Introduction to Formal Methods

Lecture 11

Hoare Logic for Concurrent Programs

Hossein Hojjat & Fatemeh Ghassemi

October 28, 2018

Program Verification with Hoare Triples

- Is the following true?

$\{x = 0\}$

```
y := x;
x := x + 1;
```

$\{x = 1 \land y = 0\}$

- YES!

Program Verification with Hoare Triples

- Is the following still true?

$\{x = 0\}$
`y := x;`
`x := x + 1;`
$\{x = 1 \land y = 0\}$
	$\|\|$	`x := 5;`

Program Verification with Hoare Triples

- Is the following still true?

$\{x = 0\}$

```
y := x;
x := x + 1;
```

$\{x = 1 \wedge y = 0\}$ $\quad \| \quad$ `x := 5;`

- NO!

Program Verification with Hoare Triples

- Is the following still true?

$\{x = 0\}$

```
y := x;
```

$\{x + 1 = 1 \wedge y = 0\}$

```
x := x + 1;
```

$\{x = 1 \wedge y = 0\}$

$\|$

```
x := 5;
```

- NO!

Program Verification with Hoare Triples

- Is the following still true?

$\{x = 0\}$
```
y := x;
```
$\cancel{\{x + 1 = 1 \wedge y = 0\}}$
```
x := x + 1;
```
$\{x = 1 \wedge y = 0\}$

$\|$

```
x := 5;
```

- NO!
- The parallel process may interfere with the intermediate assertions

Parallel Composition

- Extend the language of previous lectures with parallel composition

$$
\begin{aligned}
e ::= \;& n \mid x \mid e_1 + e_2 \mid e_1 = e_2 \\
c ::= \;& x := e \mid \texttt{if}\, e \,\texttt{then}\; c_1 \;\texttt{else}\; c_2 \mid \\
& \texttt{while}\, e \;\texttt{do}\; c \mid \texttt{skip} \mid c_1 \,;\, c_2 \mid \\
& c_1 \parallel c_2
\end{aligned}
$$

Rule for Parallel Composition

- Can we derive a Hoare triple for parallel composition from the triples of each command?

First Attempt:

$$\frac{\vdash \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash \{P_2\}\ c_2\ \{Q_2\}}{\vdash \{P_1 \land P_2\}\ c_1 \parallel c_2\ \{Q_1 \land Q_2\}}$$

- **Intuition:** if we satisfy the preconditions of c_1 and c_2, their postconditions will be satisfied too

Unsoundness of First Attempt

$$\frac{\vdash \ \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash \ \{P_2\}\ c_2\ \{Q_2\}}{\vdash \ \{P_1 \wedge P_2\}\ c_1 \parallel c_2\ \{Q_1 \wedge Q_2\}}$$

- This rule is not always sound, consider:

$\{x = 1\}\ \ y := 0\ \ \{x = 1\}$ $\qquad\qquad$ $\{\textit{true}\}\ \ x := 10\ \ \{\textit{true}\}$

- It does not hold that

$$\{x = 1 \wedge \textit{true}\} \quad y := 0 \parallel x := 10 \quad \{x = 1 \wedge \textit{true}\}$$

Second Attempt

$$\frac{\vdash \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash \{P_2\}\ c_2\ \{Q_2\}}{\vdash \{P_1 \land P_2\}\ c_1 \parallel c_2\ \{Q_1 \land Q_2\}}$$

- If c_1 and c_2 do not read and write the same variables,
  and all the pres- and post- conditions talk about different variables
- What's wrong with this?

Second Attempt

$$\frac{\vdash \ \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash \ \{P_2\}\ c_2\ \{Q_2\}}{\vdash \ \{P_1 \wedge P_2\}\ c_1 \parallel c_2\ \{Q_1 \wedge Q_2\}}$$

- If c_1 and c_2 do not read and write the same variables, and all the pres- and post- conditions talk about different variables
- What's wrong with this?
- No way to prove some program
- The rule is **incomplete**

Third Attempt

$$\frac{\vdash \ \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash \ \{P_2\}\ c_2\ \{Q_2\}}{\vdash \ \{P_1 \wedge P_2\}\ c_1 \parallel c_2\ \{Q_1 \wedge Q_2\}}$$

