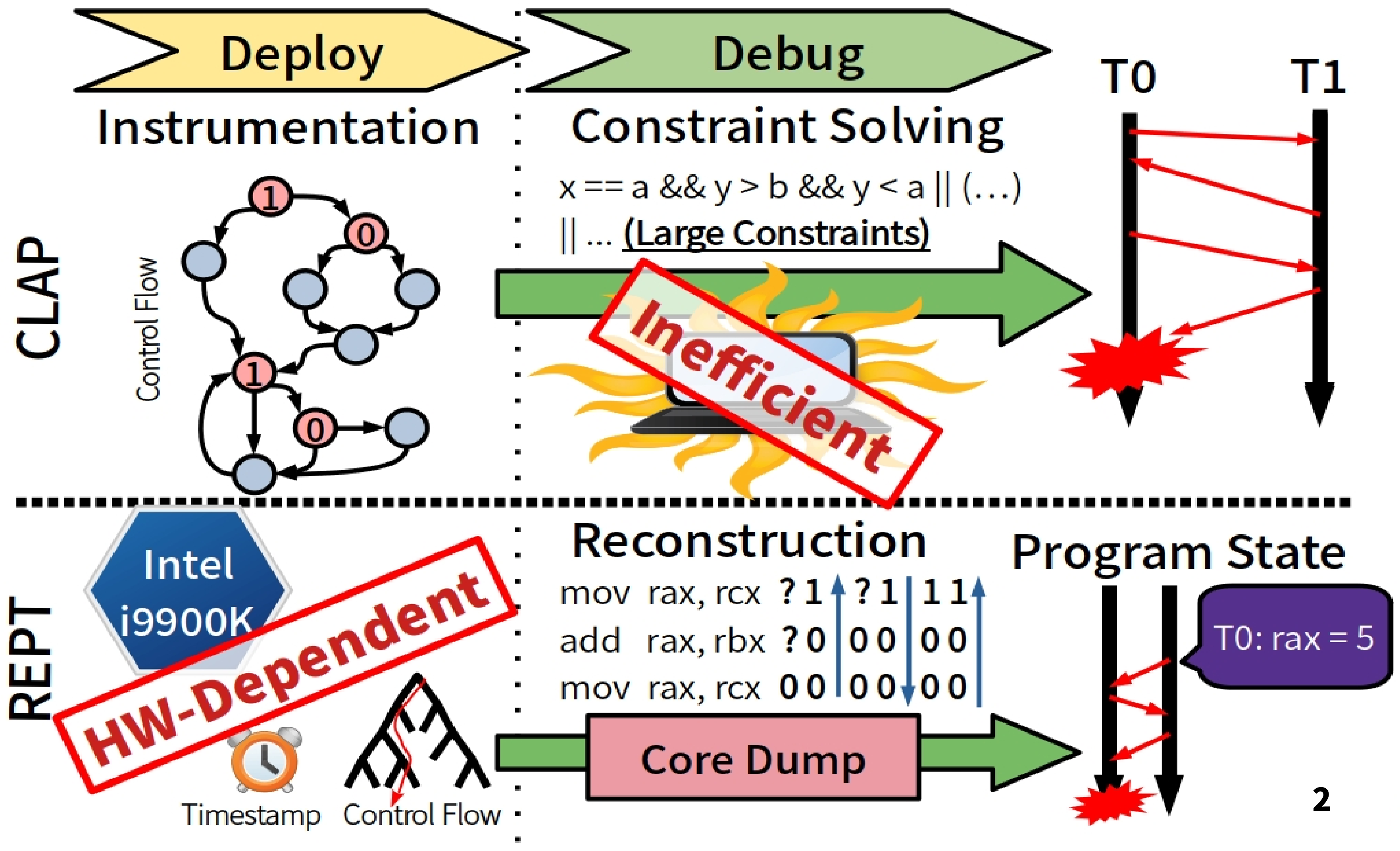
😊 No Instrumentation

😊 Hardware Indep.

Backslicing for State Recon.

