Satisfaction, Validity, and State Updates

CS 536: Science of Programming, Fall 2021

1. Say u and v stand for variables (possibly the same variable) and α and β are values (possibly equal). When is σ[u ↦ α][v ↦ β] = σ[v ↦ β][u ↦ α]? Hint: There are four cases because maybe u ≡ v and maybe α = β.

2. Let σ(*b*) = (7, 5, 12, 16). Assume out-of-bound indexes cause runtime errors.

   a. Does σ ⊨ ∃*k* . *0* ≤ *k* ∧ *k+1* < *size*(*b*) ∧ *b*[*k*] < *b*[*k+1*]? If so, what was your witness value for *k*?

   b. Does σ ⊨ ∃*k* . *0* ≤ *k-1* ∧ *k+1* < *size*(*b*) ∧ *b*[*k-1*] < *b*[*k*] < *b*[*k+1*]? If so, what was your witness value for *k*?

   c. Does σ ⊨ ∀*k* . *0* ≤ *k* < *4* → *b*[*k*] > *0*?

   d. If σ(*k*) = -5, then does σ ⊨ ∃*k* . *b*[*k*] > *0*?

3. For each of the situations below, fill in the blanks to describe when the situation holds.

   > Fill in _____ ₁ with "*some*", "*every*", or "*this*"
   > Fill in _____ ₂ with "*some*" or "*every*"
   > Fill in _____ ₃ with "σ(*x*) must *be undefined*", "σ(*x*) must *be defined and* σ ⊨ *p*", or "*nothing of* σ(*x*)"
   > Fill in _____ ₄ with "⊨ *p*" or "⊭ *p*"

   a. σ ⊨ (∃ *x* ∈ *U*. *p*) iff for _____ ₁ state σ and _____ ₂ α ∈ *U*, σ[*x* ↦ α] _____ ₄

   b. σ ⊨ (∀ *x* ∈ *U*. *p*) iff for _____ ₁ state σ and _____ ₂ α ∈ *U*, σ[*x* ↦ α] _____ ₄

   c. σ ⊨ (∃ *x* ∈ *U*. *p*) requires _____ ₃ .

   d. σ ⊨ (∀ *x* ∈ *U*. *p*) requires _____ ₃ .

   e. σ ⊭ (∃ *x* ∈ *U*. *p*) iff for _____ ₁ state σ for _____ ₂ α ∈ *U*, σ[*x* ↦ α] _____ ₄

   f. σ ⊭ (∀ *x* ∈ *U*. *p*) iff for _____ ₁ state σ for _____ ₂ α ∈ *U*, σ[*x* ↦ α] _____ ₄

   g. ⊭ (∀ *x* ∈ *U*. *p*) iff for _____ ₂ state σ, we have σ _____ ₄ (∀ *x* ∈ *U*. *p*).

   h. ⊭ (∃ *x* ∈ *U*. *p*) iff for _____ ₂ state σ, we have σ _____ ₄ (∃ *x* ∈ *U*. *p*).

   i. ⊭ (∀ *x* ∈ *U*. *p*) iff for _____ ₂ state σ, and for _____ ₂ α ∈ *U*, we have σ[*x* ↦ α] _____ ₄

   j. ⊨ (∃ *x* ∈ *U* . (∀ *y* ∈ *V* . *p*)) iff for _____ ₁ state σ, for _____ ₂ α ∈ *U*, and for _____ ₂ β ∈ *V*, we have σ[*x* ↦ α][*y* ↦ β] _____ ₄

   k. ⊭ (∃ *x* ∈ *U* . (∀ *y* ∈ *V* . *p*)) iff for _____ ₁ state σ, for _____ ₂ α ∈ *U*, and for _____ ₂ β ∈ *V*, we have σ[*x* ↦ α][*y* ↦ β] [⊨ | ⊨ ¬] *p*.

l. ⊨ (∀ x ∈ U . (∃ y ∈ V . p)) iff for _____ ₁ state σ, for _____ ₂ α ∈ U, and for _____ ₂ β ∈ V, we have σ[x ↦ α][y ↦ β] [⊨ | ⊨ ¬] p.

m. ⊭ (∀ x ∈ U . (∃ y ∈ V . p)) iff for _____ ₁ state σ, for _____ ₂ α ∈ U, and for _____ ₂ β ∈ V, we have σ[x ↦ α][y ↦ β] _____ ₄

4. Let $p \equiv \exists y . \forall x . f(x) > y$, and let $q \equiv \forall x . \exists y . f(x) > y$. (As usual, assume a domain of $\mathbb{Z}$.)

   a. Is it the case that (regardless of the definition of f), if p is valid then so is q? If so, explain why. If not, give a definition of $f(x)$ and show ⊨ p but ⊭ q.

   b. (The converse.) Is it the case that (regardless of the definition of f), if q is valid then so is p? If so, explain why. If not, give a definition of $f(x)$ and show ⊨ q but ⊭ p.

