CS766: Analysis of concurrent programs (first half) 2021

Lecture 11: Proof systems for concurrent programs

Instructor: Ashutosh Gupta

IITB, India

Compile date: 2021-02-20

Explicit events analysis is limited

- We have seen analysis of concurrent programs with a bounded set of events
- How do we analyze when we do not have such limits?

We need a proof system.

Commentary: Example and presentation ideas are borrowed from `https://fzn.fr/teaching/mpri/2010/fzn-mpri-2010-3.pdf`

Topic 11.1

Proof systems for programs

Hoare logic for sequential programs

- Hoare logic is one of the frameworks for the reasoning over programs
- Other logics reason over sets of traces and transitions instead of states
- Can we develop something for concurrent programs?

Proof systems for concurrent programs

- Näive extension of Hoare logic by treating the vector of program counters as a variable
  Not a practical solution(why?)

- Two proof systems that extend Hoare logic for concurrency
  1. Owicki-Gries
  2. Rely-Guarantee (not covered in this lecture)

Introducing parallel composition

We add parallel composition in our simple programming language.

$$\mathtt{c} \;||\; \mathtt{c}$$

We define interleaved semantics as follows

$$((v, \mathtt{c}_1||\mathtt{c}_2), (v', \mathtt{c}_1'||\mathtt{c}_2)) \in T \qquad \text{if} \qquad ((v, \mathtt{c}_1), (v', \mathtt{c}_1')) \in T$$

$$((v, \mathtt{c}_1||\mathtt{c}_2), (v', \mathtt{c}_1||\mathtt{c}_2')) \in T \qquad \text{if} \qquad ((v, \mathtt{c}_2), (v', \mathtt{c}_2')) \in T$$

$$((v, \mathtt{skip}||\mathtt{skip}), (v, \mathtt{skip})) \in T$$

Topic 11.2

Owicki-Gries proof system

How can we reason over parallel composition?

- Consider all possible interleavings
- Reasoning needs ability to summarize effect of all of them in state formulas

Commentary: A state formula only refers to variables of a program and does not relate value the variables at different time points.

Example 11.1

Consider

Assume assignments are atomic

$$x := x + 1 \quad || \quad x := x + 2$$

We my conclude : if initially $x = 0$*, the program finishes with* $x = 3$*.*

We may write Hoare triple

$$\{x = 0\} \quad x := x + 1 \quad || \quad x := x + 2 \quad \{x = 3\}$$

How can we derive the Hoare triple from the behavior of parts?

Soundness vs completeness

We will design the proof rule for parallel composition.

As we go along, we may be unsound or incomplete, or both.

We will fix those issues in small steps.

Attempt 1: Let us model it like nondeterminism (Incomplete and unsound)

$$[\textsc{ParLikeNondet}]\ \frac{\{P\}c_1\{Q\} \quad \{P\}c_2\{Q\}}{\{P\}c_1||c_2\{Q\}}$$

Example 11.2

$$\frac{\{x = 0\}x := x + 1\{x = 1\} \qquad \{y = 0\}y := y + 1\{y = 1\}}{\{x = y = 0\}x := x + 1||y := y + 1\{x = y = 1\}}\ \textit{Rejected by the rule}$$

Example 11.3

$$\frac{\{x = 0\}x := x + 1\{x = 1\} \qquad \{x = 0\}x := x + 1\{x = 1\}}{\{x = 0\}x := x + 1||x := x + 1\{x = 1\}}\ ✗$$

We need to combine the effect of both the programs.

Attempt 2: Conjunction of precondition and postcondition (Unsound)

$$[\textsc{ParConjunctive}]\frac{\{P_1\}\mathsf{c}_1\{Q_1\} \quad \{P_2\}\mathsf{c}_2\{Q_2\}}{\{P_1 \land P_2\}\mathsf{c}_1||\mathsf{c}_2\{Q_1 \land Q_2\}}$$

Example 11.4

$$\frac{\{y = 1\}x := 1\{y = 1\} \qquad \{\top\}y := 0\{\top\}}{\{y = 1\}x := 1||y := 0\{y = 1\}} ✗$$

What went wrong? Thread two interfered with truth value of (pre)postcondition of thread one.

We need to detect interference.

