CPA-SymExec

Efficient Symbolic Execution in CPAchecker

Dirk Beyer and Thomas Lemberger

Overview

CPA-SymExec is a symbolic-execution engine for C programs, implemented in CPAchecker. It tackles the path-explosion problem of symbolic execution with counterexample-guided abstraction refinement (CEGAR). In the context of symbolic execution, it provides:

- Generation of executable test cases for condition coverage
- Concrete, symbolic and executable program traces
- Interactive, visual analysis reports based on HTML

For examples of these, have a look at the demo or the YouTube video.

Downloads

cpachecker.sosy-lab.org

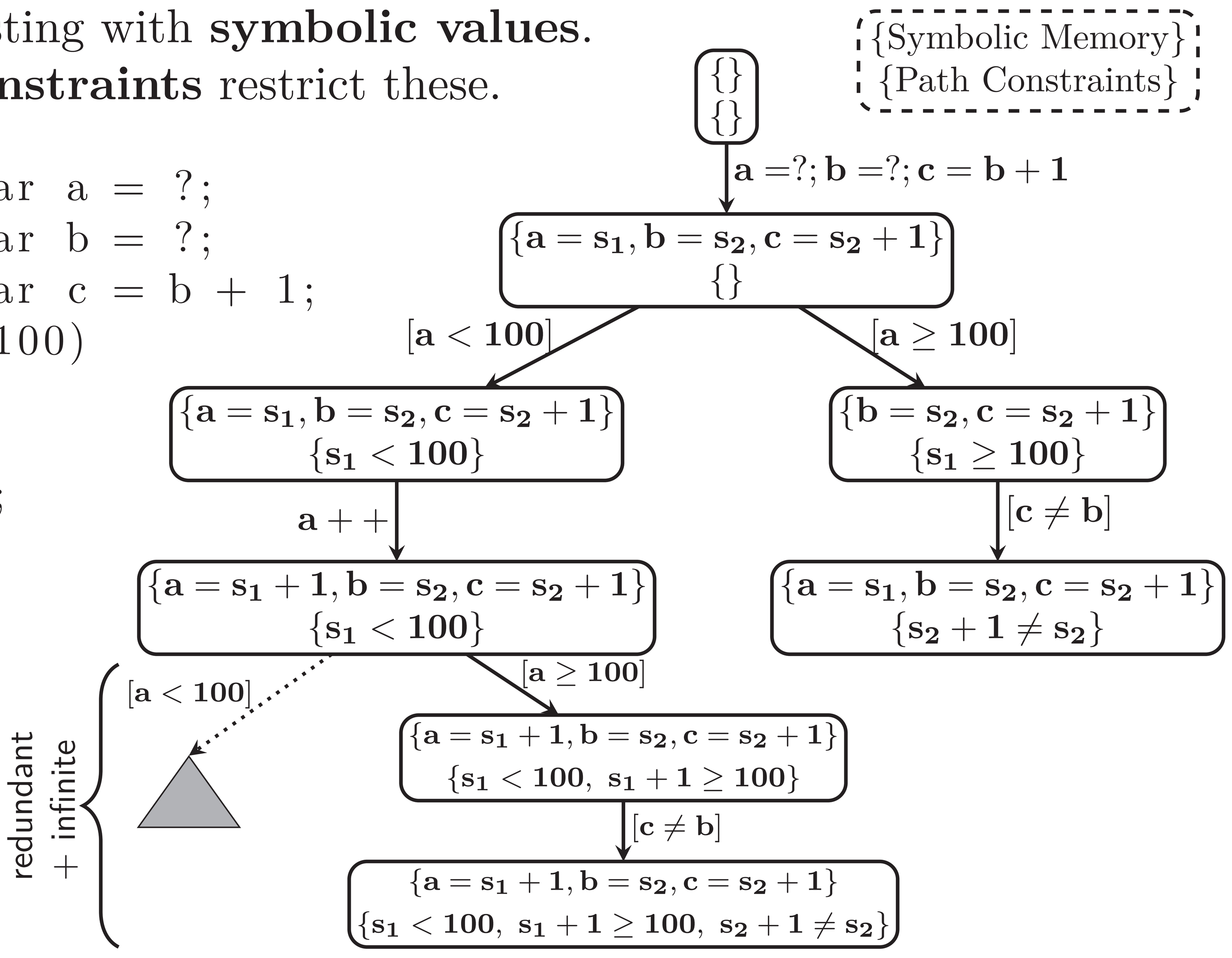
doi.org/10.5281/zenodo.1321181

youtu.be/qoBHtvPKtnw

Symbolic Execution [4]

- Idea: Testing with **symbolic values**.
- **Path constraints** restrict these.