- If a command does not modify any variable from the pre-condition of the other triple
- Let $\mathit{UPD}(c)$ be the set of variables that are updated (modified) in c

$$\mathit{FV}(P_1) \cap \mathit{UPD}(c_2) = \emptyset$$

$$\mathit{FV}(P_2) \cap \mathit{UPD}(c_1) = \emptyset$$

Third Attempt

$$\frac{\vdash \ \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash \ \{P_2\}\ c_2\ \{Q_2\}}{\vdash \ \{P_1 \wedge P_2\}\ c_1 \parallel c_2\ \{Q_1 \wedge Q_2\}}$$

- If a command does not modify any variable from the pre-condition of the other triple
- Let $\mathit{UPD}(c)$ be the set of variables that are updated (modified) in c

$$\mathit{FV}(P_1) \cap \mathit{UPD}(c_2) = \emptyset$$
$$\mathit{FV}(P_2) \cap \mathit{UPD}(c_1) = \emptyset$$

- Still incomplete: cannot prove

$$\{\mathsf{x} = 0\}$$
$$\texttt{x := x + 1} \parallel \texttt{x := x + 2}$$
$$\{\mathsf{x} = 3\}$$

Fourth Attempt

$$\frac{\vdash\ \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash\ \{P_2\}\ c_2\ \{Q_2\}}{\vdash\ \{P_1 \land P_2\}\ c_1 \parallel c_2\ \{Q_1 \land Q_2\}}$$

If $\mathit{UPD}(c_1) \cap (\mathit{FV}(P_2) \cup \mathit{FV}(Q_2)) = \emptyset$ and $\mathit{UPD}(c_2) \cap (\mathit{FV}(P_1) \cup \mathit{FV}(Q_1)) = \emptyset$

Fourth Attempt

$$\frac{\vdash \ \{P_1\}\ c_1\ \{Q_1\} \qquad \vdash \ \{P_2\}\ c_2\ \{Q_2\}}{\vdash \ \{P_1 \wedge P_2\}\ c_1 \parallel c_2\ \{Q_1 \wedge Q_2\}}$$

If $\mathit{UPD}(c_1) \cap (\mathit{FV}(P_2) \cup \mathit{FV}(Q_2)) = \emptyset$ and $\mathit{UPD}(c_2) \cap (\mathit{FV}(P_1) \cup \mathit{FV}(Q_1)) = \emptyset$

Still unsound. Consider:

$\{x = 0\}\ \ y := x; z := y\ \ \{z = 0\}$ $\qquad\qquad$ $\{\mathit{true}\}\ \ y := 10\ \ \{\mathit{true}\}$

It does not hold that

$$\{x = 0 \wedge \mathit{true}\}\ \ y := x; z := y \parallel y := 10\ \ \{z = 0 \wedge \mathit{true}\}$$

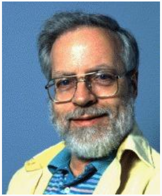
Diagnose: $y := 10$ interferes with the proof of

$$\{x = 0\}\ \ y := x; z := y\ \ \{z = 0\}$$

$$y = 0$$

Owicki-Gries Reasoning

- Susan Owicki,
  "Axiomatic proof techniques for parallel programs",
  Cornell University, Ithaca, NY, 1975
  - Under supervision of Prof. David Gries
- First complete logic for partial correctness of concurrent programs that communicate using shared variables



Interference Freedom

- **Interference Freedom:** every assertion used in the local verification is not invalidated by the execution of the other process

P_1:	P_2:
$\{p_1\}$	$\{q_1\}$
c_1	a_1
$\{p_2\}$	$\{q_2\}$
c_2	a_2
$\dots$	$\dots$

We say that they are interference free iff

$$\forall p_i \in \text{assertions of } P_1 \wedge \forall a_j \in \text{atomic actions of } P_2,$$
$$\{p_i \wedge \text{pre } a_j\}$$
$$a_j$$
$$\{p_i\}$$
(and vice versa)

- If P_1 has n statements and P_2 has m statements, proving interference freedom requires proving $O(n \times m)$ correctness formulas

Owicki-Gries method (1976)

$$\dfrac{\begin{array}{c}\vdash \ \{P_1\}\ c_1\ \{Q_1\} \\ \vdash \ \{P_2\}\ c_2\ \{Q_2\} \\ \text{the two proofs are non-interfering}\end{array}}{\vdash \ \{P_1 \wedge P_2\}\ c_1 \parallel c_2\ \{Q_1 \wedge Q_2\}}$$

