

Satisfaction, Validity, and State Updates

CS 536 Lecture 4, Mon Jan 23, 2012

version Mon, Mar 5, 2012

A. Why

- A predicate is satisfied or unsatisfied relative to a state.
- A predicate is valid if it is satisfied in all states.
- State updates occur when we introduce new variables or change the values of existing variables.

B. Outcomes

At the end of today, you should

- Know what it means for a predicate to be satisfied in a state or valid.
- Know what it means to update a state.

C. New and Improved Notation for States

- I thought it over and decided to avoid the same notation for predicates and states.
- But writing all those pairs is annoying, so let's try this:
 - **Notation:** For a binding in a state, writing $v = \alpha$ (where v is a variable and α is a value) means the pair (v, α) . E.g., $\{x = 1, y = 2, b = (3, 4, 5)\}$ means $\{(x, 1), (y, 2), (b, (3, 4, 5))\}$.
 - States are still sets. E.g., $\{x = 0, y = 2\} = \{y = 2, x = 0\} = \{x = 0\} \cup \{y = 2\}$, etc.
- **Notation:** $\sigma(v) = \alpha$ means that the binding $v = \alpha$ is a member of σ . If v has no binding in σ , then $\sigma(v)$ is undefined.

D. Truth (Satisfaction and Validity) of Predicates

- **Definition:** A predicate p is **satisfied in state** σ if it evaluates to true in state σ . (Note σ must be proper for p .) We also say σ **satisfies** p .
 - E.g., $x = y \wedge y = 0$ is satisfied in $\{x = 0, y = 0\}$; it's not satisfied in $\{x = 0, y = 2\}$.
- **Notation:** $\sigma \models p$ means σ satisfies p . (The " \models " symbol is a "double turnstile".) Similarly, $\sigma \not\models p$ means σ (is proper but) does not satisfy p .
- One reason to be interested in satisfaction is that our program specifications will only tell us things about state/program pairs where the state initially satisfies some predicate. E.g., "**if** (we're in a state satisfying) $x > 0$, then after assigning $x-1$ to y , we know (that we'll be in a state satisfying) $y \geq 0$ ".
- **Definition:** A predicate p is **valid** if it is satisfied in all states: For all σ , $\sigma \models p$.
 - E.g., (assuming x ranges over integers) $x = x+0$ is valid. So is $\forall x : x > x+1$.
 - **Notation:** $\models p$ means p is valid.

- **Note:** If p is not valid, that means there are one or more states in which it is not satisfied. That's different from saying it's not satisfied in any state. So $x^2 > x$ is not valid because it's not satisfied in $\{x = 0\}$. (Other states that are counterexamples are $\{x = 1\}$ and $\{x = -1\}$.)
- For **non-quantified predicates**, satisfaction is straightforward:
 - For $\sigma \models e_1 < e_2$, we need $\sigma(e_1)$ less than $\sigma(e_2)$. Other comparisons are similar.
 - $\sigma \models T$ always; $\sigma \not\models F$ always.
 - $\sigma \models \neg p$ means $\sigma \not\models p$.
 - $\sigma \models p \wedge q$ means $\sigma \models p$ and $\sigma \models q$
 - $\sigma \models p \vee q$ means $\sigma \models p$ or $\sigma \models q$ (or both).
 - $\sigma \models p \rightarrow q$ means $\sigma \models \neg p$ or $\sigma \models q$ (or both).
 - $\sigma \models p \leftrightarrow q$ means $\sigma \models p$ and $\sigma \models q$, or, $\sigma \models \neg p$ and $\sigma \models \neg q$.