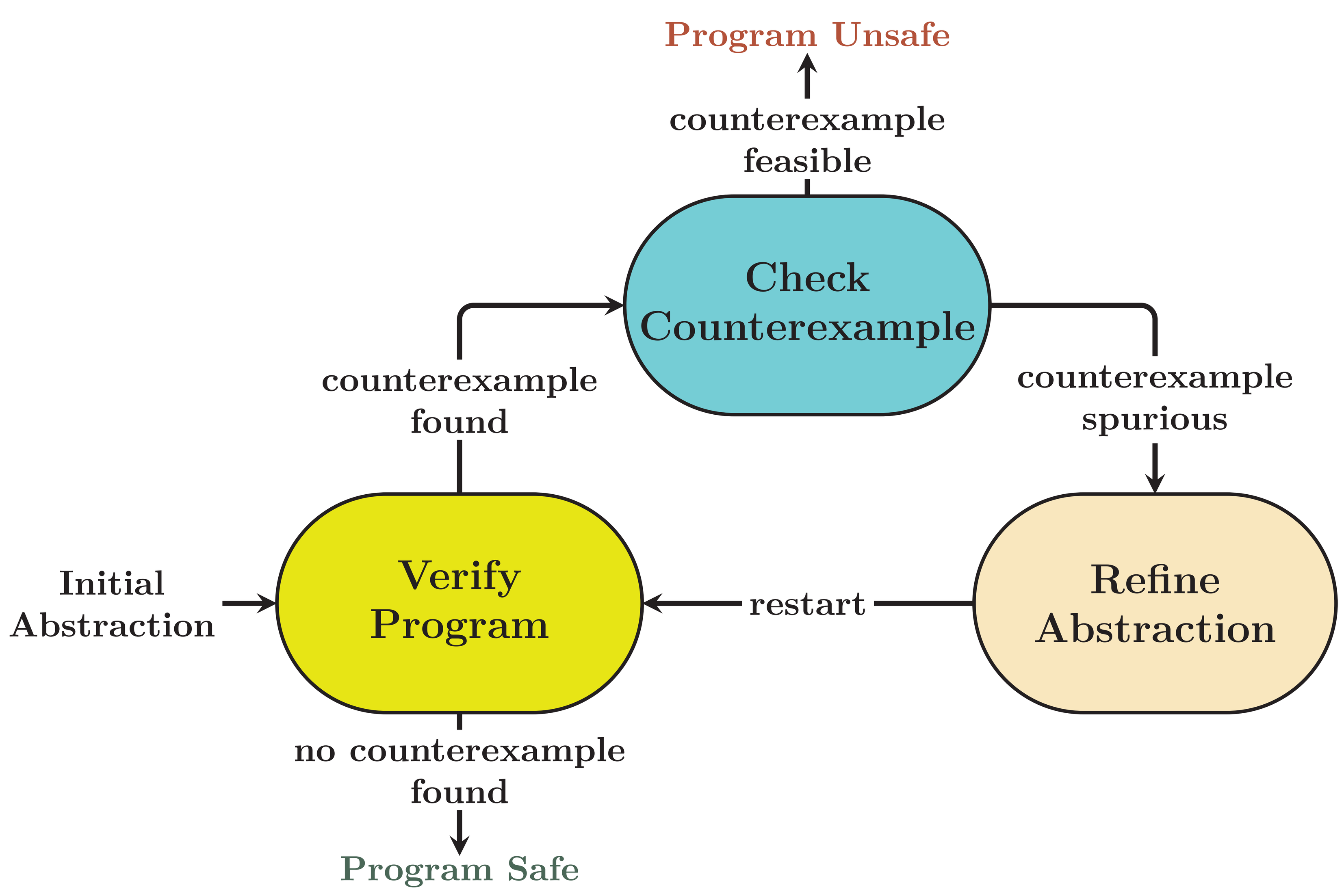
```
unsigned char a = ?;
unsigned char b = ?;
unsigned char c = b + 1;
while (a < 100)
    a++;
if (c == b)
    error();
```



⇒ Issue with scalability: **path explosion**.
Because of high precision, amount of states may grow exponentially and loops may be unrolled infinitely.

