CS 536 — Solution for Homework 2 (Satisfaction, Validity, State Updates, Programs, Operational Semantics)

1. (State updates. Below, \(\sigma = \{ x = 2, y = 4, b = \gamma \} \) where \(\gamma = (11, 21, 31, 41)\).
   a. \(\sigma[x \mapsto 8][x \mapsto 5] = \{ x = 2, y = 4, b = 8 \} \) \(
      \{ x = 2, y = 4, b = 5 \} \) \(= \{ x = 5, y = 4, b = \gamma \}\)
   b. \(\sigma[y \mapsto 5](x) = \{ x = 2, y = 4, b = \gamma \} \) \(= \{ x = 2, y = 5, b = \gamma \}(x) = 2\)
   c. For \(\sigma[b[1] \mapsto 13] \) \(\sigma(\sigma[b[1] / 2])\), note that we're taking the value of \(b[1] / 2\) in \(\sigma\), not \(\sigma[b[1] \mapsto 13]\).
      To get the value of \(b[1] / 2\) in that state, can write \(\sigma[b[1] \mapsto 13](b[1] / 2)\).
      Let \(\gamma = \sigma(b) = (11, 21, 31, 41)\) so that \(\sigma(b[1] / 2) = \sigma(b)(1) / 2 = \gamma(1) / 2 = 21/2 = 10^1\). Writing tons of detail (you didn't have to be this explicit),
      \(\sigma[b[1] \mapsto 13][y \mapsto \sigma(b[1] / 2)] = \sigma[b[1] \mapsto 13][y \mapsto 10]\)
      \(= \{ x = 2, y = 4, b = \gamma \} \) \(\{ x = 2, y = 10, b = (11, 13, 31, 41)\}\)
   d. (When does \(\sigma[u \mapsto \alpha] \) \(\sigma[v \mapsto \beta] = \sigma[v \mapsto \beta][u \mapsto \alpha] \) ?)
      Case 1 \((u \equiv v)\) On the left-hand side, we find \(\sigma[u \mapsto \alpha][v \mapsto \beta] = \sigma[v \mapsto \beta]\). On the right-hand side, we find
      \(\sigma[v \mapsto \beta][u \mapsto \alpha] = \sigma[u \mapsto \alpha]\) (since \(u \equiv v\)). The two sides are \(\equiv\) when \(\sigma[v \mapsto \beta] = \sigma[v \mapsto \alpha]\),
      which happens when \(\alpha = \beta\).
      Case 2 \((u \not\equiv v)\) In this case, the two sides are equal: updating \(v\) after updating \(u\) won't affect the binding for \(u\)
      and vice versa, so it doesn't matter whether we update \(u\) then \(v\) or \(v\) then \(u\).
      Summary: The two are equal when \((u \equiv v \) and \(\alpha = \beta)\) or \((u \not\equiv v)\). Some logically equivalent ways to write this
      are \((u \equiv v \mapsto \alpha = \beta)\) and \((u \not\equiv v \mapsto \alpha = \beta)\).

2. (Satisfaction of predicates) We're given \(\sigma(x) = 2\) and \(\sigma(y)\) is undefined.
   a. \(\sigma[y \mapsto 5] \models \exists x. x > y \) with 6 as one possible witness: \(\sigma[y \mapsto 5] \models \exists x. x > y \) if \(\sigma[y \mapsto 5][x \mapsto 6] \models x > y\) if \(x > 6\), which is true.
   b. \(\sigma[x \mapsto 3] \models (\forall y. y \geq 2 \rightarrow x < y^2) \) iff for all \(\alpha \in \mathbb{Z}\), \(\sigma[x \mapsto 3][y \mapsto \alpha] \models y \geq 2 \rightarrow x < y^2\), which holds
      iff \((\alpha \geq 2 \) implies \(3 \leq \alpha^2\), which is true. Note \(\sigma[x \mapsto 3]\) is proper for \((\forall y. y \geq 2 \rightarrow x < y^2)\) because
      the status of \(\sigma(y)\) doesn't matter when we ask "Does \(\sigma[x \mapsto 3][y \mapsto \alpha] \models \ldots?\)"
   c. State \(\sigma\) is not proper for \(0 \times x = 0 \times y\) because it has no binding for \(y\).
   d. \(\sigma[z \mapsto 5][y \mapsto 3][z \mapsto 6] \models z = x \times y\) is true: The second update of \(z\) overwrites the first update, so in
      effect, we have \(\sigma[y \mapsto 3][z \mapsto 6]\). The final state is \(\{ x = 2 \} [y \mapsto 3][z \mapsto 6] = \{ x = 2, y = 3, z = 6 \}\),
      which does \(\models z = x \times y\).
3. (Updated states) We’re given \( \sigma(x) = 2 \) and \( \sigma \) may be undefined for any variable \( \neq x \).
   a. \[ \sigma[z \mapsto 4] = \sigma \cup \{z = 4\} \] — This is legal if \( \sigma(z) \) is undefined; our assumption allows this but doesn’t require it.
   b. The state \( \sigma[y \mapsto 0][b \mapsto (1, 3, x+y)] \) is not legal: \( x+y \) is a syntactic value and needs to be replaced with a semantic value like \( \sigma(x+y) \).
   c. \( \sigma[v \mapsto 5](x) \) is undefined — This is false: Since \( v \neq x \), \( \sigma[v \mapsto 5](x) = \sigma(x) = 2 \).