Attempt 3: Monitor interference (Still unsound)

The following condition says that program c does not modify any variable in set of formulas Σ.

$$NoMod(\mathtt{c}, \Sigma) \triangleq modifyVars(\mathtt{c}) \cap FreeVars(\Sigma) = \emptyset$$

Commentary: We choose FreeVars because we may have quantified formulas in our pre/postcondition

$$[\textsc{ParNoMod}]\frac{\{P_1\}\mathtt{c}_1\{Q_1\} \quad \{P_2\}\mathtt{c}_2\{Q_2\}}{\{P_1 \land P_2\}\mathtt{c}_1||\mathtt{c}_2\{Q_1 \land Q_2\}} NoMod(\mathtt{c}_2, \{P_1, Q_1\}) \text{ and } NoMod(\mathtt{c}_1, \{P_2, Q_2\})$$

Example 11.5

Is the above rule applicable?

$$\frac{\{\mathtt{z} = 0\}\mathtt{x} := \mathtt{z}; \mathtt{y} := \mathtt{x}\{\mathtt{y} = 0\} \qquad \{\top\}\mathtt{x} := 2\{\top\}}{\{\mathtt{z} = 0\}\mathtt{x} := \mathtt{z}; \mathtt{y} := \mathtt{x}||\mathtt{x} := 2\{\mathtt{y} = 0\}} ✗$$

What went wrong? We did not check for interference on intermediate formulas.

We need to detect interference at all intermediate steps.

Example : interference explicated

Example 11.6

Let us look at our example again and write the expanded proof.

x := 2 interferes with x = 0

$$\dfrac{\dfrac{\{z = 0\}\mathrm{x} := \mathrm{z};\{x = 0\} \quad \{x = 0\}\mathrm{y} := \mathrm{x}\{y = 0\}}{\{z = 0\}\mathrm{x} := \mathrm{z};\mathrm{y} := \mathrm{x}\{y = 0\}} \qquad \{\top\}\mathrm{x} := 2\{\top\}}{\{z = 0\}\mathrm{x} := \mathrm{z};\mathrm{y} := \mathrm{x}\|\mathrm{x} := 2\{y = 0\}} \text{ ✗}$$

Collect intermediate formulas

We modify all proof rules to collect intermediate formulas. For example,

$$[\textsc{Assign}]\frac{}{\{P[exp/x]\}\texttt{x := exp}\{P, \{P, P[exp/x]\}\}} \qquad [\textsc{Seq}]\frac{\{P\}c_1\{Q, \Sigma_1\} \quad \{Q\}c_2\{R, \Sigma_2\}}{\{P\}c_1; c_2\{R, \Sigma_1 \cup \Sigma_2\}}$$

Example 11.7

$$\frac{}{\{x > 1\}x := x - 1\{x > 0, \{x > 1, x > 0\}\}}$$

Exercise 11.1

Write collecting version of all the rules of Hoare logic.

Attempt 4: No interference on collected formulas (Sound, but incomplete)

$$[\text{ParNoModCollect}]\frac{\{P_1\}c_1\{Q_1,\Sigma_1\}\quad \{P_2\}c_2\{Q_2,\Sigma_2\}}{\{P_1 \land P_2\}c_1||c_2\{Q_1 \land Q_2, \Sigma_1 \cup \Sigma_2\}}\ \mathit{NoMod}(c_2,\Sigma_1) \text{ and } \mathit{NoMod}(c_1,\Sigma_2)$$

This proof rule is correct.
But too restrictive

Example 11.8

A good derivation:

$$\frac{\{x > 0\}y := z; \{x > 0, \{x > 0\}\} \qquad \{\top\}x := x + 1\{\top, \{\top\}\}}{\{x > 0\}y := z||x := x + 1\{x > 0, \{x > 0, \top\}\}}\ \textit{Rejected by the rule!}$$

Because $\mathit{NoMod}(x := x + 1, \{x > 0\})$ *is false.*

What went wrong? We went overboard. NoMod is a syntactic check.

Let us make *NoMod* false only if modifications really interfere.

Collect writes

Since only writes interfere, let us collect them explicitly.

We modify all proof rules to collect writes along with intermediate formulas. For example,

$$[\textsc{Assign}]\frac{}{\{P[exp/x]\}\texttt{x := exp}\{P, \{P, P[exp/x]\}, \{\texttt{x := exp}\}\}}$$

$$[\textsc{Seq}]\frac{\{P\}\mathtt{c}_1\{Q, \Sigma_1, Wrs_1\} \quad \{Q\}\mathtt{c}_2\{R, \Sigma_2, Wrs_2\}}{\{P\}\mathtt{c}_1;\mathtt{c}_2\{R, \Sigma_1 \cup \Sigma_2, Wrs_1 \cup Wars_2\}}$$

Example 11.9

$$\frac{}{\{x > 0\}y := z;\{x > 0, \{x > 0\}, \{y := z\}\}}$$

Exercise 11.2

Write collecting version of all the rules of Hoare logic.

