COMP2111 Week 6
Term 1, 2019
Week 5 recap

Week 5 recap

Hoare Logic:

- Simple imperative language $\mathcal{L}$
- Hoare triple $\{\varphi\}\, P\, \{\psi\}$ (SYNTAX)
- Derivation rules (PROOFS)
- Semantics for Hoare logic (SEMANTICS)

The language $\mathcal{L}$

The language $\mathcal{L}$ is a simple imperative programming language made up of four statements:

Assignment: $x := e$

where x is a variable and e is an arithmetic expression.

Sequencing: $P ; Q$

Conditional: **if** b **then** P **else** Q **fi**

where b is a boolean expression.

While: **while** b **do** P **od**

Hoare triple (Syntax)

$$\{\varphi\}\,P\,\{\psi\}$$

Intuition:
If φ holds in a state of some computational model
then ψ holds in the state reached after a successful execution of P.

$$\vdash \{\varphi\}\,P\,\{\psi\}$$

$\{\varphi\}\,P\,\{\psi\}$ is **derivable** using the proof rules of Hoare Logic

$$\models \{\varphi\}\,P\,\{\psi\}$$

$\{\varphi\}\,P\,\{\psi\}$ is **valid** according to the semantic interpretation.

Hoare triple (Syntax)

$$\{\varphi\}\, P\, \{\psi\}$$

Intuition:
If φ holds in a state of some computational model
then ψ holds in the state reached after a successful execution of P.

$$\vdash \{\varphi\}\, P\, \{\psi\}$$

$\{\varphi\}\, P\, \{\psi\}$ is **derivable** using the proof rules of Hoare Logic

$$\models \{\varphi\}\, P\, \{\psi\}$$

$\{\varphi\}\, P\, \{\psi\}$ is **valid** according to the semantic interpretation.

Hoare triple (Syntax)

$$\{\varphi\}\, P\, \{\psi\}$$

Intuition:
If φ holds in a state of some computational model
then ψ holds in the state reached after a successful execution of P.

$$\vdash \{\varphi\}\, P\, \{\psi\}$$

$\{\varphi\}\, P\, \{\psi\}$ is **derivable** using the proof rules of Hoare Logic

$$\models \{\varphi\}\, P\, \{\psi\}$$

$\{\varphi\}\, P\, \{\psi\}$ is **valid** according to the semantic interpretation.

Hoare logic rules

$$\frac{}{\{\varphi[e/x]\}\ x := e\ \{\varphi\}} \quad \text{(ass)}$$

$$\frac{\{\varphi\}\ P\ \{\psi\} \qquad \{\psi\}\ Q\ \{\rho\}}{\{\varphi\}\ P; Q\ \{\rho\}} \quad \text{(seq)}$$

$$\frac{\{\varphi \wedge g\}\ P\ \{\psi\} \qquad \{\varphi \wedge \neg g\}\ Q\ \{\psi\}}{\{\varphi\}\ \mathbf{if}\ g\ \mathbf{then}\ P\ \mathbf{else}\ Q\ \mathbf{fi}\ \{\psi\}} \quad \text{(if)}$$

Hoare logic rules

$$\frac{}{\{\varphi(e)\}\ x := e\ \{\varphi(x)\}} \quad \text{(ass)}$$

$$\frac{\{\varphi\}\ P\ \{\psi\} \qquad \{\psi\}\ Q\ \{\rho\}}{\{\varphi\}\ P; Q\ \{\rho\}} \quad \text{(seq)}$$

$$\frac{\{\varphi \wedge g\}\ P\ \{\psi\} \qquad \{\varphi \wedge \neg g\}\ Q\ \{\psi\}}{\{\varphi\}\ \textbf{if}\ g\ \textbf{then}\ P\ \textbf{else}\ Q\ \textbf{fi}\ \{\psi\}} \quad \text{(if)}$$

Hoare logic rules

$$\frac{\{\varphi \wedge g\}\ P\ \{\varphi\}}{\{\varphi\}\ \textbf{while}\ g\ \textbf{do}\ P\ \textbf{od}\ \{\varphi \wedge \neg g\}} \quad \text{(loop)}$$

$$\frac{\varphi' \to \varphi \qquad \{\varphi\}\ P\ \{\psi\} \qquad \psi \to \psi'}{\{\varphi'\}\ P\ \{\psi'\}} \quad \text{(cons)}$$

Hoare logic semantics

$\textsc{Env}$: set of environments (functions that map variables to numeric values)

$\langle\cdot\rangle : \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env})$, given by:

$$\langle\varphi\rangle := \{\eta \in \textsc{Env} \ : \ [\![\varphi]\!]^\eta = \texttt{true}\}.$$