Example

- These two proof outlines are correct but not interference free
- For example, the assertion $x = 0$ is not preserved against the atomic action $x := x + 2$

$$\begin{array}{c} \{x = 0\} \\ x := x + 2; \\ \{x = 2\} \end{array} \quad \Bigg\| \quad \begin{array}{c} \{\mathit{true}\} \\ x := 0; \\ \{x = 0\} \end{array} \qquad \begin{array}{c} \{x = 0 \wedge x = 0\} \\ x := x + 2; \\ \{x = 0\} \end{array}$$

- By weakening the postconditions we obtain both correct and interference free proof outlines:

$$\begin{array}{c} \{x = 0\} \\ x := x + 2; \\ \{x = 0 \vee x = 2\} \end{array} \Bigg\| \begin{array}{c} \{\mathit{true}\} \\ x := 0; \\ \{x = 0 \vee x = 2\} \end{array} \qquad \begin{array}{c} \{(x = 0 \vee x = 2) \wedge x = 0\} \\ x := x + 2; \\ \{x = 0 \vee x = 2\} \end{array} \quad \Rightarrow \begin{array}{c} \{x = 0\} \\ x := x + 2; \\ \{x = 0 \vee x = 2\} \end{array}$$

Completeness

- Can you prove the following?

$$\{x = 0\}$$

$$x := x + 1 \quad \Big\| \quad x := x + 1$$

$$\{x = 2\}$$

Completeness

- Can you prove the following?
- We can prove something weaker

$$\begin{array}{ccc} & \{x = 0\} & \\ \begin{array}{c}\{x = 0 \lor x = 1\} \\ x := x + 1 \\ \{x = 1 \lor x = 2\}\end{array} & \Big\| & \begin{array}{c}\{x = 0 \lor x = 1\} \\ x := x + 1 \\ \{x = 1 \lor x = 2\}\end{array} \\ & \{x = 1 \lor x = 2\} & \end{array}$$

- But how can we derive the postcondition $x = 2$?
- We need **auxiliary** variables:
- Variables that do not affect the control flow nor the data flow of the other variables, but record information useful for the proof

Auxiliary Variables

Add two auxiliary variables a and b:
Represent the contribution of each thread to x

$$\{x = 0\}$$
$$(a, b) := (0, 0)$$

$$(x, a) := (x + 1, 1) \qquad || \qquad (x, b) := (x + 1, 1)$$

$$\{x = 2\}$$

$(x_1, x_2) := (e_1, e_2)$ atomic parallel assignment

Auxiliary Variables

Add two auxiliary variables a and b:
Represent the contribution of each thread to x

$$\{x = 0\}$$
$$(a, b) := (0, 0)$$
$$\{x = a + b \wedge a = 0 \wedge b = 0\}$$

$$\begin{array}{ccc} \{x = a + b \wedge a = 0\} & & \{x = a + b \wedge b = 0\} \\ (x, a) := (x + 1, 1) & \| & (x, b) := (x + 1, 1) \\ \{x = a + b \wedge a = 1\} & & \{x = a + b \wedge b = 1\} \end{array}$$

$$\{x = 2\}$$

$(x_1, x_2) := (e_1, e_2)$ atomic parallel assignment

Horn Clauses for Concurrent Counters

Global Variable: n

Left Thread

$$
\begin{array}{rcl}
n = 0 & \rightarrow & P_1(n) \\
P_1(n) \land n' = n + 1 & \rightarrow & P_2(n') \\
P_2(n) \land n' = n - 1 & \rightarrow & P_1(n')
\end{array}
$$

Right Thread

$$
\begin{array}{rcl}
n = 0 & \rightarrow & Q_1(n) \\
Q_1(n) \land n' = n - 1 & \rightarrow & Q_2(n') \\
Q_2(n) \land n' = n + 1 & \rightarrow & Q_1(n')
\end{array}
$$

$$Q_2(n) \land P_2(n) \land (n = 0) \rightarrow false$$

Horn Clauses for Concurrent Counters

Global Variable: n

Left Thread

$$
\begin{array}{lcl}
n = 0 & \rightarrow & P_1(n) \\
P_1(n) \land n' = n + 1 & \rightarrow & P_2(n') \\
P_2(n) \land n' = n - 1 & \rightarrow & P_1(n')
\end{array}
$$