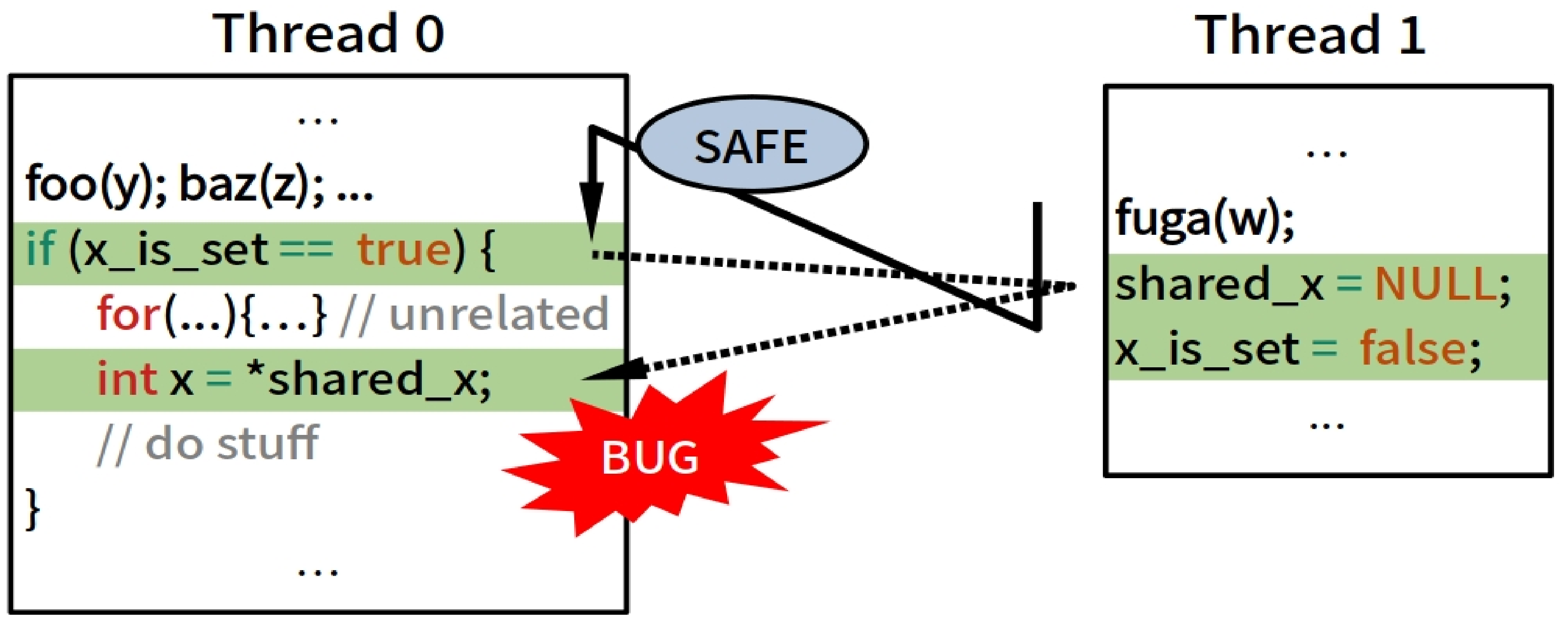
1

Problems w/ SOTA Reverse Debuggers



2

Motivating Example



Thread 0

```
...
foo(y); baz(z); ...
if (x_is_set == true) {
    for(...){...} // unrelated
    int x = *shared_x;
    // do stuff
}
...
```

