Static Analysis: Automated Bug Hunting and Beyond

Julian Erhard Michael Schwarz {julian.erhard, m.schwarz}@tum.de

Chair for Formal Languages, Compiler Construction, Software Construction Department of Informatics, Technical University of Munich

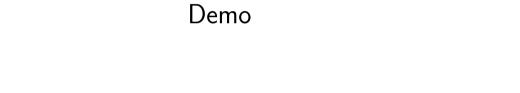
Summer Term 2023





Writing programs is hard.

Writing correct programs is very hard.



Abstract Interpretation

- ▶ Widely used both in Academia & Industry
- ► Can scale to huge industry-scale codebases
- ► The technique covered in Program Optimization Course (IN2053)

GOBLINT

- Analysis of multi-threaded, real-world C
- Efficient solvers for computation of fixpoints
- ▶ Winner of race-detection category at Software Verification Competition 2023
- ▶ https://goblint.in.tum.de

C

Static Analysis: Automated Bug Hunting and Beyond

Topics

- ► **Termination** analysis
 - ► Loops & recursion as sources of non-termination
 - ► Loops: Introduce ghost variables (c.f. ranking functions)
 - ► Recursion: Check abstract call graph for cycles

Topics

- ► **Termination** analysis
 - ► Loops & recursion as sources of non-termination
 - ► Loops: Introduce ghost variables (c.f. ranking functions)
 - Recursion: Check abstract call graph for cycles
- ► Analyzing **C11** code: C11 finally gaining traction
 - How can the analysis profit from new features such as thread_local variables?
 - New threading library with support for different weak memory models

Benefits

- Prevent the next starship from exploding (maybe)
- Deepen your understanding of
 - ► The Semantics of C and typical programming errors
 - Static Analysis by Abstract Interpretation
- Level up your functional programming skills
- Become connected to the research we do day-to-day

Requirements

- ► Program Optimization Course (IN2053)
- ► Knowledge of a functional programming language (we use OCaml)
- ▶ Be in your Master's (Advanced Bachelor's students welcome)

Questions?



O/goblint/analyzer