Proving Soundness of Top-Down Solvers with Widening and Narrowing

Alexandra Graß
alexandra.grass@tum.de

Supervisors: Jasmin Blanchette and Helmut Seidl

Collaborators: Sarah Tilscher (ConVeY) and Yannick Stade

ConVeY

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Motivation

Static analysis serves as justification for complex code optimization and can certify the absence of runtime errors. For example, the if block below can obviously be removed.

```
1 i = 0; sum = 0;
2 while (i < 100) {
3   sum += i;
4   i++;
5 }
6
7 if (sum < 0) {
8   ( Big block of code
9 }
```

Obviously? Static analyzers are complex and highly optimized pieces of code themselves. Are their claims sound?

⇒ To increase the trust in static analysis, we provide a machine-checked verification of the soundness of the top-down solver TD with warrowing, using Isabelle/HOL.

Generic Solvers

Using abstract interpretation [1], generic solvers reduce the problem statement to an abstract domain of values at (possibly abstract) program points. E. g., to analyze the value of `i` at the loop head vs its body, it suffices to consider

$$x = (\texttt{if } x < 100 \texttt{ then } y \texttt{ else } 100) \qquad y = x + 1$$

where the program state is abstracted to the value of `i`. An answer to the problem statement is obtained by computing a solution to the equations system. In contrast, non-generic solvers usually follow a syntax-based approach.

The Top-Down Solver TD

The solver TD is a local generic solver. It is implemented by three mutually recursive functions:

- `query` obtains the value of an unknown's right-hand side
- `iterate` implements the fixpoint iteration necessary to solve systems with cyclic dependencies
- `eval` traverses the rhs and recursively queries unknowns

The vanilla version of the TD is sound (Stade et al. [2]).

Acceleration by Widening and Narrowing

In a fixpoint iteration, let a be the previous and b the latest value. To accelerate the fixpoint iteration, results are deliberately overapproximated by widening (∇). Following narrowing (Δ) applications can recover lost precision.

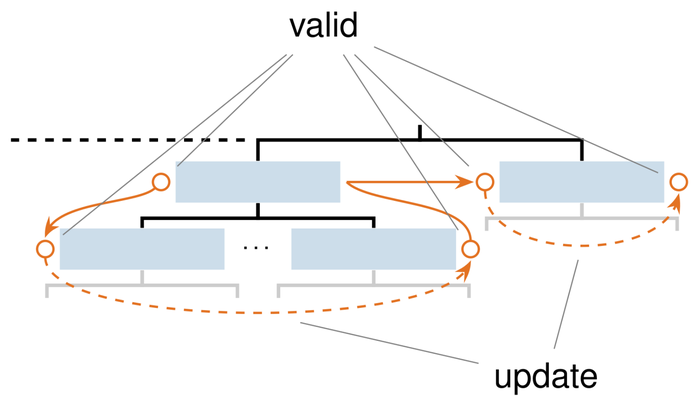
$$a, b \leq a \nabla b$$
$$a \sqcap b \leq a \Delta b \leq a$$

The warrowing operator ($\boxtriangledown$) dynamically chooses the appropriate operator.

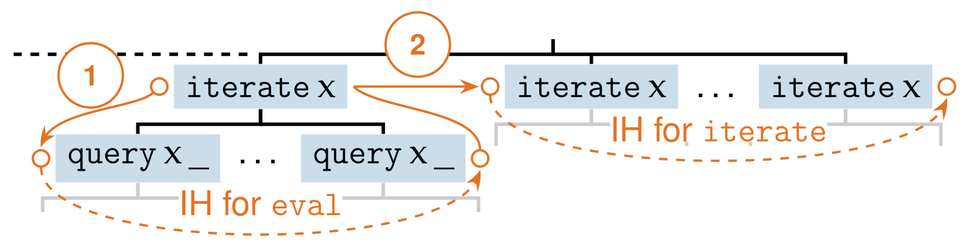
$$a \boxtriangledown b := \begin{cases} a \Delta b & \text{if } b \leq a \\ a \nabla b & \text{otherwise} \end{cases}$$

Proof of Soundness

The proof is done by a computation induction, closely following Stade et al. [2]. Show that (a) the computation preserves the state invariant **valid** and (b) recursive calls adapt the solver state according to the **update** relation.



Warrowing is applied after `eval` returns the latest value of a right-hand side. Verify that a warrowing application also preserves the consistency of the solver state (2nd point):



We identify two crucial properties of warrowing:

- The 2nd operand is a lower bound: $b \leq a \boxtriangledown b$
- Fixpoint implies narrowing: $a \boxtriangledown b = a \implies b \leq a$

This completes the proof of soundness. □

Current, Future and Other Work

- Termination proof for TD_{wn}
- Verification of TD with side effects
- CvxLean and/or lean-egg

References

[1] P. Cousot et al. "Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints". In: Conference Record of the Fourth ACM Symposium on Principles of Programming Languages, Los Angeles, California, USA, January 1977. Ed. by R. M. Graham et al. ACM, 1977, pp. 238–252.

[2] Y. Stade et al. "The Top-Down Solver Verified: Building Confidence in Static Analyzers". In: Computer Aided Verification - 36th International Conference, CAV 2024, Montreal, QC, Canada, July 24-27, 2024, Proceedings, Part I. Ed. by A. Gurfinkel et al. Vol. 14681. Lecture Notes in Computer Science. Springer, 2024, pp. 303–324.