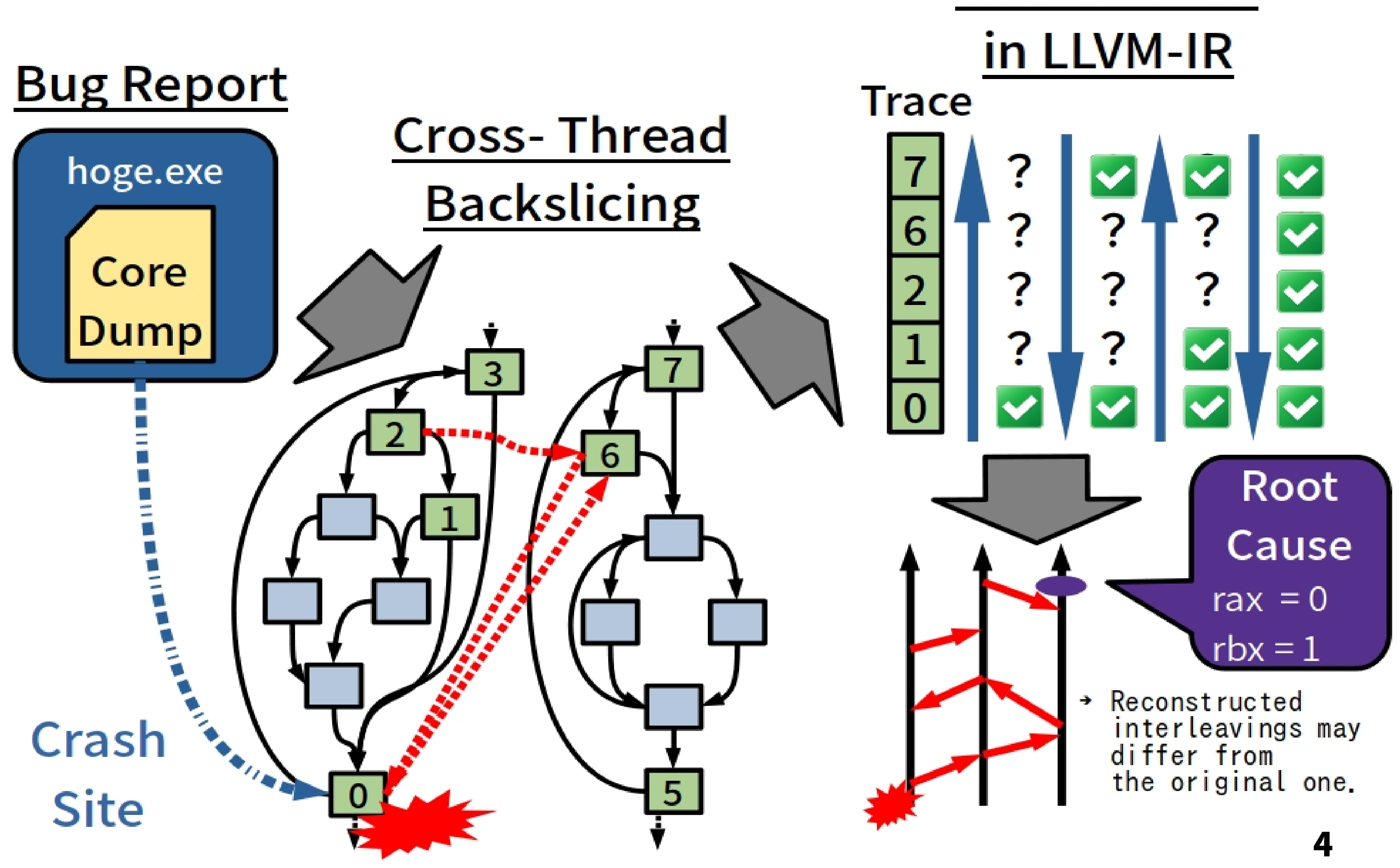
Thread 1

```
...
fuga(w);
shared_x = NULL;
x_is_set = false;
...
```

- Efficient & Software-only reverse debugging is possible with...
  I. Cross-thread backward slices
  II. Repeated state reconstruction
  ➔ All done in LLVM-IR