Right Thread

$$
\begin{array}{lcl}
n = 0 & \rightarrow & Q_1(n) \\
Q_1(n) \land n' = n - 1 & \rightarrow & Q_2(n') \\
Q_2(n) \land n' = n + 1 & \rightarrow & Q_1(n')
\end{array}
$$

$$Q_2(n) \land P_2(n) \land (n = 0) \rightarrow false$$

Unsound: proves to be correct although the real system does not have the property

$P_1(n) \equiv (n = 0) \quad P_2(n) \equiv (n = 1) \quad Q_1(n) \equiv (n = 0) \quad Q_2(n) \equiv (n = -1)$

Owicki-Gries Interference-Free Conditions

Global Variable: n

$$
\begin{aligned}
&P_1(n,1) \wedge Q_1(n,1) \wedge n' = n+1 \rightarrow Q_1(n',2)\\
&P_1(n,2) \wedge Q_2(n,1) \wedge n' = n+1 \rightarrow Q_2(n',2)\\
&P_2(n,1) \wedge Q_1(n,2) \wedge n' = n-1 \rightarrow Q_1(n',1)\\
&P_2(n,2) \wedge Q_2(n,2) \wedge n' = n-1 \rightarrow Q_2(n',1)\\
&Q_1(n,1) \wedge P_1(n,1) \wedge n' = n-1 \rightarrow P_1(n',2)\\
&Q_1(n,2) \wedge P_2(n,1) \wedge n' = n-1 \rightarrow P_2(n',2)\\
&Q_2(n,1) \wedge P_1(n,2) \wedge n' = n+1 \rightarrow P_1(n',1)\\
&Q_2(n,2) \wedge P_2(n,2) \wedge n' = n+1 \rightarrow P_2(n',1)
\end{aligned}
$$

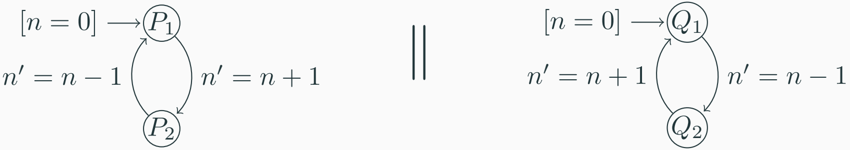
Owicki-Gries Interference-Free Conditions

Global Variable: n

$$P_1(n,1) \land Q_1(n,1) \land n' = n + 1 \to Q_1(n',2)$$
$$P_1(n,2) \land Q_2(n,1) \land n' = n + 1 \to Q_2(n',2)$$
$$P_2(n,1) \land Q_1(n,2) \land n' = n - 1 \to Q_1(n',1)$$
$$P_2(n,2) \land Q_2(n,2) \land n' = n - 1 \to Q_2(n',1)$$
$$Q_1(n,1) \land P_1(n,1) \land n' = n - 1 \to P_1(n',2)$$
$$Q_1(n,2) \land P_2(n,1) \land n' = n - 1 \to P_2(n',2)$$
$$Q_2(n,1) \land P_1(n,2) \land n' = n + 1 \to P_1(n',1)$$
$$Q_2(n,2) \land P_2(n,2) \land n' = n + 1 \to P_2(n',1)$$

Monolithic Encoding

Global Variable: n



- Uses only one relation symbol to model the system: $\mathbf{R}(id, n, t_1, t_2)$
- Invariant covering the whole system
- Simpler and creates more elegant solutions

$$
\begin{array}{rcl}
(n = 0) \wedge (t_1 = 1) \wedge (t_2 = 1) & \rightarrow & \mathbf{R}(id, n, t_1, t_2) \\
\mathbf{R}(1, n, 1, t_2) \wedge (n' = n + 1) & \rightarrow & \mathbf{R}(1, n', 2, t_2) \\
\mathbf{R}(1, n, 2, t_2) \wedge (n' = n - 1) & \rightarrow & \mathbf{R}(1, n', 1, t_2) \\
\mathbf{R}(2, n, t_1, 1) \wedge (n' = n - 1) & \rightarrow & \mathbf{R}(2, n', t_1, 2) \\
\mathbf{R}(2, n, t_1, 2) \wedge (n' = n + 1) & \rightarrow & \mathbf{R}(2, n', t_1, 1)
\end{array}
$$

