COMP2111 Week 7
Term 1, 2019
Week 6 recap

Week 6 recap

Hoare Logic:

- Soundness proof
- Finding a derivation
  - Weakest precondition
  - Invariants
- Total correctneess (termination) and variants
- Operational semantics
- $\mathcal{L}^+$: $\mathcal{L}$ with non-determinism
- Refinement calculus

Soundness and (relative) completeness

Theorem (Soundness)

Every derivable Hoare triple is valid: If $\vdash \{\varphi\}\, P\, \{\psi\}$ *then* $\models \{\varphi\}\, P\, \{\psi\}$.

Theorem (Relative completeness)

Given an oracle that can determine the truth of predicates, every valid Hoare triple is derivable: If $\models \{\varphi\}\, P\, \{\psi\}$ *then* $\vdash \{\varphi\}\, P\, \{\psi\}$.

Finding a proof: weakest precondition

Given a program P and a postcondition ψ, the weakest precondition $wp(P, \psi)$ is the predicate φ such that

- $\{\varphi\}\, P\, \{\psi\}$ is valid
- If $\{\varphi'\}\, P\, \{\psi\}$ is valid then $\varphi' \to \varphi$.

Computable based on the structure of P. Difficulty with loops...

Finding a proof: Invariants

In order to establish the validity of

$$\{\varphi\}\ \textbf{while}\ b\ \textbf{do}\ P\ \textbf{od}\ \{\psi\}$$

we find an **Invariant**, Inv, such that:

- $\varphi \to \mathsf{Inv}$ (establish)
- $\{b \wedge \mathsf{Inv}\}\ P\ \{\mathsf{Inv}\}$ (maintain)
- $\neg b \wedge \mathsf{Inv} \to \psi$ (conclude)

Total correctness (termination)

$$[\varphi]\, P\, [\psi]$$

Represents the statement that with precondition φ, program P will terminate at a state that satsifies ψ.

Can derive validity of $[\varphi]\, P\, [\psi]$ using Hoare logic with modified loop command:

$$\frac{[\varphi \land g \land (v = N)]\, P\, [\varphi \land (v < N)] \qquad (\varphi \land g) \to (v > 0)}{[\varphi]\, \textbf{while}\ g\ \textbf{do}\ P\ \textbf{od}\, [\varphi \land \neg g]} \quad \text{(loop)}$$

Finding a (total correctness) proof: Variants

In order to establish the validity of

$$[\varphi]\ \textbf{while}\ b\ \textbf{do}\ P\ \textbf{od}\ [\psi]$$

we find an **Invariant**, Inv, such that:

- $\varphi \rightarrow \mathsf{Inv}$ (establish)
- $[b \wedge \mathsf{Inv}]\ P\ [\mathsf{Inv}]$ (maintain)
- $\neg b \wedge \mathsf{Inv} \rightarrow \psi$ (conclude)

and a **variant**, $\mathsf{Var} \in \textsc{Exp}$, such that:

- $(b \wedge \mathsf{Inv}) \rightarrow (\mathsf{Var} > 0)$ (positivity)
- $[\mathsf{Inv} \wedge b \wedge (\mathsf{Var} = N)]\ P\ [\mathsf{Inv} \wedge (v < N)]$ (progress)

Operational semantics

- **Denotational semantics**: Assign a mathematical object (relation between states) to Programs
- **Operational semantics**: Construct (inductively) a relation, $\Downarrow$, between Programs and pairs of states

Example rule:

$$\frac{[P, \eta] \Downarrow \eta' \qquad [Q, \eta'] \Downarrow \eta''}{[P; Q, \eta] \Downarrow \eta''}$$

Non-determinism

Non-determinism = unspecified program branching

- More powerful: Encompasses deterministic behaviour
- More abstract: Mathematically nicer

$\mathcal{L}^+$: Non-deterministic extension of $\mathcal{L}$

- $P + Q$ – non-deterministic choice between P and Q
- P^* – loop for a non-deterministic number of times

Refinement calculus

Process for transforming abstract specifications into concrete code.

- Start with the most abstract program relating pre- and post-conditions
- Use refinement rules (based on rules of Hoare Logic) to **refine** the program – i.e. make it less abstract
- The end result will be some program in $\mathcal{L}$ (or similar)

Need to know for this course

Nothing will be assessed in great detail, however, a good understanding of:

- Weak precondition
- Invariants
- Termination and variants
- Non-determinism

will help a lot.