3

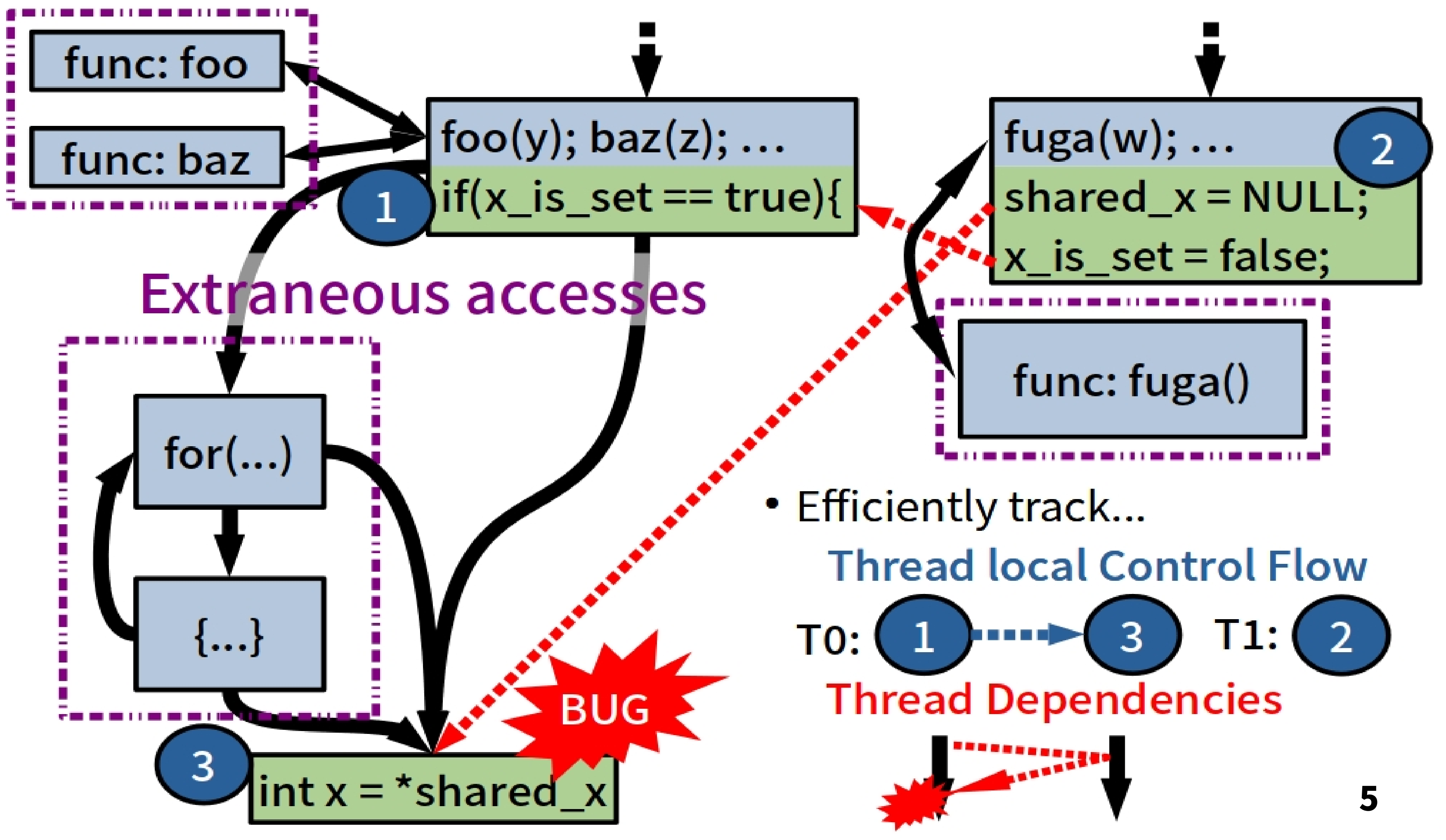
Our Approach



→ Reconstructed interleavings may differ from the original one.