E. Updated States

- For quantified predicates, we need to look at different modifications of the state.
 - For $\sigma \models \exists v. p$, we need a "**witness**" value such that σ satisfies p when we add a binding for v to that value. E.g., $\{y = 1\} \models \exists x. x^2 < y$ using 0 as the witness for x : $\{y = 1, x = 0\} \models x^2 < y$. (For this value of 1 there is only the one possible witness, but for existentials in general there can be > 1 possible witness and the witness may depend on values in the state.)
 - For $\sigma \models \forall v. p$, we need that for every possible value, σ satisfies p when we add a binding for v to that value. E.g., $\{y = 0\} \models \forall x. x^2 \geq y$ because no matter which value α we use, we find $\{y = 0, x = \alpha\} \models x^2 \geq y$.
- For the two examples above, there wasn't already a binding for x , so adding the new binding simply made the state set larger. If the state had already had a binding for x , we would have needed to replace it with the new binding.
 - E.g., if $\sigma = \{x = 8, y = 1\}$, then if we add the binding $x = 0$ to σ , we want to end up with $\{x = 0, y = 1\}$, not $\{x = 8, y = 1, x = 0\}$.
- **Definition:** For any state σ and variable v , the state $\sigma - v$ (" σ less v ") is σ with any binding for v removed: if $\sigma = \tau \cup \{v = \beta\}$, then $\sigma - v = \tau$. If there is no binding for v in σ , then $\sigma - v = \sigma$.
 - Another way to phrase this definition is say that that $(\sigma - v)(u)$ is undefined if $u \equiv v$, (regardless of whether $\sigma(v)$ is defined or not), and $(\sigma - v)(u) \equiv \sigma(u)$ otherwise.
 - $(\sigma - v)(u)$ means the state $\sigma - v$ applied to the variable u .
- **Definition:** For any state σ , variable v , and value α , the state $\sigma[v \mapsto \alpha]$, the **update of σ at v with α** is $(\sigma - v) \cup \{v = \alpha\}$: If v has no binding in σ , then $\sigma[v \mapsto \alpha] = \sigma \cup \{v = \alpha\}$; if $\sigma = \tau \cup \{v = \beta\}$, then $\sigma[v \mapsto \alpha] = \tau \cup \{v = \alpha\}$. (Note we've lost the connection to β .)
 - Another way to phrase the definition of $\sigma[v \mapsto \alpha]$ is that $\sigma[v \mapsto \alpha](v) = \alpha$ and for any variable $u \neq v$, $\sigma[v \mapsto \alpha](u) = \sigma(u)$.

- **Examples:** Let $\sigma = \{x = 2, y = 6\}$, so $\sigma[x \mapsto 0] = \{x = 0, y = 6\}$ in the following:
 - $\sigma[x \mapsto 0](x) = 0$ (regardless of $\sigma(x)$ being 2)
 - $\sigma[x \mapsto 0](y) = 6$ (since we didn't update y).
 - $\sigma[x \mapsto 0](x+y) = 0+6 = 6$ because the "x" in "x+y" gets evaluated to 0.
 - $\sigma[x \mapsto 0] \models x^2 \leq 0$ even though the original $\sigma \not\models x^2 \leq 0$
- Multiple updates
 - We can do a sequence of updates on a state. E.g., $\sigma[x \mapsto 0][y \mapsto 8]$ is a doubly updated state. If $\sigma = \{x = 2, y = 6\}$, then $\sigma[x \mapsto 0][y \mapsto 8] = \{x = 0, y = 8\}$.
 - The order of update doesn't matter if you have two different variables.
 - E.g., $\sigma[x \mapsto 0][y \mapsto 8] = \sigma[y \mapsto 8][x \mapsto 0]$.
 - If you update the same variable twice; the second update replaces the first.
 - E.g., $\sigma[x \mapsto 0][x \mapsto 17] = \sigma[x \mapsto 17]$.
- **Updating of array elements**
 - To update an array element, we need the name of the array, the index whose value we're to change, and the new value. The array must already exist.
 - We need some notation to talk about the updated array:
 - Let's extend the *state[variable \mapsto value]* notion to functions in general: If γ is a function then $\gamma[\alpha \mapsto \beta]$ is the same function as γ except that $\gamma[\alpha \mapsto \beta](\alpha) = \beta$ instead of $\gamma(\alpha)$. Example: If $\gamma = (1, 3, 5, 7)$, then $\gamma[0 \mapsto 9] = (9, 3, 5, 7)$, $\gamma[1 \mapsto 9] = (1, 9, 5, 7)$, etc.
 - Updating $b[\alpha]$ to be β corresponds to updating the function $\sigma(b)$ that b stands for:
 - $\sigma[b[\alpha] \mapsto \beta] = \sigma[b \mapsto \gamma_1]$ where $\gamma_1 = \sigma(b)[\alpha \mapsto \beta]$.
 - **Example:** If $\sigma(b) = (1, 3, 5, 7)$, then $\sigma[b[0] \mapsto 9] = \sigma[b \mapsto (9, 3, 5, 7)]$ because (as we saw above), if $\gamma = \sigma(b)$, then $\gamma[0 \mapsto 9] = (9, 3, 5, 7)$.
 - Treating the update $\sigma(b)$ this way doesn't mean that we're conceptually changing all its values (it's just a conceptual way to handle the update).