Global Variable: n

Interference-Free Conditions

$$
\begin{array}{lcl}
\mathbf{R}(1, n, 1, t2) \wedge \mathbf{R}(2, n, 1, t_2) \wedge (n' = n + 1) & \rightarrow & \mathbf{R}(2, n', 2, t_2) \\
\mathbf{R}(1, n, 2, t_2) \wedge \mathbf{R}(2, n, 2, t_2) \wedge (n' = n - 1) & \rightarrow & \mathbf{R}(2, n', 1, t_2) \\
\mathbf{R}(2, n, t_1, 1) \wedge \mathbf{R}(1, n, t_1, 1) \wedge (n' = n - 1) & \rightarrow & \mathbf{R}(1, n', t_1, 2) \\
\mathbf{R}(2, n, t_1, 2) \wedge \mathbf{R}(1, n, t_1, 2) \wedge (n' = n + 1) & \rightarrow & \mathbf{R}(1, n', t_1, 1)
\end{array}
$$

Rule for Parallel Composition

$$\frac{\begin{array}{c}\vdash \{P_1\}\ c_1\ \{Q_1\} \\ \vdash \{P_2\}\ c_2\ \{Q_2\} \\ \textit{interference freedom}\end{array}}{\vdash \{P_1 \wedge P_2\}\ c_1 \parallel c_2\ \{Q_1 \wedge Q_2\}}$$

- This rule is **not** compositional
- The specification of a program c is not just $\{P\}\_\{Q\}$, but also all the intermediate assertions in the outline
- A change in one of the components may affect the proof, not only of the modified component, but also of all the others

Rely-Guarantee

- **Rely-Guarantee** is a well-known compositional method for proving Hoare logic properties of concurrent programs
- Rough idea: instead of trying to write interference-free proofs, explicitly account for the allowed interference
- No additional interference checks required
- Pioneered by Cliff Jones (1981, 1983)

Rely-Guarantee

$$\boxed{R, G \vdash \{P\}\ c\ \{Q\}}$$

- Pre-condition $P(x)$ assertion describing initial state
- Rely condition $R(x, x')$ relation describing atomic steps of environment
- Post-condition $Q(x)$ assertion describing final state
- Guarantee condition $G(x, x')$ relation describing atomic steps of the program

$$s_0 \xrightarrow{\text{env}} s_1 \xrightarrow{\text{prog}} s_2 \xrightarrow{\text{env}} s_3 \xrightarrow{\text{prog}} s_4 \xrightarrow{\text{prog}} s_5 \xrightarrow{\text{env}} s_6 \cdots s_{n-1} \xrightarrow{\text{prog}} s_n$$

If $P(s_0)$ and $R(s_i, s_{i+1})$ for all $s_i \xrightarrow{\text{env}} s_{i+1}$,
then $G(s_j, s_{j+1})$ for all $s_j \xrightarrow{\text{prog}} s_{j+1}$, and $Q(s_n)$ (the final state).

Example

$$\{x = 0\}$$

$$G_2, G_1 \vdash \{x = 0 \lor x = 2\} \quad x := x + 1 \quad \{x = 1 \lor x = 3\}$$

$$\|$$

$$G_1, G_2 \vdash \{x = 0 \lor x = 1\} \quad x := x + 2 \quad \{x = 2 \lor x = 3\}$$

$$\{x = 3\}$$

$$G_1 \equiv (x = 0 \land x' = 1) \lor (x = 2 \land x' = 3)$$
$$G_2 \equiv (x = 0 \land x' = 2) \lor (x = 1 \land x' = 3)$$

References

- Susan Owicki and David Gries: “An Axiomatic Proof Technique for Parallel Programs”, Acta Informatica, 1976.
- Viktor Vafeiadis: “Modular fine-grained concurrency verification”, PhD thesis, University of Cambridge, 2007.
- Sergey Grebenshchikov, Nuno P. Lopes, Corneliu Popeea, Andrey Rybalchenko: “Synthesizing Software Verifiers from Proof Rules”, PLDI 2012.
- Hossein Hojjat, Philipp Rümmer, Pavle Subotic, Wang Yi: “Horn Clauses for Communicating Timed Systems”, HCVS 2014.