4. (Satisfaction and validity of quantified predicates)
   a. \( \not\exists (\exists x \in U \cdot p) \) iff for this \( \sigma \) and every \( \alpha \in U, \sigma[x \mapsto \alpha] \not \in p \).
   b. \( \exists (\exists x \in U \cdot p) \) iff for some \( \sigma \) and every \( \alpha \in U, \sigma[x \mapsto \alpha] \vDash p \).
   c. \( \vDash (\exists x \in U \cdot (\forall y \in V \cdot p)) \) iff for every \( \sigma \in U \), some \( \alpha \in U \), and every \( \beta \), \( \sigma[x \mapsto \alpha] [y \mapsto \beta] \vDash p \).
   d. \( \not\exists (\exists x \in U \cdot (\forall y \in V \cdot p)) \) iff for some \( \sigma \), every \( \alpha \in U \), and some \( \beta \in U \), \( \sigma[x \mapsto \alpha] [y \mapsto \beta] \not \vDash p \).
   e. \( \not\exists (\forall x \in U \cdot (\exists y \in V \cdot p)) \) iff for some \( \sigma \), some \( \alpha \in U \), and every \( \beta \in U \), \( \sigma[x \mapsto \alpha] [y \mapsto \beta] \not \vDash p \).

5. (Relations involving \( \exists \exists, \forall \forall, \exists \forall \), and \( \forall \exists \) predicates)
   a. \( \exists x, \exists y \cdot p(x, y) \Leftrightarrow \exists y, \exists x \cdot p(x, y) \), always
   b. \( \forall x, \forall y \cdot p(x, y) \Leftrightarrow \forall y, \forall x \cdot p(x, y) \), always
   c. \( \forall x, \exists y \cdot p(x, y) \Leftrightarrow \exists y, \forall x \cdot p(x, y) \), in general. (\( \Leftrightarrow \) holds in some cases but not in all cases)

6. (Translate from C)
   a. \( j := m; x := 1; \text{while } j >= 1 \text{ do } j := j - 1; x := x*y[j] \text{ od} \)
   b. \( j := m; x := 1; j := j - 1; \text{while } j >= 0 \text{ do } x := x*y[j]; j := j - 1 \text{ od} \)
   c. \( x := 1; j := 0; \text{while } j < m \text{ do } j := j + 1; x := x*y[j] \text{ od}; j := j + 1 \)

7. (Examples of operational execution)
   a. (Here’s a very detailed solution.) Let \( \sigma = \{(x, \alpha), (y, \beta)\} \) and \( S \equiv t := x; x := y; y := t \), then
      \[ \langle S, \sigma \rangle = \langle t := x; x := y; y := t, \sigma \rangle \]
      \[ \rightarrow \langle x := y; y := t, \sigma[t \mapsto \sigma(x)] \rangle \]
      \[ \rightarrow \langle x := y; y := t, \sigma[t \mapsto \alpha] \rangle \quad \text{since } \sigma(x) = \alpha \quad \text{(Note it’s } "=\text{" here, not } \rightarrow\text{")} \]
      \[ \rightarrow \langle y := t, \sigma_1[x \mapsto \sigma_1(y)] \rangle \quad \text{where } \sigma_1 = \sigma[t \mapsto \alpha] \]
      \[ \rightarrow \langle y := t, \sigma_1[x \mapsto \beta] \rangle \quad \text{since } \sigma_1(y) = \sigma[t \mapsto \alpha](y) = \sigma(y) = \beta \]
      \[ \rightarrow \langle y := t, \sigma_2 \rangle \quad \text{where } \sigma_2 = \sigma_1[x \mapsto \beta] = \sigma[t \mapsto \alpha][x \mapsto \beta] \]
      \[ \rightarrow \langle E, \sigma_2[y \mapsto \sigma_2(t)] \rangle \]
      \[ \rightarrow \langle E, \sigma_2[y \mapsto \alpha] \rangle \quad \text{since } \sigma_2(t) = \sigma_1[x \mapsto \beta][t] = \sigma_1(t) = \sigma[t \mapsto \alpha](t) = \alpha \]
   b. (This solution has less detail.) Let \( \sigma = \{(x, \alpha), (y, \beta)\} \) and \( S \equiv \text{if } x < 0 \text{ then } y := -x \text{ else } y := x \fi \). If \( \alpha < 0 \), then \( \langle S, \sigma \rangle \rightarrow \langle y := -x, \sigma \rangle \rightarrow \langle E, \sigma[y \mapsto -\alpha] \rangle \). Otherwise, \( \alpha \geq 0 \) so \( \langle S, \sigma \rangle \rightarrow \langle y := x, \sigma \rangle \rightarrow \langle E, \sigma[y \mapsto \alpha] \rangle \).
8. (Operational semantics of loops) Let $S \equiv s := 0$; $W$ where $W \equiv \text{while } n \geq 0 \text{ do } S_1 \text{ od}$ and $S_1 \equiv s := s + n$; $n := n - 1$.