4

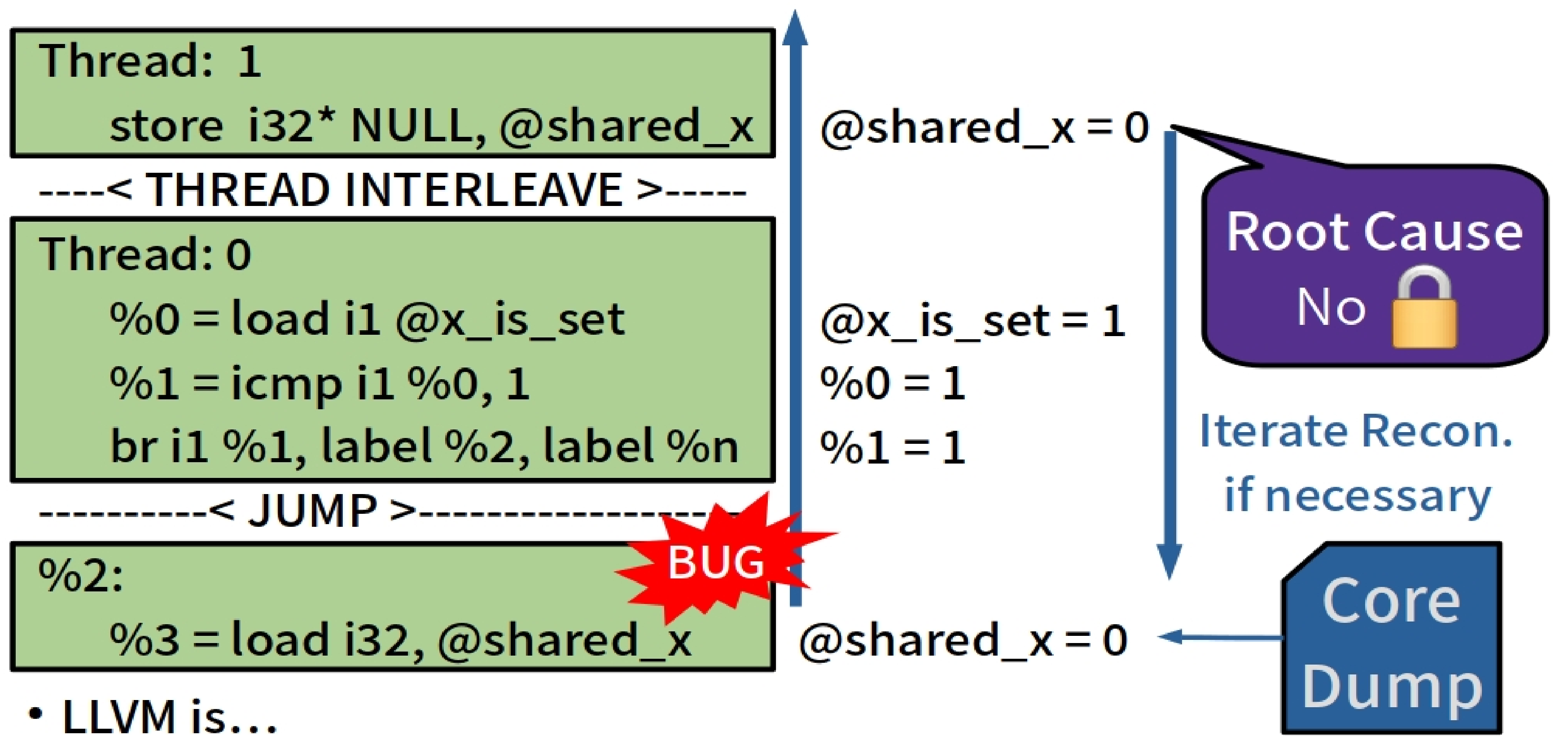
Cross-Thread Backslicing



- Efficiently track...
  Thread local Control Flow
  Thread Dependencies

5

Repeated State Recon. in LLVM-IR



```
Thread: 1
    store i32* NULL, @shared_x          @shared_x = 0
----< THREAD INTERLEAVE >-----
Thread: 0
    %0 = load i1 @x_is_set              @x_is_set = 1
    %1 = icmp i1 %0, 1                  %0 = 1
    br i1 %1, label %2, label %n        %1 = 1
----------< JUMP >-----------------
%2:
    %3 = load i32, @shared_x            @shared_x = 0
```

- LLVM is…
  - Higher level than assembly
  - Architecture Independent
  - Has properties for program analysis (SSA, type etc...)

