LIV

Marian Lingsch-Rosenfeld
Dirk Beyer, Martin Spiessl

April 7, 2024
LMU Munich, SoSy-Lab

Establish Full Proofs

```
1 int x = 0;
2 int sum = 0 ;
3 //@ loop invariant I;
4 while (x<10) {
5   x++;
6   sum+=x;
7 }
8 assert(sum<=55);
```

$$\dfrac{\{P\}\, s_0\, \{R\} \qquad \dfrac{R \Rightarrow I \qquad \{I \wedge C\}\, B\, \{I\} \qquad I \wedge \neg C \Rightarrow Q}{\{R\}\ \texttt{while}\ C\ \texttt{do}\ B\ \{Q\}}\ \text{while}}{\{P\}\, s_0\ \texttt{while}\ C\ \texttt{do}\ B\, \{Q\}}\ \text{comp}$$

Proof Obligations:

- $\{P\}\, s_0\, \{I\}$
- $\{I \wedge C\}\, B\, \{I\}$
- $\{I \wedge \neg C\}\ \{Q\}$

From Proof Obligations to Straight-Line Programs

Proof Obligations:

- $\{P\}\ s_0\ \{I\}$ (Reachability)
- $\{I \wedge C\}\ B\ \{I\}$ (Inductiveness)
- $\{I \wedge \neg C\}\ \{Q\}$ (Safety)

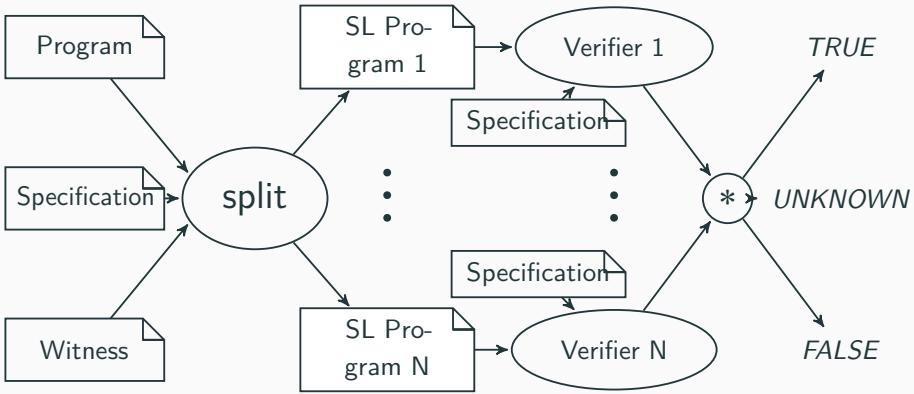
Straight-Line Programs:

```
int x = 0;
int sum = 0;
assert(I);
```

```
int x = nondet();
int sum = nondet();
assume(I && C);
x++;
sum += x;
assert(I);
```

```
int x = nondet();
int sum = nondet();
assume(I && !C);
assert(Q);
```

Workflow of LIV



- can use any off-the-shelf verifier from SV-COMP as backend

Summary

- LIV: a correctness-witness validator
- splits a program into multiple straight-line programs
- delegates validation to verifiers
- allows insights into why a proof fails
- complements existing validators

sosy-lab.org/research/liv
Supplementary Webpage