CS 536 — Solution for Homework 3 (Errors, Nondeterminism, Hoare Triples)

10/17: pp.1, 2

1. Let \( \alpha = \sigma(x) \). Operationally, if \( \alpha \) is odd (say it equals \( 2\beta + 1 \)), then \[10/17\]
\[
(\text{if } x \text{ odd then } x := x - 1 \text{ fi}; x := x / 2, \sigma) \rightarrow (x := x - 1; x := x / 2, \sigma) \rightarrow^2 (E, x = \beta)
\]
If \( \alpha \) is even (say it equals \( 2\beta \)), then
\[
(\text{if } x \text{ odd then } x := x - 1 \text{ fi}; x := x / 2, \sigma) \rightarrow (x := x / 2, \sigma) \rightarrow (E, x = \beta)
\]
So if \( IF \equiv \) our if statement, if \( \sigma(x) = \alpha = 2\beta \) or \( 2\beta + 1 \), then \( \text{M}(IF, \sigma) = \{\{x = \beta\}\} \)

2. (Calculate meaning of loop) We're given \( W \equiv \text{while } x \neq 3 \text{ do } S \text{ od } \) and \( S \equiv x := x + 1; y := y + y \). To calculate \( \text{M}(W, \{x = 0, y = 1\}) \), we need to find all the final states the loop can terminate in (plus \( \perp_d \) if the loop can diverge). First let's look at the general semantics of the loop body \( S \). In an arbitrary state: Let \( \tau(x) = \alpha \) and \( \tau(y) = \beta \), then
\[
(S, \tau) = (x := x + 1; y := y + y, \tau) \rightarrow (y := y + y, \tau[x \mapsto \alpha + 1]) \rightarrow (E, \tau[x \mapsto \alpha + 1][y \mapsto 2\beta])
\]
Returning to our loop,
\[
(W, \sigma_0) \rightarrow (S; W, \sigma_0) \rightarrow^2 (W, \sigma_1) \rightarrow (S; W, \sigma_1) \rightarrow (W, \sigma_2) \rightarrow (S; W, \sigma_2) \rightarrow (W, \sigma_3) \rightarrow (E, \sigma_3)
\]
where \( \sigma_0 = \{x = 0, y = 1\} \)
\[ \text{because } \sigma_0(x \neq 3) = F \]
\[ \text{where } \sigma_1 = \sigma_0[x \mapsto \sigma_0(x) + 1][y \mapsto 2 \times \sigma_0(y)] = \{x = 1, y = 2\} \]
\[ \text{because } \sigma_1(x \neq 3) = F \]
\[ \text{where } \sigma_2 = \{x = 2, y = 4\} \]
\[ \text{because } \sigma_2(x \neq 3) = F \]
\[ \text{where } \sigma_3 = \{x = 3, y = 8\} \]
\[ \text{because } \sigma_3(x \neq 3) = T \]
So \( \text{M}(W, \sigma_0) = \{\sigma_3\} = \{\{x = 3, y = 8\}\} \).

3. (Discover an infinite loop)

We're to calculate \( \text{M}(W, \{x = 4, y = 1\}) \) for the same \( W \) as in the previous problem. Here, we find an infinite loop. Let \( \sigma_0 = \{x = 4, y = 1\}, \sigma_1 = \{x = 5, y = 2\}, \sigma_2 = \{x = 6, y = 4\} \) and in general let
\[
\sigma_{k+1} = \{x = \sigma_k(x) + 1, y = 2 \times \sigma_k(y)\}
\]
From the previous problem, we know \( (S, \sigma_k) = (E, \sigma_{k+1}) \), and since \( \sigma_k(x) \geq 4 \) for all \( k \), we know the loop won't halt. Specifically,
\[
(W, \sigma_0) \rightarrow^3 (W, \sigma_2) \rightarrow^3 (W, \sigma_2) \rightarrow^3 (W, \sigma_3) \rightarrow^3 \ldots
\]
So \( \text{M}(W, \sigma_0) = \{\perp_d\} \)

4. (Runtime errors) We have \( S \equiv v := b[x]; w := x / v; y := \sqrt{rt}(w) \) and \( \sigma = \{b = (3, 0, -2), x = \alpha\} \)

4a. If \( \sigma(x) = \alpha = -1 \), then \( \sigma(b[x]) = \perp_d \), since the array index \( x \) is -1, which is illegal. So
\[
(v := b[x]; w := x / v; y := \sqrt{rt}(w), \sigma) \rightarrow (w := x / v; y := \sqrt{rt}(w), \perp_d) \rightarrow (E, \perp_d)
\]
and thus \( \text{M}(S, \sigma[x \mapsto -1]) = \{\perp_d\} \). [Note: \( \sigma[x \mapsto -1] = \sigma \) here; I'm just writing it this way to emphasize the value of \( x \); you certainly didn't have to do that.]
4b. If σ(x) = α = 0, then \( v := b[x]; w := x/v; y := \sqrt{w}, σ \) \\
\( \rightarrow \langle w := x/v; y := \sqrt{w}, σ[v \mapsto 3] \rangle \) \\
\( \rightarrow \langle y := \sqrt{w}, σ[v \mapsto 3][y \mapsto 0] \rangle \) \[10/17\] \\
\( \rightarrow \langle E, σ[v \mapsto 3][x \mapsto 0][y \mapsto 0] \rangle \).

