CS4004/CS4504: FORMAL VERIFICATION

Lecture 13: Hoare Logic

Vasileios Koutavas

School of Computer Science and Statistics
Trinity College Dublin

Goal: Software without bugs.

We have seen:

→ how to reason about mathematical properties in First-Order Logic (FOL)

Goal: Software without bugs.

We have seen:

- → how to reason about mathematical properties in First-Order Logic (FOL)

From now on:

- → how to reason about simple **imperative programs**
  - → FOL for specifications
- → how to reason about properties (e.g., termination) of recursive functional programs using:
  - → FOL natural deduction proofs
  - → Well-founded induction
  - → Case analysis on data types (natural numers, lists, etc.)
  - → variants (when reasoning about termination)

```
y := 1;
z := 0;
while (z != x) {
  z := z + 1;
  y := y * z;
}
```

The goal is to write specifications such as:

- “Program *P* computes a number *y* whose square is less than the input *x*”
- “Program *P* is a program such that at the end of *P*, array of integers *a* contains numbers in increasing order”

FLOYD-HOARE LOGIC [1969]

The goal is to write specifications such as:

- "Program P computes a number y whose square is less than the input x"
- "Program P is a program such that at the end of P, array of integers a contains numbers in increasing order"

We will do this by using the following logical formalism combining programs with logical specifications.

$$(\!| F |\!)\ P\ (\!| G |\!)$$

- $(\!| x > 0 |\!)\ P\ (\!| y * y < x |\!)$
- $(\!| \top |\!)\ P\ (\!| \forall i.((0 \leq i < |a| - 1) \to a[i] \leq a[i+1] |\!)$

Hoare triple: $(\!|F|\!)\ P\ (\!|G|\!)$

- F and G are FOL formulas.
  - Terms: integers, booleans, sequences and their operations (we will be more precise when needed).
  - Predicates: all standard arithmetic predicates about integers, booleans, and their operations.
  - F is called precondition
  - G is called postcondition
- P is a program written in an imperative programming language, which has:
  - a state[*] which is a **function** l **mapping any variable name** x **to an integer** $l(x)$
  - a given grammar.
- state l satisfies F (l is a F-state), written $l \models F$ when
  - $\mathcal{M} \models_l F$, for some **standard model** $\mathcal{M}$ (contains all integers and interprets terms and predicates in the "standard way")
- quantifiers in F and G contain variables not occuring in P.

[*] A state is very similar to the **environment** in FOL semantics

for any state l such that $l(x) = -2$ and $l(y) = 5$ and $l(z) = -1$:

- $l \models \neg(x + y < z)$ holds
- $l \models y - x * z < z$ does not hold

$$
\begin{array}{lll}
\text{Arithmetic Expressions:} & E ::= & 0 \mid 1 \mid 2 \mid 3 \mid \ldots \mid -1 \mid -2 \mid -3 \mid \ldots \\
& & \mid x \mid (-E) \mid (E + E) \mid (E - E) \mid (E * E) \\
& & \mid \texttt{if } B \texttt{ then } E \texttt{ else } E \mid S[E] \mid |S| \\
\text{Sequence Expressions:} & S ::= & s \mid S[..E] \mid S[E..] \mid S[E..E] \\
\text{Boolean Expressions:} & B ::= & \texttt{true} \mid \texttt{false} \mid (!B) \mid (B \,\&\, B) \mid (B \,||\, B) \\
& & \mid (E < E) \mid (E > E) \mid (E = E) \\
\text{Commands:} & C ::= & (x := E) \mid C; C \mid \texttt{if } B \texttt{ then } C \texttt{ else } C \mid \texttt{while } B \, \{C\}
\end{array}
$$

The book has a simpler language. Here (and in assignments/exams) we will use this richer language.

- Binding precedence for arithmetic expressions:
  - Negation $(-E)$ binds more tightly than
  - multiplication $(E_1 * E_2)$ which binds more tightly than
  - subtraction$(E_1 - E_2)$ and addition $(E_1 + E_2)$
- Binding precedence for boolean expressions:
  - Nation $(!E)$ binds more tightly than
  - conjunction $(E_1 \,\&\, E_2)$ disjunction $(E_1 \,||\, E_2)$.