6

Related Work and Our Vision

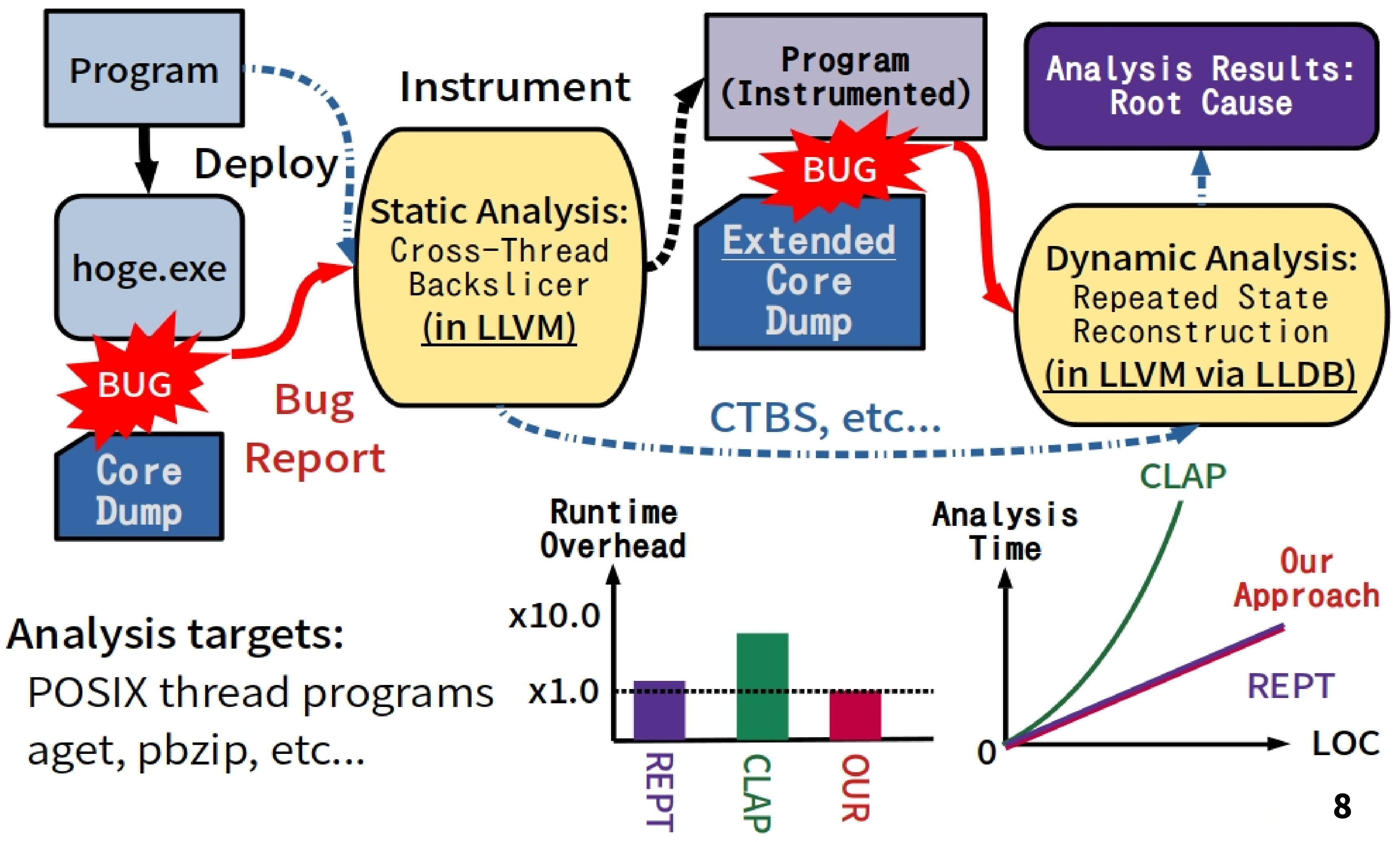
	Hardware Dep.	Intrusive	Runtime Overhead	Analysis Overhead	Arch. Dep.	State Recon. Prec.	State Recon. Recall
CLAP [0]	NO	YES	MED	VERY HIGH	NO	H	H
REPT [1]	YES	YES	VERY LOW	LOW	YES	H	L
Failure Sketch [2]	Y/N	YES	LOW	MED	Y/N	M	M
ConSeq [3]	NO	NO	MED	MED	YES	N/A	L
Our Approach	NO	NO	LOW	LOW	NO	H*	M

*debug execution

[0] CLAP: Recording Local Executions to Reproduce Concurrency Failures, PLDI'13
[1] REPT: Reverse Debugging of Failures in Deployed Software, OSDI'18
[2] Failure Sketching: A Technique for Automated Root Cause Diagnosisof In-Production Failures
[3] Conseq: Detecting Concurrency Bugs through Sequential Errors, ASPLOS'11

7

Exp. Design & Expected Results



Analysis targets:
POSIX thread programs
aget, pbzip, etc...

8