CS 536: Solution to Activity 4 (Satisfaction, Validity, and State Updates)

1. $\sigma[u \mapsto \alpha][v \mapsto \beta] = \sigma[v \mapsto \beta][u \mapsto \alpha]$ iff $u \not\equiv v$ or ($u \equiv v$ and) $\alpha = \beta$. Another way to phrase this is ($\alpha = \beta$ or $u \not\equiv v$)

2. (Quantified statements over arrays) Let $\sigma(b) = (7, 5, 12, 16)$.

   a. Yes, $\sigma \models \exists k . 0 \leq k \wedge k{+}1 < size(b) \wedge b[k] < b[k{+}1]$ with 1 and 2 as possible witnesses for k.

   b. Yes, $\sigma \models \exists k . 0 \leq k{-}1 \wedge k{+}1 < size(b) \wedge b[k{-}1] < b[k] < b[k{+}1]$ with 2 as the only witness that works.

   c. Yes, $\sigma \models \forall k . b[k] > 0$

   d. Yes, if $\sigma(k) = -5$, we still have $\sigma \models \exists k . b[k] > 0$, with witnesses 0, 1, 2, 3. The key is that for σ to satisfy the existential with witness call it α, then we need $\sigma[k \mapsto \alpha] \models b[k] > 0$, which doesn't depend on $\sigma(k)$ because the update of σ uses $k = \alpha$, not $k = $ whatever $\sigma(k)$ happens to be. Here's a step-by-step explanation (this is way too much detail for appearing on a test):

$\sigma[k \mapsto \alpha] \models b[k] >$	
iff $\sigma[k \mapsto \alpha](b[k]) > \sigma[k \mapsto \alpha](0)$	defn state $\models$ relational test
iff $(\sigma[k \mapsto \alpha](b))(\sigma[k \mapsto \alpha](k)) > 0$	the value of 0 is zero
iff $(\sigma(b))(\sigma[k \mapsto \alpha](k)) > 0$	$\sigma[k \mapsto \alpha](b) = \sigma(b)$ because $b \not\equiv k$
iff $(\sigma(b))(\alpha) \quad > 0$	$\sigma[k \mapsto \alpha](k)) = \alpha$
iff 7, 5, 12, or 16 > 0	depending on $\alpha = 0, 1, 2,$ or 3

3. (Validity/invalidity of quantified predicates)
   a. this σ, some α, $\models p$
   b. this σ, every α, $\models p$
   c. nothing of $\sigma(x)$
   d. nothing of $\sigma(x)$
   e. this σ, every α, $\not\models p$
   f. this σ, some α, $\not\models p$
   g. some σ, $\not\models \forall x \in U. p$
   h. some σ, every α, $\not\models p$
   i. some σ, some α, $\not\models p$
   j. every σ, some α, every β, $\models p$
   k. some σ, every α, some β, $\not\models p$
   l. every σ, every α, some β, $\models p$
   m. some σ, some α, every β, $\not\models p$

4. (∃∀ predicates versus ∀∃ predicates, specifically $p \equiv \exists y . \forall x . f(x) > y$, and $q \equiv \forall x . \exists y . f(x) > y$)

    a. The relation does hold: $\models p$ implies $\models q$. The short explanation is that for each value α for x, we need to find a value β for y that satisfies the body, but p says that there's a value that works for every α, so we can use that value for β. In more detail, assume p is valid: for every state σ, there is some value β where for every value α, σ[$y \mapsto β$][$x \mapsto α$] $\models f(x) > y$. To show that q is valid, take an arbitrary state τ with value α for x. We need a witness value for the ∃y; using p with τ for σ, we get a β for the ∃y of p and use that as the witness for the ∃y in q. So then we need τ[$x \mapsto α$][$y \mapsto β$] $\models f(x) > y$. Substituting σ for τ and swapping the order of the updates, we need σ[$y \mapsto β$][$x \mapsto α$] $\models f(x) > y$. But that's exactly what p provided.

    b. The relation does not hold: We can have $\models q$ but $\not\models p$. The easiest example is $f(x) = x$, then validity of p would require us to find an integer value for y that is > every possible integer value of x, but no such value exists.

        As an aside, you can use an arbitrary predicate over x and y instead of $f(x) > y$ as the body of the ∃∀ and ∀∃ predicates. I use $f(x) > y$ here just because it's nice and concrete.