Arithmetic Expressions:	$E ::= 0 \mid 1 \mid 2 \mid 3 \mid \ldots \mid -1 \mid -2 \mid -3 \mid \ldots$
	$\mid x \mid (-E) \mid (E + E) \mid (E - E) \mid (E * E)$
	$\mid \texttt{if}\ B\ \texttt{then}\ E\ \texttt{else}\ E \mid S[E] \mid \lvert S \rvert$
Sequence Expressions:	$S ::= s \mid S[..E] \mid S[E..] \mid S[E..E]$
Boolean Expressions:	$B ::= \texttt{true} \mid \texttt{false} \mid (!B) \mid (B \,\&\, B) \mid (B \,\vert\vert\, B)$
	$\mid (E < E) \mid (E > E) \mid (E = E)$
Commands:	$C ::= (x := E) \mid C; C \mid \texttt{if}\ B\ \texttt{then}\ C\ \texttt{else}\ C \mid \texttt{while}\ B\ \{C\}$

- There is a 1-1 correspondence between arithmetic/boolean expressions, and terms/fomulas of the FOL we use to write specifications.
  - the program expression $!(x < y)$ corresponds to the FOL formula $\neg(x < y)$
  - the program expression $!(x < z) \,\&\, (y < z)$ corresponds to the FOL formula $\neg(x < z) \wedge (y < z)$
  - what program expression corresponds to the FOL formula $(x = y)$?

$$(\!|F|\!)\ C\ (\!|G|\!)$$

High-level meaning of a Hoare triple: if we execute C in any state l that satisfies F, [...]$^{\text{(next slides)}}$ the final state will satisfy G.

$$l \text{ satisfies } F: \quad l \models F$$

l maps **variables** to **integers**.

Are the following true?

- $\{x \mapsto 1, y \mapsto 2\} \models x < y$
- $\{x \mapsto 1, y \mapsto 2\} \models x > y$
- $\{x \mapsto 1, y \mapsto 2\} \models (x + 1) = y$

$$\vdash_{\text{par}} (\!|F|\!)\ C\ (\!|G|\!)$$

Meaning of partial Hoare triple: if we execute C in any state l that satisfies F, and if C terminates, then the final state will satisfy G.

Equivalently: for all l such that $l \models F$, if $l, C \downarrow l'$ then for the final state l' we have $l' \models G$.

- **Correctness:** the pre- and post-conditions F and G give a specification of the program
- **Partial:** the above statement does not guarantee that C will terminate (which is a part of its correct operation)
- An infinite loop statisfies all pairs of pre-/post-conditions.

```
while (true) { x := 0 }
```

$$\vdash_{\mathrm{tot}} (\!|F|\!)\ C\ (\!|G|\!)$$

Meaning of total Hoare triple: if we execute C in any state l that satisfies F, then C terminates and the final state will satisfy G.

Equivalently: for all l such that $l \models F$ we have $l, C \downarrow l'$ and for the final state l' we have $l' \models G$.

- **Correctness:** the pre- and post-conditions F and G give a specification of the program
- **Total:** the above statement **does** guarantee that C will terminate

Q: are the following statements correct?

- If $\vdash_{\mathrm{tot}} (\!|F|\!)\ C\ (\!|G|\!)$ holds then $\vdash_{\mathrm{par}} (\!|F|\!)\ C\ (\!|G|\!)$ holds.
- If $\vdash_{\mathrm{par}} (\!|F|\!)\ C\ (\!|G|\!)$ holds then $\vdash_{\mathrm{tot}} (\!|F|\!)\ C\ (\!|G|\!)$ holds.

$$\vdash_{\text{tot}} (\!|F|\!)\ C\ (\!|G|\!)$$

Meaning of total Hoare triple: if we execute C in any state l that satisfies F, then C terminates and the final state will satisfy G.

Equivalently: for all l such that $l \models F$ we have $l, C \downarrow l'$ and for the final state l' we have $l' \models G$.

- **Correctness:** the pre- and post-conditions F and G give a specification of the program
- **Total:** the above statement **does** guarantee that C will terminate

Q: are the following statements correct?

- If $\vdash_{\text{tot}} (\!|F|\!)\ C\ (\!|G|\!)$ holds then $\vdash_{\text{par}} (\!|F|\!)\ C\ (\!|G|\!)$ holds. **Yes**
- If $\vdash_{\text{par}} (\!|F|\!)\ C\ (\!|G|\!)$ holds then $\vdash_{\text{tot}} (\!|F|\!)\ C\ (\!|G|\!)$ holds. **No**

Consider the imperative factorial function:

```
y := 1;
z := 0;
while (z != x) {
  z := z + 1;
  y := y * z;
}
```

Should we be able to prove the following?

- $\vdash_{\text{tot}} (\!| x \geq 0 |\!)\ C\ (\!| F |\!)$
- $\vdash_{\text{par}} (\!| x \geq 0 |\!)\ C\ (\!| F |\!)$
- $\vdash_{\text{tot}} (\!| \top |\!)\ C\ (\!| F |\!)$
- $\vdash_{\text{par}} (\!| \top |\!)\ C\ (\!| F |\!)$

where F is a formula, not specified here.

Consider the imperative factorial function:

```
y := 1;
z := 0;
while (z != x) {
  z := z + 1;
  y := y * z;
}
```

Should we be able to prove the following?