Attempt 5: Semantic no interference condition (Still incomplete)

The following condition checks writes in Ws do not interfere invariants in Σ.

$$NoI(Ws, \Sigma) \triangleq \bigwedge_{c \in Ws} \bigwedge_{P \in \Sigma} \{P\}c\{P\} holds$$

$$[\textsc{ParNoInter}]\ \frac{\{P_1\}c_1\{Q_1, \Sigma_1, Ws_1\} \quad \{P_2\}c_2\{Q_2, \Sigma_2, Ws_2\}}{\{P_1 \wedge P_2\}c_1||c_2\{Q_1 \wedge Q_2, \Sigma_1 \cup \Sigma_2, Ws_1 \cup Ws_2\}} NoI(Ws_2, \Sigma_1) \text{ and } NoI(Ws_1, \Sigma_2)$$

Example 11.10

$$\frac{\{x > 0\}y := z; \{x > 0, \{x > 0\}, \{y := z\}\} \qquad \{\top\}x := x + 1\{\top, \{\top\}, \{x := x + 1\}\}}{\{x > 0\}y := z||x := x + 1\{x > 0, \{x > 0, \top\}, \{y := z, x := x + 1\}\}} \checkmark$$

Are we done?

Not quite.

Example 11.11

Consider the following correct derivation which is disallowed by [ParNoInter].

$$\frac{\{x > 10\}y := x; \{y > 10, \{x > 10, y > 10\}, \{y := x\}\} \qquad \{x > 20\}x := x - 1\{\top, \{x > 20\}, \{x := x - 1\}\}}{\{x > 20\}y := x||x := x - 1\{y > 10, \{...\}, \{...\}\}}$$

The derivation is not possible because

$NoI(\{x := x - 1\}, \{x > 10, y > 10\})$

$= \underbrace{\{x > 10\}x := x - 1\{x > 10\}}_{\text{Does not hold}}$ and $\{y > 10\}x := x - 1\{y > 10\} = \bot$

We are not complete. We are still rejecting good proofs.

How can we weaken our rule, while preserving soundness?

Collect writes with context

We modify [Assign] rule again to collect writes with their contexts. For example,

$$[\textsc{Assign}]\,\frac{}{\{P[exp/x]\}\texttt{x := exp}\{P, \{P, P[exp/x]\}, \{\ \{P[exp/x]\}\texttt{x := exp}\ \}\}}$$

We also need to modify [Havoc]. Rest remains the same.

Example 11.12

$$\frac{}{\{\mathrm{x} > 0\}\mathrm{y} := \mathrm{z}; \{\mathrm{x} > 0, \{\mathrm{x} > 0\}, \{\ \{\mathrm{x} > 0\}\mathrm{y} := \mathrm{z}\ \}\}}$$

Write with the condition under which it executes.

Attempt 6: Owicki-Gries proof rule (Sound and complete)

The following condition checks writes in Ws do not interfere invariants in Σ.

$$NoInter(Ws, \Sigma) \triangleq \bigwedge_{\{Q\}c \in Ws} \bigwedge_{P \in \Sigma} \{P \wedge Q\}c\{P\} holds$$

$$[\textsc{Par}] \frac{\{P_1\}c_1\{Q_1, \Sigma_1, Ws_1\} \quad \{P_2\}c_2\{Q_2, \Sigma_2, Ws_2\}}{\{P_1 \wedge P_2\}c_1||c_2\{Q_1 \wedge Q_2, \Sigma_1 \cup \Sigma_2, Ws_1 \cup Ws_2\}} NoInter(Ws_2, \Sigma_1) \text{ and } NoInter(Ws_1, \Sigma_2)$$

Example: interference checking with context

Example 11.13

$$\frac{\{x > 1\}y := x; \{y > 1, \{x > 1, y > 1\}, \{\{x > 1\}y := x\}\} \quad \{x{>}3\}x := x{-}1\{\top, \{x > 3\}, \{\{x > 3\}x := x{-}1\}\}}{\{x > 3\}y := x||x := x - 1\{y > 1, \{...\}, \{...\}\}}$$

The above derivation is acceptable by the PAR *rule because the side conditions are satisfied.*

$$\begin{aligned} \mathit{NoInter}(\mathit{Ws}_2, \Sigma_1) &= \mathit{NoInter}(\{\{x > 3\}x := x - 1\}, \{x > 1, y > 1\}) \\ &= \{x > 1 \land x > 3\}x := x - 1\{x > 1\} \textit{ and } \{y > 1 \land x > 3\}x := x - 1\{y > 1\} = \top \end{aligned}$$

Exercise 11.3

Show $\mathit{NoInter}(\mathit{Ws}_1, \Sigma_2)$ *is true.*

Commentary: Please check if this proof matches with earlier the proof-rule like notation.