CEGAR [2]

Start with **coarse** abstraction. Refine based on spurious counterexamples.

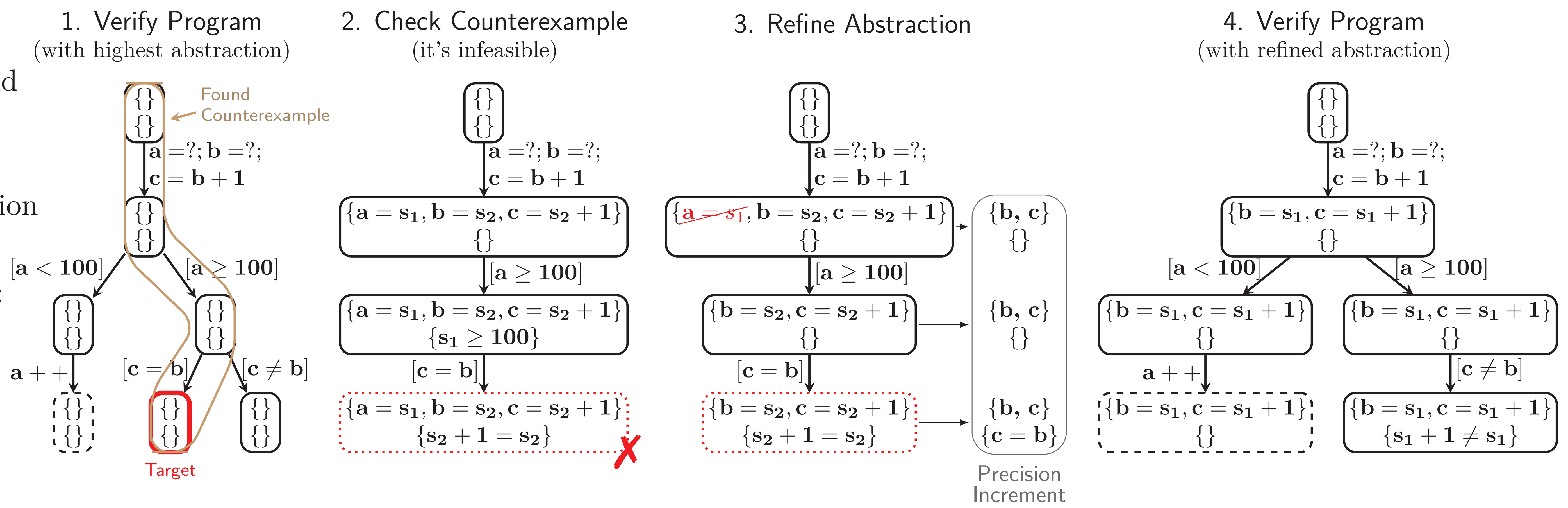


Symbolic Execution with CEGAR [1]

