Formal Methods for Java
Lecture 21: Proofs in Jahob

Jochen Hoenicke
Software Engineering
Albert-Ludwigs-University Freiburg

Jan 18, 2012
Goal:
- finds bugs at compile-time,
- proves that there is no violation.

Static Checking:
- e.g. Jahob and ESC/Java
- fully automatic (after annotation)
- can only verify simple properties

Theorem Proving:
- e.g. KeY Prover
- Needs lot of manual interaction
- complete calculus, can verify any property.
The Jahob Proof Language

Goals

- Improve the strength of the provable properties.
- Still fully automatic (after annotation).
- Have intermediate proof steps in annotation.

Paper:

We already know one command

\[
\text{note } \ell : F
\]

which abbreviates

\[
\text{assert } \ell : F; \text{ assume } \ell : F
\]

- $\ell$ is a label (or name) for the formula $F$
- When $F$ cannot be proven Jahob tells that the check for $\ell$ failed.
- $\ell$ can also be used to tell the Jahob which formulas are relevant:

\[
\text{assert } G \text{ from } \ell
\]

This rule is correct, i.e., $wp(\text{note } F, H) \rightarrow H$:

\[
wp(\text{note } F, H) \leftrightarrow F \land (F \rightarrow H) \\
\leftrightarrow F \land H \\
\rightarrow H
\]
Proving implications

To prove an implication $F \rightarrow G$, the syntax is

```
assuming F
:
:
note G
```

This is an abbreviation for

```
( assume F
:
assert G
assume false

□

assume F → G
)
```

• : stands for arbitrary proof statements
The implication rule is correct, provided the proof statements used in between are correct.

\[
wp\left(\text{assume } F; p; \text{ assert } G; \text{ assume false } \square \text{ assume } F \rightarrow G, H\right)
\equiv (F \rightarrow wp(p, G)) \land ((F \rightarrow G) \rightarrow H)
\rightarrow [\text{assuming that proof statements } p \text{ are correct}]
(F \rightarrow G) \land ((F \rightarrow G) \rightarrow H)
\rightarrow H
\]
Case Splits

One can split cases, e. g.

\[ \text{cases } x \geq 0, x < 0 \text{ for } abs(x) \geq 0 \]

is an abbreviation for

\begin{align*}
\text{assert } F_1 & \lor \cdots \lor F_n; \\
\text{assert } F_1 & \rightarrow G; \\
\text{assert } F_n & \rightarrow G; \\
\text{assume } G \\
\end{align*}

\begin{itemize}
  \item Proof that \( F_1, \ldots, F_n \) are all possible cases.
  \item Proof for each case \( G \) separately.
  \item Assume \( G \) holds.
\end{itemize}
Proving Universal Quantifiers

To prove a universal quantified formula the syntax is

```
pickAny x
:
note F
```

This is an abbreviation for

```
( havoc x
:
assert F[x]
assume false

assume ∀x. F[x]
)
```
The inverse operation removes universal quantifiers:

\[ \text{instantiate } \forall x. F[x] \text{ with } t \]

This is an abbreviation for

\[ \text{assert } \forall x. F[x] \]
\[ \text{assume } F[t] \]
To prove an existential quantified formula the syntax is

\[ \text{witness } t \text{ for } \exists x. F[x] \]

This is an abbreviation for

\[ \text{assert } F[t] \]
\[ \text{assume } \exists x. F[x] \]
Removing Existential Quantifiers

The syntax is

\[
\text{pickWitness } x \text{ for } F[x]
\]

: where \( x \) does not occur in \( G \)

\text{note } G

This is an abbreviation for

\[
( \text{assert } \exists x. F[x] \\
\text{havoc } x \\
\text{assume } F[x] \\
: \\
\text{assert } G \\
\text{assume false} \\
\widehat{\quad} \\
\text{assume } G \\
)
\]