So \( M(S, σ[x \mapsto 0]) = \{σ[v \mapsto 3][x \mapsto 0][y \mapsto 0]\} \)

4c. If σ(x) = α = 1, then \( v := b[x]; w := x/v; y := \sqrt{w}, σ \) \\
\( \rightarrow \langle w := x/v; y := \sqrt{w}, σ[v \mapsto 0] \rangle \) \\
\( \rightarrow \langle E, \bot \rangle \) \hspace{1cm} \text{(evaluation of } x/v \text{ fails because } v \text{ is zero in } σ[v \mapsto 0]) \)

So \( M(S, σ[x \mapsto 1]) = \{\bot\} \).

5. (Similar nondeterministic IF statements)

- \( IF_1 \) and \( IF_2 \) always have the same semantics because it doesn’t matter what order the guarded commands are listed because it doesn’t matter in what order we put the statements to choose nondeterministically from. When comparing \( IF_1 \) and \( IF_2 \) against \( IF_3 \), there are three cases.

- (Case 1: \( B_1 \) and \( B_2 \) are both true) Then \( ¬B_1 ∧ ¬B_2 \), is false, so \( IF_3 \) chooses nondeterministically from \( \{S_1, S_2\} \), which means it has the same semantics as \( IF_1 \) and \( IF_2 \).

- (Case 2: \( B_1 \) is true and \( B_2 \) is false, or vice versa) Once again \( ¬B_1 ∧ ¬B_2 \), is false, so \( IF_3 \) chooses nondeterministically from \( \{S_1, S_2\} \), which means it has the same semantics as \( IF_1 \) and \( IF_2 \).

- (Case 3: \( B_1 \) and \( B_2 \) are both false) Here, \( IF_1 \) and \( IF_2 \) both cause a runtime error because none of their guards are satisfied. However, since \( ¬B_1 ∧ ¬B_2 \), satisfied, \( IF_1 \), executes the \text{skip} guarded by that test (and so it terminates without doing anything to the state).

6. Define the abbreviation \texttt{if } \( B \) \texttt{then } \( S \) \texttt{elif } \( B' \) \texttt{then } \( S' \) \texttt{elif } \( B'' \) \texttt{then } \( S'' \) \texttt{...else } \( S''' \) \texttt{fi} for the nested statement \texttt{if } \( B \) \texttt{then } \( S \) \texttt{else if } \( B' \) \texttt{then } \( S' \) \texttt{else if } \( B'' \) \texttt{then } \( S'' \) \texttt{...else } \( S''' \) \texttt{fi ... fi fi fi.}

To implement \texttt{if } \( B_1 \rightarrow S_1 \sqcup B_2 \rightarrow S_2 \) \texttt{fi}, we can use

\[
\text{if } B_1 ∧ B_2 \text{ then } \\
\quad \text{if } \text{T_or_F()} \text{ then } S_1 \text{ else } S_2 \text{ fi} \\
\quad \text{else if } B_1 \text{ then } S_1 \\
\quad \text{else if } B_2 \text{ then } S_2 \\
\quad \text{else error()} \text{ fi}
\]

(A brief justification: If both guards are true, then we want to nondeterministically execute one of \( S_1 \) and \( S_2 \). If only \( B_1 \) is true then we execute \( S_1 \) and similarly if only \( B_2 \) is true then we execute \( S_2 \). The semantics of nondeterministic \texttt{if–fi} says that if both guards are false, we get a runtime error.)
7. (Nondeterministic loop execution)

7a. We're given \( IF \equiv \begin{cases} \text{if } x \neq 0 \rightarrow x := x + 1; y := y + 1 \quad \text{if } x \neq 0 \rightarrow x := x - 1; y := y + 1 \end{cases} \) and an arbitrary state \( \tau \). The behavior of \( IF \) has two basic cases: If \( \tau(x) = 0 \), then \( \langle IF, \sigma \rangle \rightarrow \langle \bot_x, \sigma \rangle \) because neither of the guards are true. Otherwise, if \( \tau(x) \neq 0 \), then we have two possible execution paths:

\[
\langle IF, \sigma \rangle \rightarrow \langle x := x + 1; y := y + 1, \sigma \rangle \rightarrow^2 \langle E, \sigma[x \mapsto \sigma(x)+1][y \mapsto \sigma(y)+1] \rangle
\]

or

\[
\langle x := x - 1; y := y + 1, \sigma \rangle \rightarrow^2 \langle E, \sigma[x \mapsto \sigma(x)-1][y \mapsto \sigma(y)+1] \rangle
\]