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

Hoare logic semantics

$\textsc{Env}$: set of environments (functions that map variables to numeric values)

$\langle\cdot\rangle : \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env})$, given by:

$$\langle\varphi\rangle := \{\eta \in \textsc{Env} \ : \ [\![\varphi]\!]^{\eta} = \texttt{true}\}.$$

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

Hoare logic semantics

$\textsc{Env}$: set of environments (functions that map variables to numeric values)

$\langle \cdot \rangle : \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env})$, given by:

$$\langle \varphi \rangle := \{\eta \in \textsc{Env} \;:\; [\![\varphi]\!]^{\eta} = \texttt{true}\}.$$

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

Hoare logic semantics

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

For predicates: $[\![\varphi]\!] = \{(\eta, \eta) \ : \ \eta \in \langle\varphi\rangle\}$

For programs: Inductively:

Hoare logic semantics

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

For predicates: $[\![\varphi]\!] = \{(\eta, \eta) \ : \ \eta \in \langle \varphi \rangle\}$

For programs: Inductively:

- $[\![P; Q]\!] = [\![P]\!]; [\![Q]\!]$
- $[\![\textbf{if } b \textbf{ then } P \textbf{ else } Q \textbf{ fi}]\!] = [\![b; P]\!] \cup [\![\neg b; Q]\!]$
- $[\![\textbf{while } b \textbf{ do } P \textbf{ od}]\!] = [\![b; P]\!]^*; [\![\neg b]\!]$

where $R; S$ is the **relational composition** of R and S, and R^* is the **transitive closure of** R...

Hoare logic semantics

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

For predicates: $[\![\varphi]\!] = \{(\eta, \eta) \ : \ \eta \in \langle\varphi\rangle\}$

For programs: Inductively:

- $[\![P; Q]\!] = [\![P]\!]; [\![Q]\!]$
- $[\![\textbf{if } b \textbf{ then } P \textbf{ else } Q \textbf{ fi}]\!] = [\![b; P]\!] \cup [\![\neg b; Q]\!]$
- $[\![\textbf{while } b \textbf{ do } P \textbf{ od}]\!] = [\![b; P]\!]^*; [\![\neg b]\!]$

where $R; S$ is the **relational composition** of R and S, and R^* is the **transitive closure of** R...

Hoare logic semantics

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \to \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

For predicates: $[\![\varphi]\!] = \{(\eta, \eta) \;:\; \eta \in \langle\varphi\rangle\}$

For programs: Inductively:

- $[\![P; Q]\!] = [\![P]\!]; [\![Q]\!]$
- $[\![\textbf{if } b \textbf{ then } P \textbf{ else } Q \textbf{ fi}]\!] = [\![b; P]\!] \cup [\![\neg b; Q]\!]$
- $[\![\textbf{while } b \textbf{ do } P \textbf{ od}]\!] = [\![b; P]\!]^*; [\![\neg b]\!]$

where $R; S$ is the **relational composition** of R and S, and R^* is the **transitive closure of** R...

Hoare logic semantics

$[\![\cdot]\!] : \textsc{Programs} \cup \textsc{Predicates} \rightarrow \mathsf{Pow}(\textsc{Env} \times \textsc{Env})$

For predicates: $[\![\varphi]\!] = \{(\eta, \eta) \ : \ \eta \in \langle\varphi\rangle\}$

For programs: Inductively:

- $[\![P; Q]\!] = [\![P]\!]; [\![Q]\!]$
- $[\![\textbf{if } b \textbf{ then } P \textbf{ else } Q \textbf{ fi}]\!] = [\![b; P]\!] \cup [\![\neg b; Q]\!]$
- $[\![\textbf{while } b \textbf{ do } P \textbf{ od}]\!] = [\![b; P]\!]^*; [\![\neg b]\!]$

where $R; S$ is the **relational composition** of R and S, and R^* is the **transitive closure of** R...

Transitive closure

Given a binary relation $R \subseteq A \times A$ we define R^n inductively:

- $R^0 = \Delta$ the diagonal relation
- $R^{i+1} = R^i; R$ for $i \geq 0$.

The **transitive closure**, R^* is then defined to be:

$$
\begin{aligned}
R^* &:= \bigcup_{i=0}^{\infty} R^i \\
&= \{(x, y) : (x, y) \in R^i \text{ for some } i \in \mathbb{N}\}.
\end{aligned}
$$

Need to know for this course

- Write programs in $\mathcal{L}$.
- Give proofs using the Hoare logic rules (full and outline)
- Definition of $[\![\cdot]\!]$
- Definition of composition and transitive closure