- $\vdash_{\text{tot}} (\!|x \geq 0|\!)\ C\ (\!|F|\!)$
- $\vdash_{\text{par}} (\!|x \geq 0|\!)\ C\ (\!|F|\!)$
- $\vdash_{\text{tot}} (\!|\top|\!)\ C\ (\!|F|\!)$ **No**: does not terminate in starting states with $x < 0$
- $\vdash_{\text{par}} (\!|\top|\!)\ C\ (\!|F|\!)$

where F is a formula, not specified here.

Consider the imperative factorial function:

```
y := 1;
z := 0;
while (z != x) {
  z := z + 1;
  y := y * z;
}
```

Specification: $\vdash_{\text{par}} (\!| x \geq 0 |\!)\ C\ (\!| F |\!)$

What is the right postcondition F for the above code?

Consider the imperative factorial function:

```
y := 1;
z := 0;
while (z != x) {
  z := z + 1;
  y := y * z;
}
```

Specification: $\vdash_{\text{par}} (\!| x \geq 0 |\!) \; C \; (\!| F |\!)$

What is the right postcondition F for the above code?

$$\vdash_{\text{par}} (\!| x \geq 0 |\!) \; C \; (\!| y = x! |\!)$$

What is the right post-condition for this version of factorial?

```
y := 1;
while (x != 0) {
  y := y * x;
  x := x - 1;
}
```

† Dafny calls them ghost variables

What is the right post-condition for this version of factorial?

```
y := 1;
while (x != 0) {
  y := y * x;
  x := x - 1;
}
```

$$\vdash_{\text{par}} (\!|\,(x = x_0) \wedge (x \geq 0)\,|\!)\; C\; (\!|\, y = x_0!\,|\!)$$

- logical variables†: variable x_0 is used only in formulas to "remember" some value from the starting state.
- program variables: variables used by the program

† Dafny calls them ghost variables

$$\frac{\vdash_{\mathrm{AR}} F' \rightarrow F \qquad (\!|F|\!)\ C\ (\!|G|\!) \qquad \vdash_{\mathrm{AR}} G \rightarrow G'}{(\!|F'|\!)\ C\ (\!|G'|\!)}\ \textsc{Impl}$$

$\vdash_{\mathrm{AR}} F' \rightarrow F$ means that the implication is derivable in FOL with natural numbers, equality, standard predicates etc., when all known properties of arithmetic are in our assumptions.

$$\dfrac{}{(\!|G[E/x]|\!)\ x = E\ (\!|G|\!)}\ \textsc{Asg}$$

$$\frac{(\!|F \wedge B|\!)\ C_1\ (\!|G|\!) \qquad (\!|F \wedge \neg B|\!)\ C_2\ (\!|G|\!)}{(\!|F|\!)\ \texttt{if}\ B\ \texttt{then}\ C_1\ \texttt{else}\ C_2\ (\!|G|\!)}\ \textsc{Cond}$$

$$\frac{(\!| F |\!) \; C_1 \; (\!| \eta |\!) \qquad (\!| \eta |\!) \; C_2 \; (\!| G |\!)}{(\!| F |\!) \; C_1; C_2 \; (\!| G |\!)} \; \textsc{Comp}$$

Prove the following Hoare triples:

- $(\!|y > 0|\!)\ x = y + 1\ (\!|x > 0|\!)$
- $(\!|x \geq y|\!)\ x = x - y\ (\!|x \geq 0|\!)$
- $(\!|x \geq y|\!)\ x = x - y;\ y = -x\ (\!|y \leq 0|\!)$
- Swap without temp:
  $(\!|(x = x_0) \wedge (y = y_0)|\!)\ x = y - x;\ y = y - x;\ x = x + y\ (\!|(x = y_0) \wedge (y = x_0)|\!)$
- $(\!|\top|\!)\ \texttt{if}\ x < 2\ \texttt{then}\ x = 2\ \texttt{else}\ x = x\ (\!|x \geq 2|\!)$

$$\frac{}{(\!|G[E/x]|\!)\ x = E\ (\!|G|\!)}\ \textsc{Asg} \qquad \frac{(\!|F \wedge B|\!)\ C_1\ (\!|G|\!) \qquad (\!|F \wedge \neg B|\!)\ C_2\ (\!|G|\!)}{(\!|F|\!)\ \texttt{if}\ B\ \texttt{then}\ C_1\ \texttt{else}\ C_2\ (\!|G|\!)}\ \textsc{Cond}$$

$$\frac{(\!|F|\!)\ C_1\ (\!|\eta|\!) \qquad (\!|\eta|\!)\ C_2\ (\!|G|\!)}{(\!|F|\!)\ C_1; C_2\ (\!|G|\!)}\ \textsc{Comp} \qquad \frac{\vdash_{AR} F' \to F \qquad (\!|F|\!)\ C\ (\!|G|\!) \qquad \vdash_{AR} G \to G'}{(\!|F'|\!)\ C\ (\!|G'|\!)}\ \textsc{Impl}$$