F. Satisfaction of Quantified Predicates

- Using updated states, we can define satisfaction for quantified predicates:
 - $\sigma \models \exists v. p$ means that for one or more "witness" values α (of the right type for v), we have $\sigma[v \mapsto \alpha] \models p$. Note we don't have to specify all possible witnesses; one will do.
 - E.g., $\sigma \models \exists x. x^2 \leq 0$, using 0 as the witness
 - $\sigma[x \mapsto 0] \models x^2 \leq 0$, since $\sigma[x \mapsto 0](x^2) = 0^2 = 0 \leq \sigma[x \mapsto 0](0)$.
 - $\sigma \models \forall v. p$ means that for every possible value α (of the right type for v), we have $\sigma[v \mapsto \alpha] \models p$.

- E.g., $\sigma \models \forall x. x^2 \geq 0$, because for every integer α , we have $\sigma[x \mapsto \alpha] \models x^2 \geq 0$, since $\sigma[x \mapsto \alpha](x^2) = \alpha^2 \geq 0 = \sigma[x \mapsto \alpha](0)$. I.e.,
 - ...
 - $\sigma[x \mapsto -1] \models x^2 \geq 0$, since $\sigma[x \mapsto -1](x^2) = (-1)^2 = 1 \geq 0 = \sigma[x \mapsto -1](0)$
 - $\sigma[x \mapsto 0] \models x^2 \geq 0$, since $\sigma[x \mapsto 0](x^2) = 0^2 = 0 \geq 0 = \sigma[x \mapsto 0](0)$
 - $\sigma[x \mapsto 1] \models x^2 \geq 0$, since $\sigma[x \mapsto 1](x^2) = 1^2 = 1 \geq 0 = \sigma[x \mapsto 1](0)$
 - ...
- When checking for satisfaction, the value of a quantified variable comes from an updated state, so any old value of the variable is lost.
- **Free and bound occurrences of a variable:** Whenever a variable occurs in a predicate, the occurrence may or may not be within the textual scope of a quantifier. Occurrences in the scope of a quantifier are **bound** to the quantifier; occurrences not in a scope are **free**.
 - E.g., $x > 5 \wedge \exists x. x^2 = y$ has two occurrences of x : the one in $x > 5$ (which is free) and the one in $x^2 = y$ (which is bound).
 - When asking if $\sigma \models p$, the value of $\sigma(x)$ is important for the free occurrences of x (and only the free occurrences); for the bound occurrences, the value of x is connected to the quantifier.
 - E.g., for $\sigma \models x > 5 \wedge \exists x. x^2 = y$, the value of $\sigma(x)$ is critical for deciding whether or not $\sigma \models x > 5$, but it's irrelevant for $\sigma \models \exists x. x^2 = y$.
 - E.g., if $\sigma(x) = 6$ and $\sigma(y) = 4$, then σ does satisfy $x > 5 \wedge \exists x. x^2 = y$, using 2 or -2 as the witness for the quantified x :
 - $\sigma \models x > 5 \wedge \exists x. x^2 = y$ if
 - $\sigma \models x > 5$ and $\sigma \models \exists x. x^2 = y$
 - $\sigma \models x > 5$ is true
 - $\sigma \models \exists x. x^2 = y$ if for some α (e.g., $\alpha=2$)
 - $\sigma[x \mapsto \alpha] \models x^2=y$, which is true
 - If $\sigma(x) = 6$ and $\sigma(y) = 5$, then σ does not satisfy the predicate because we can find no witness for the quantified x .
 - If $\sigma(x) = 4$ and $\sigma(y) = 9$, then σ does not satisfy the predicate because we don't satisfy the $x > 5$ sub-predicate
 - In some sense, it doesn't matter which variable we use as a quantified variable, as long as we're consistent.
 - E.g., $\exists x. x^2 = y$ and $\exists z. z^2 = y$ have the same meaning: $\sigma \models \exists x. x^2 = y$ exactly when $\sigma \models \exists z. z^2 = y$, irrespective of whether or not $\sigma(x)$ and $\sigma(z)$ are defined or not (or what values they have if they are defined).

- On the other hand, **we can't arbitrarily rename free occurrences of variables**:
 $\exists x . x^2 = y$ and $\exists x . x^2 = z$ are very different predicates. Depending on σ , we might have $\sigma \models \exists x . x^2 = y$ but $\sigma \not\models \exists x . x^2 = z$. (E.g., if $\sigma = \{y = 4, z = 5\}$.)