Example: Let us prove a program

Let us prove.

$$\{x = 0\}x := x + 1 || x := x + 2\{x = 3\}$$

Let us display the Owicki-Gries proof in a more convenient notation

$$\{x = 0\}$$

$\{P_1 : x = 0 \lor x = 2\}$ $x := x + 1;$ $\{Q_1 : x = 1 \lor x = 3\}$	$\|\|$	$\{P_2 : x = 0 \lor x = 1\}$ $x := x + 2;$ $\{Q_2 : x = 2 \lor x = 3\}$

$$\{x = 3\}$$

Noninterference checks:

- $\{P_2 \land P_1\}x := x + 1\{P_2\}$
- $\{Q_2 \land P_1\}x := x + 1\{Q_2\}$
- $\{P_1 \land P_2\}x := x + 2\{P_1\}$
- $\{Q_1 \land P_2\}x := x + 2\{Q_1\}$

Exercise 11.4

a. Check $x = 0 \Rightarrow P_1 \land P_2$ *and* $Q_1 \land Q_2 \Rightarrow x = 3$. *b. Check noninterference checks.*

Example: let us prove one more

Let us suppose we need to prove.

$$\{x = 0\} x := x + 1 || x := x + 1 \{x = 2\}$$

Here is a Owicki-Gries proof.

$$\{x = 0 \land pc_1 = 0 \land pc_2 = 0\}$$

$$\begin{array}{l} \{pc_1 = 0 \land (pc_2 = 0 \Rightarrow x = 0) \land (pc_2 = 1 \Rightarrow x = 1)\} \\ x := x + 1; pc_1 := 1; \\ \{pc_1 = 1 \land (pc_2 = 0 \Rightarrow x = 1) \land (pc_2 = 1 \Rightarrow x = 2)\} \end{array} || \begin{array}{l} \{pc_2 = 0 \land (pc_1 = 0 \Rightarrow x = 0) \land (pc_1 = 1 \Rightarrow x = 1)\} \\ x := x + 1; pc_2 := 1; \\ \{pc_2 = 1 \land (pc_1 = 0 \Rightarrow x = 1) \land (pc_1 = 1 \Rightarrow x = 2)\} \end{array}$$

$$\{x = 2\}$$

Noninterference check remain the same. Please verify!

Locals may appear in the proof of the other thread.

Thread modular proofs

Definition 11.1

An Owicki-Gries proof is thread modular if the proof of a thread only refer to its locals and the globals.

Proofs are not thread modular, when globals lack information to describe the invariants.

Example 11.14

In a mutual exclusion protocol, if globals do not record who has the lock, then we need to refer to program counters of threads in the proofs.

Non-thread modular proofs tend to be cumbersome. As a principle, it is desirable to minimize reference to the locals of other threads.

Another example: proving victim mutual exclusion

$\{\top\}$

$\{P_1 : \top\}$
0 : $victim = 0$;
$\{Q_1 : (\mathtt{pc}_2 \neq 1 \Rightarrow victim = 0)\}$
1 : $\mathtt{while}(victim == 0)$;
$\{R_1 : \mathtt{pc}_1 = 2 \land \mathtt{pc}_2 = 1 \land victim = 1\}$
2 : //critical section

$\|$

$\{P_2 : \top\}$
0 : $victim = 1$;
$\{Q_2 : (\mathtt{pc}_1 \neq 1 \Rightarrow victim = 1)\}$
1 : $\mathtt{while}(victim == 1)$;
$\{R_2 : \mathtt{pc}_2 = 2 \land \mathtt{pc}_1 = 1 \land victim = 0\}$
2 : //critical section

$\{\bot\}$

Says both the threads cannot finish

Some noninterference checks for thread 1 invariants against thread 2 writes:

- No write can interfere with P_1, since it is $\top$.
- $\{Q_1 \land \top\}\mathtt{pc}_2 = 0 \land victim' = 1 \land \mathtt{pc}_2' = 1\{Q_1\}$
- $\{Q_1 \land (\mathtt{pc}_1 \neq 1 \Rightarrow victim = 1)\}\underbrace{\mathtt{pc}_2 = 1 \land victim = 0 \land victim = victim' \land \mathtt{pc}_2' = 2;}_{\text{Exit branch of the loop in the second thread}}\{Q_1\}$

Since exit from the loop modifies $\mathtt{pc}_2$, we need to check the formulas that mention it.

Exercise 11.5

a. Show R_1, P_2, Q_2, R_2 are free from interference.

b. How many noninterference checks are needed?

End of Lecture 11