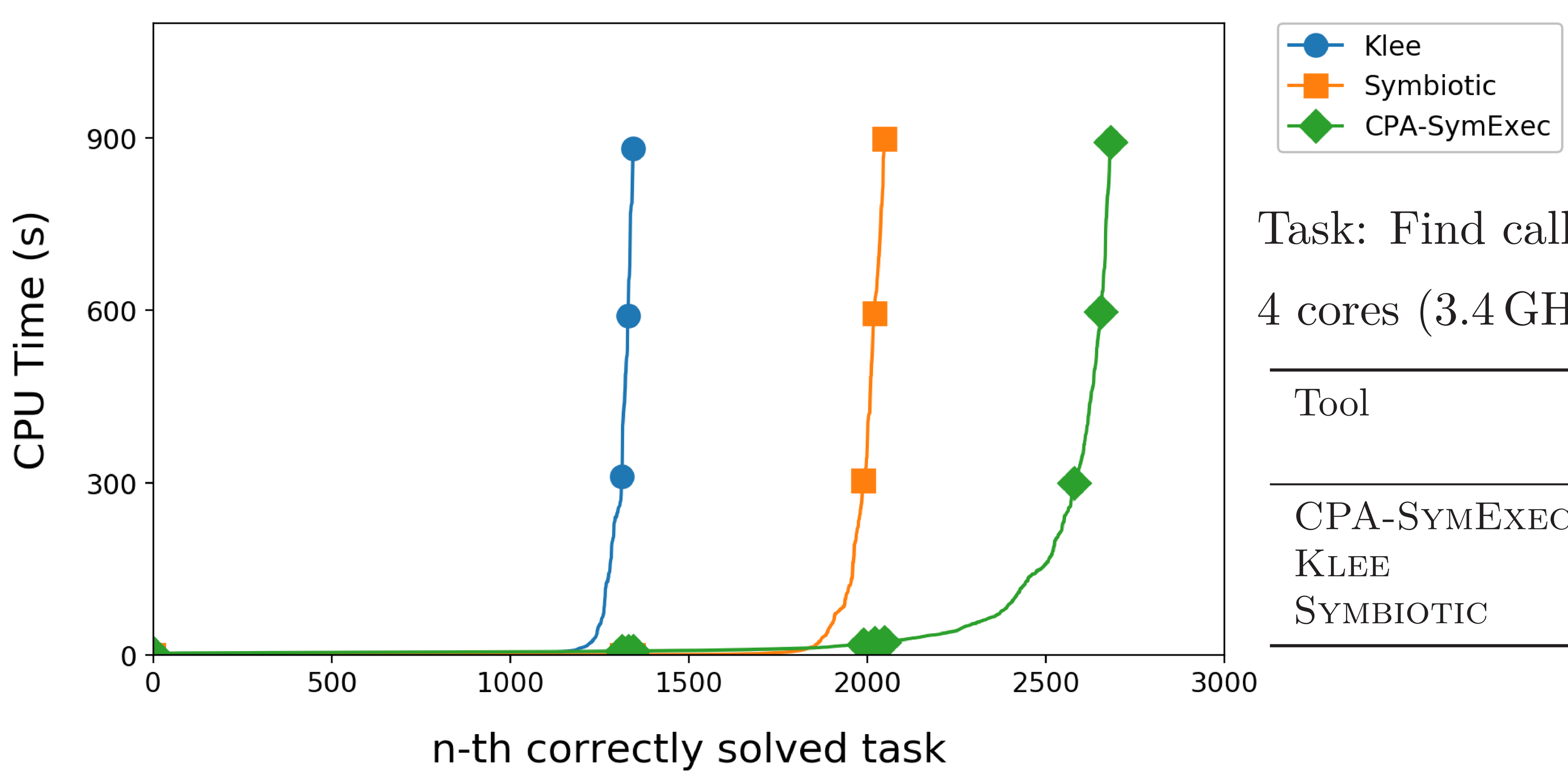
→ **Abstraction**: ("Precision")
Which symbolic memory and path constraints to track.

→ **Counterexample check**:
Traditional symbolic execution over found counterexample.

→ **Abstraction Refinement**:
Trial & Error based on Craig interpolation [3]:
"Information 'x' needed to show counterexample infeasible?" ⇒ track 'x'



Experimental Results



Task: Find calls to error function.

4 cores (3.4 GHz), 900 s CPU time, 15 GB memory.

Tool	correct TRUE	correct FALSE	incorrect TRUE	incorrect FALSE
CPA-SymExec	**2137**	545	**0**	22
Klee	446	**899**	6	33
Symbiotic	1201	848	3	**9**

References

[1] D. Beyer and T. Lemberger. Symbolic execution with CEGAR. In *Proc. ISoLA*, LNCS 9952, pages 195–211. Springer, 2016.

[2] E. M. Clarke, O. Grumberg, S. Jha, Y. Lu, and H. Veith. Counterexample-guided abstraction refinement for symbolic model checking. *J. ACM*, 50(5):752–794, 2003.

[3] W. Craig. Linear reasoning. A new form of the Herbrand-Gentzen theorem. *J. Symb. Log.*, 22(3):250–268, 1957.

[4] J. C. King. Symbolic execution and program testing. *Commun. ACM*, 19(7):385–394, 1976.