a. We’re given $\sigma(n) < 0$. Then $\langle S, \sigma \rangle = \langle s := 0; W, \sigma \rangle \rightarrow \langle W, \sigma[s \mapsto 0] \rangle = \langle \text{while } n \geq 0 \ldots, \sigma[s \mapsto 0] \rangle \rightarrow \langle E, \sigma[s \mapsto 0] \rangle$.

b. For the loop body $S_1$ in an arbitrary state $\tau[s \mapsto \alpha][n \mapsto \beta]$, we have

$\langle S_1, \tau[s \mapsto \alpha][n \mapsto \beta] \rangle$

$= \langle s := s + n; n := n - 1, \tau[s \mapsto \alpha][n \mapsto \beta] \rangle$

$\rightarrow \langle n := n - 1, \tau[n \mapsto \beta][s \mapsto \alpha + \beta] \rangle$

$\rightarrow \langle E, \tau[s \mapsto \alpha + \beta][n \mapsto \beta - 1] \rangle$

c. If $\sigma(n) = 3$, then

$\langle S, \sigma \rangle \rightarrow \langle W, \sigma_0 \rangle$ where $\sigma_0 = \sigma[s \mapsto 0][n \mapsto 3]$

$\rightarrow \langle S_1; W, \sigma_0 \rangle$ bec. $\sigma_0(n \geq 0) = (3 \geq 0) = T$

$\rightarrow^2 \langle W, \sigma_1 \rangle$ where $\sigma_1 = \sigma_0[s \mapsto 0 + 3][n \mapsto 3 - 1] = \sigma[s \mapsto 3][n \mapsto 2]$

$\rightarrow \langle S_1; W, \sigma_1 \rangle$ bec. $\sigma_1(n \geq 0) = (2 \geq 0) = T$

$\rightarrow^2 \langle W, \sigma_2 \rangle$ where $\sigma_2 = \sigma_1[s \mapsto 3 + 2][n \mapsto 2 - 1] = \sigma[s \mapsto 5][n \mapsto 1]$

$\rightarrow \langle S_1; W, \sigma_2 \rangle$ bec. $\sigma_2(n \geq 0) = (1 \geq 0) = T$

$\rightarrow^2 \langle W, \sigma_3 \rangle$ where $\sigma_3 = \sigma_2[s \mapsto 5 + 1][n \mapsto 1 - 1] = \sigma[s \mapsto 6][n \mapsto 0]$

$\rightarrow \langle S_1; W, \sigma_3 \rangle$ bec. $\sigma_3(n \geq 0) = (0 \geq 0) = T$

$\rightarrow^2 \langle W, \sigma_4 \rangle$ where $\sigma_4 = \sigma_3[s \mapsto 6 + 0][n \mapsto 0 - 1] = \sigma[s \mapsto 6][n \mapsto -1]$

$\rightarrow \langle E, \sigma_4 \rangle$ bec. $\sigma_4(n \geq 0) = (-1 \geq 0) = F$

A shorter version of this is

$\langle S, \sigma \rangle \rightarrow \langle W, \sigma_0 \rangle$ where $\sigma_0 = \sigma[s \mapsto 0][n \mapsto 3]$

$\rightarrow^3 \langle W, \sigma_1 \rangle$ where $\sigma_1 = \sigma[s \mapsto 3][n \mapsto 2]$

$\rightarrow^3 \langle W, \sigma_2 \rangle$ where $\sigma_2 = \sigma[s \mapsto 5][n \mapsto 1]$

$\rightarrow^3 \langle W, \sigma_3 \rangle$ where $\sigma_3 = \sigma[s \mapsto 6][n \mapsto 0]$

$\rightarrow^3 \langle W, \sigma_4 \rangle$ where $\sigma_4 = \sigma[s \mapsto 6][n \mapsto -1]$

$\rightarrow \langle E, \sigma_4 \rangle$ bec. $\sigma_4(n \geq 0) = F$

In both versions above, it’s also okay to write $\rightarrow^k$ instead of $\rightarrow^2$ or $\rightarrow^3$.

9. a. $b[i..j] > x \equiv 0 \leq i, j < \text{size}(b) \land \forall k. i \leq k < j \rightarrow b[i] > x$.†

b. $\text{split}(b, m, p, n) \equiv (0 \leq m < n \leq \text{size}(b)) \land b[0..m) \leq b[m] \land b[p..n) \geq b[m]$. You’ll often see people extend the notation and write things like $b[0..m) \leq b[m] \leq b[p..n)$.

---

† By $0 \leq i, j < \text{size}(b)$, I mean $(0 \leq i < \text{size}(b)) \land (0 \leq j < \text{size}(b))$. 

CS 536: Science of Programming – 3 – © James Sasaki, 2018