7b. Operationally, for every \( N \geq 0 \), either the \( DO \) halts after \( M \) iterations (for some \( 0 \leq M \leq N \)), or it completes all \( N \) iterations without halting.

\[
\langle DO, \{x = 1, y = 0\} \rangle \rightarrow^N \langle DO, \{x = \alpha, y = N\} \rangle \text{ where } 0 \leq \alpha \leq N + 1
\]

where \( M \) \( \leq \) \( N \) for some \( \alpha \leq M \leq N \).

Since \( y \) starts at 0, after \( N \) iterations, \( y = N \). The value of \( x \) is somewhere between 0 and \( N+1 \) (the maximum \( N+1 \) is achieved if we do \( N \) increments from \( x = 1 \)).

When it comes to overall behavior, the loop can diverge, of course. To halt, it requires an odd number of iterations: If we start with \( x = 1 \) and do altogether say \( K \) increments, then we need \( K+1 \) decrements to get down to \( x = 0 \). Thus \( M(DO, \sigma_d) = \{ \{x = 0, y = N\} | N \in \{1, 3, 5, \ldots\} \} \cup \{ \bot_d \} \).

8. For a general statement \( S \), we know \( M(S, \sigma) \subseteq \Sigma_\tau \) (which = \( \Sigma = \{\bot_d, \bot_y\} \)). Since \( S \) is loop-free, \( \bot_d \) will not happen, so if \( S \) cannot cause an error, then \( \bot_y \) also will not happen. This leaves \( M(S, \sigma) \subseteq \Sigma \), so \( M(S, \sigma) \models q \) if and only if \( M(S, \sigma) - \{\bot\} \models q \), which implies that total and partial correctness of \( \{ p \} S \{ q \} \) are equivalent.

9. If \( M(S, \sigma) \not\subseteq \Sigma \), then \( \bot \in M(S, \sigma) \), so \( M(S, \sigma) \not\models q \). We know \( \sigma \models_{\text{tot}} \{ p \} S \{ \top \} \) means \( \sigma \models \neg p \) or \( M(S, \sigma) \models q \). (We know \( \sigma \) satisfies the precondition \( \top \).) Since \( M(S, \sigma) \not\models q \), we must have \( \sigma \models \neg p \).

10. (Can \( M(S, \sigma) \not\models_{\text{tot}} \{ p \} S \{ q \} \) and \( \not\models_{\text{tot}} \{ p \} S \{ \neg q \} \) simultaneously?)

a. If \( S \) is deterministic then \( M(S, \sigma) = \{ \tau \} \subseteq \Sigma_\tau \). If \( \tau \neq \bot \), either \( \tau \models q \) or \( \tau \models \neg q \). It follows that \( \sigma \models_{\text{tot}} \{ p \} S \{ q \} \) or \( \sigma \models_{\text{tot}} \{ p \} S \{ \neg q \} \) respectively. So to get \( \not\models_{\text{tot}} \) for both triples, then, we need \( \tau = \bot \).

b. If \( S \) is nondeterministic but always halts then \( M(S, \sigma) \subseteq \Sigma \). If \( M(S, \sigma) \models q \), then \( \sigma \models_{\text{tot}} \{ p \} S \{ q \} \).

Similarly, if \( M(S, \sigma) \models \neg q \), then \( \sigma \models_{\text{tot}} \{ p \} S \{ \neg q \} \). To get \( \not\models_{\text{tot}} \) for both triples, we need \( M(S, \sigma) \not\models q \) and \( \not\models \neg q \) simultaneously. This happens iff at least one state in \( M(S, \sigma) \) satisfies \( q \) and at least one state in \( M(S, \sigma) \) satisfies \( \neg q \).

11. (Does \( \sigma \models \{ p \} S \{ q \} \) imply \( \sigma \models p \) ? \( \sigma \models \neg p \)?) Since \( \sigma \models \{ p \} S \{ q \} \) iff \( \sigma \models \neg p \) or \( M(S, \sigma) - \bot \models q \), we find that \( \sigma \models \{ p \} S \{ q \} \) is compatible with both \( \sigma \models p \) and \( \sigma \models \neg p \) and implies neither of them.

