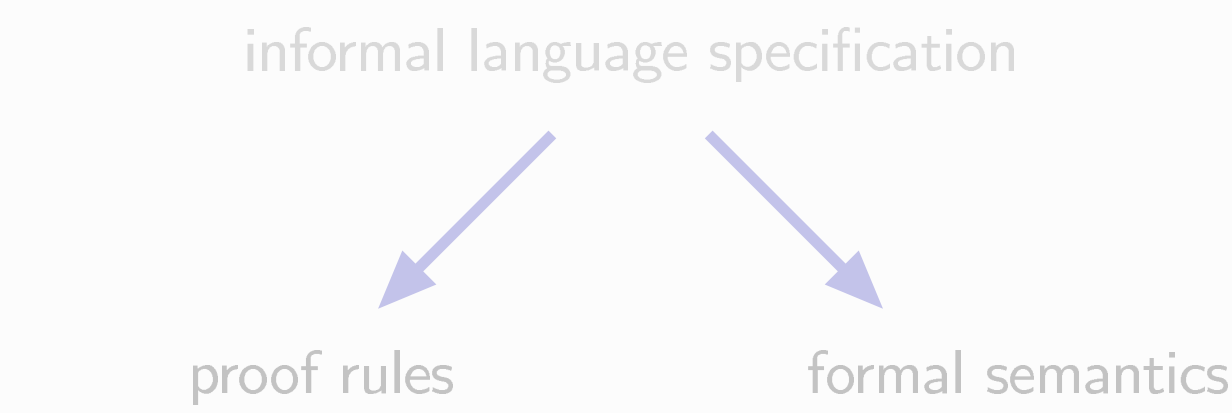
Verification Calculus Soundness

A fundamental problem!



Verification Calculus Soundness

A fundamental problem!

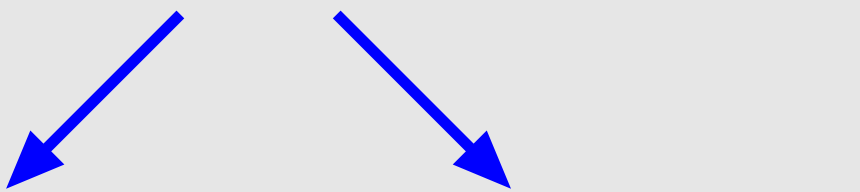
informal language specification

proof rules

formal semantics

Verification Calculus Soundness

A fundamental problem!



Validating Soundness of Proof Rules

Bootstrapping

Validate a core set of rules,
generate and prove verification conditions for additional rules

Cross-verification

- against the BALI calculus for Java formalized in Isabelle/HOL
  [D. von Oheimb, T. Nipkow]
- against the Java semantics in the MAUDE system
  [J. Meseguer]

Tests

Using the compiler test suite Jacks

Validating Soundness of Proof Rules

Bootstrapping

Validate a core set of rules,
generate and prove verification conditions for additional rules

Cross-verification

- against the BALI calculus for Java formalized in Isabelle/HOL
  [D. von Oheimb, T. Nipkow]
- against the Java semantics in the MAUDE system
  [J. Meseguer]

Tests

Using the compiler test suite Jacks

Validating Soundness of Proof Rules

Bootstrapping

Validate a core set of rules,
generate and prove verification conditions for additional rules

Cross-verification

- against the BALI calculus for Java formalized in Isabelle/HOL
  [D. von Oheimb, T. Nipkow]
- against the Java semantics in the MAUDE system
  [J. Meseguer]

Tests

Using the compiler test suite Jacks

From the Java Language Specification

`PostIncrementExpression:`
`PostfixExpression ++`

At run time, if evaluation [...] completes abruptly, then the postfix increment expression completes abruptly and no incrementation occurs.
Otherwise, the value 1 is added to the value of the variable and the sum is stored back into the variable. Before the addition, binary numeric promotion is performed on the value [...]
The value of the postfix increment expression is the value of the variable before the new value is stored.

From the Java Language Specification

`PostIncrementExpression:`
`PostfixExpression ++`

At run time, if evaluation [...] completes abruptly, then the postfix increment expression completes abruptly and no incrementation occurs.
Otherwise, the value 1 is added to the value of the variable and the sum is stored back into the variable. Before the addition, binary numeric promotion is performed on the value [...]
The value of the postfix increment expression is the value of the variable before the new value is stored.

Rule for Postfix Increment

Intuitive rule (not correct!)

$$\frac{==>\langle\pi\ \texttt{x=y;}\ \texttt{y=y+1;}\ \omega\rangle\,\phi}{==>\langle\pi\ \texttt{x=y++;}\ \omega\rangle\,\phi}$$

But ...

$x = 5 ==> \langle\texttt{x=x++;}\rangle(x = 6)$ INVALID

Correct rule

$$\frac{==>\langle\pi\ \texttt{v=y;}\ \texttt{y=y+1;}\ \texttt{x=v;}\ \omega\rangle\,\phi}{==>\langle\pi\ \texttt{x=y++;}\ \omega\rangle\,\phi}$$

Rule for Postfix Increment

Intuitive rule (not correct!)

$$\frac{==> \langle \pi \ \texttt{x=y;} \ \texttt{y=y+1;} \ \omega \rangle \, \phi}{==> \langle \pi \ \texttt{x=y++;} \ \omega \rangle \, \phi}$$

But ...

$$x = 5 ==> \langle \texttt{x=x++;} \rangle (x = 6) \qquad \text{INVALID}$$

Correct rule

$$\frac{==> \langle \pi \ \texttt{v=y;} \ \texttt{y=y+1;} \ \texttt{x=v;} \ \omega \rangle \, \phi}{==> \langle \pi \ \texttt{x=y++;} \ \omega \rangle \, \phi}$$

Rule for Postfix Increment

Intuitive rule (not correct!)

$$\frac{==> \langle \pi \ \texttt{x=y;} \ \texttt{y=y+1;} \ \omega \rangle \, \phi}{==> \langle \pi \ \texttt{x=y++;} \ \omega \rangle \, \phi}$$

But ...

$x = 5 ==> \langle \texttt{x=x++;} \rangle (x = 6)$ INVALID

Correct rule

$$\frac{==> \langle \pi \ \texttt{v=y;} \ \texttt{y=y+1;} \ \texttt{x=v;} \ \omega \rangle \, \phi}{==> \langle \pi \ \texttt{x=y++;} \ \omega \rangle \, \phi}$$

From the Jacks Conformance Test Suite

```
class T1241r1a {
    final int i=1; static final int j=1;
    static { }
}

class T1241r1b {
     /*@ public normal_behavior
       @ ensures \result == 7;  @ */
    public static int main() {
      int s = 0; T1241r1a a = null;
      s = s + a.j;
      try {s = s + a.i;}
      catch (Exception e) {
        s = s + 2; a = new T1241r1a();
        s = s + a.i + 3;  }
      return s; }
}
```