12. (Does \( \sigma \not\models \{ p \} S \{ q \} \) imply \( \sigma \models \{ p \} S \{ \neg q \} \) or \( \sigma \not\models \{ p \} S \{ \neg q \} \)?) From \( \sigma \not\models \{ p \} S \{ q \} \), we know \( \sigma \models p \) and \( M(S, \sigma) \subseteq \Sigma \) and \( M(S, \sigma) \not\models q \). Since \( \sigma \models p \), to show \( \sigma \models \{ p \} S \{ \neg q \} \), we need \( M(S, \sigma) \models \neg q \).

(Case 1) If \( M(S, \sigma) = \{ \tau \} \) for some \( \tau \in \Sigma \), (i.e., \( S \) is deterministic or just happens to always terminate in one state), then \( M(S, \sigma) \not\models q \) implies \( M(S, \sigma) \models \neg q \). It follows that \( \sigma \models \{ p \} S \{ \neg q \} \).

(Case 2) If \( S \) is nondeterministic and can terminate in more than one state, then \( M(S, \sigma) \not\models q \) is compatible with both \( M(S, \sigma) \models \neg q \), so \( \sigma \not\models \{ p \} S \{ q \} \) implies neither \( \sigma \models \) or \( \not\models \{ p \} S \{ \neg q \} \).
13. (Consequences of $\sigma \not\models \{ p \} S \{ q \})$  Failure of partial correctness implies
   a. $\sigma \models p$
   b. $M(S, \sigma) \subseteq \Sigma$, and
   c. $M(S, \sigma) \not\models q$
   d. If $M(S, \sigma) = \{ \tau \}$, then $M(S, \sigma) \subseteq \Sigma$ implies $\tau \in \Sigma$, so $M(S, \sigma) \not\models q$ implies $M(S, \sigma) \not\models \neg q$. If $M(S, \sigma)$ has more than one element, then $M(S, \sigma) \not\models q$ implies neither $M(S, \sigma) \models \neg q$ nor $\not\models \neg q$.

14. (Consequences of $\sigma \models p$ and $\sigma \not\models \{ p \} S \{ q \}$)  Partial correctness tells us that $M(S, \sigma) - \bot \models q$. Therefore,
   a. Both $M(S, \sigma) \subseteq \Sigma$ and $\not\subseteq \Sigma$ are possible.
   b. Both $M(S, \sigma) \models q$ and $\not\models q$ are possible: $M(S, \sigma) \models q$ if $\bot \not\in M(S, \sigma)$ but $M(S, \sigma) \not\models q$ if $\bot \in M(S, \sigma)$.
   c. Only $M(S, \sigma) \not\models q$ is possible: If $\bot \in M(S, \sigma)$ then $M(S, \sigma) \not\models q$, and if $\bot \not\in M(S, \sigma)$ then $M(S, \sigma) \models q$, in which case $M(S, \sigma) \not\models q$.

15. (Consequences of $\sigma \models p$ and $\sigma \not\models \{ p \} S \{ q \}$.)  Total correctness implies $M(S, \sigma) \models q$. Therefore,
   a. $M(S, \sigma) \subseteq \Sigma$ because $M(S, \sigma) \models q$ implies $\bot \not\in M(S, \sigma)$.
   b. $M(S, \sigma) \models q$ holds.
   c. $M(S, \sigma) \not\models \neg q$ because $M(S, \sigma) \models q$.

16. (Consequences of $\sigma \not\models \{ p \} S \{ q \}$)  From $\sigma \not\models \{ p \} S \{ q \}$ we know $\sigma \models p$ and $M(S, \sigma) \not\models q$, so
   a. $M(S, \sigma) \not\models q$ implies neither $M(S, \sigma) \subseteq \Sigma$ or $\not\subseteq \Sigma$.
   b. $M(S, \sigma) \not\models q$ holds.
   c. $M(S, \sigma) \models \neg q$ if $M(S, \sigma)$ is a singleton set, otherwise both $M(S, \sigma) \models \neg q$ and $\not\models \neg q$ are possible.
   d. Both $M(S, \sigma) - \bot \models q$ are possible $\not\models q$, so $\sigma \models$ and $\not\models \{ p \} S \{ \neg q \}$ are both possible.

17. (Consequences of $\sigma \models \{ p \} S \{ q \}$ and $\sigma \not\models \{ p \} S \{ q \}$)
   Lack of total correctness tells us that $\sigma \models p$ and $M(S, \sigma) \not\models q$. On the other hand, partial correctness of the triple tells us that since $\sigma \models p$, we know $M(S, \sigma) - \bot \models q$. So we must have $\bot \in M(S, \sigma)$.
   a. $M(S, \sigma) \not\subseteq \Sigma$ because $\bot \in M(S, \sigma)$.
   b. $M(S, \sigma) \not\models q$ because of lack of total correctness.
   c. $M(S, \sigma) \not\models \neg q$ because $\bot \in M(S, \sigma)$. 