Syntactic Substitution

CS 536: Science of Programming, Fall 2019

9/24

A. *Why*

- Syntactic substitution is used in the assignment rules to calculate weakest preconditions (and later, strongest postcondition).

B. *Objectives*

At the end of today you should

- Know what syntactic substitution is and how to do it.
- Be able to carry out substitution on an expression or predicate.

C. *Syntactic Substitution*

- Recall that $wp(v \mathbin{\textbf{:=}} e, P(v)) \equiv P(e)$
- The operation of going from $P(v)$ to $P(e)$ is called **syntactic substitution**.
- A common notation is $p\,[\,e\,/\,v\,]$. The advantage of this notation is that it's easier to do multiple ("iterated") substitutions. There are other notations people use, such as $p\,[\,v := e\,]$, $p\,[\,v \mapsto e\,]$, and p_v^e .

D. *Substitution Into An Expression*

- As part of substitution into a predicate, we need to be able to **substitute into an expression**; the idea is to take an expression e and replace its occurrences of variable v with expression e'.
- **Notation**: We write $e\,[\,e'\,/\,v\,]$, pronounced "e with e' (substituted) for v". We'll treat the substitution brackets as having very high precedence, so we'll need parentheses around e for complex expressions.
- **Example 1**: `x + y[5/x]` ≡ `x + (y[5/x])` ≡ `x + y` but `(x+y)[5/x]` ≡ `5+y`.
- To carry out $e\,[\,e'\,/\,v\,]$, go through e, and everywhere we see an occurrence of v, replace it by (e'). If the parentheses are redundant, we can omit them.
  - If e has no occurrence of v (there's no v to replace), then $e\,[\,e'\,/\,v\,] \equiv e$. Another way to say this is that if e only uses variables $\not\equiv v$, then $e\,[\,e'\,/\,v\,] \equiv e$.
- Note: Substitution is a textual operation. For example, `(x+x)[2/x]` ≡ `2+2`, which = `4` in any state, but `(x+x) [2/x]` ≢ `4`.
- **Example 2**: `(a – x)[2/x]` ≡ `a – (2)` ≡ `a – 2` (the parentheses are redundant)
- **Example 3**: `(x * (x+1))[b–c/x]` ≡ `(b – c) * (b – c + 1)` (the parentheses are required).
- **Example 4**: `(b[x*y ])[x+3/x]` ≡ `b[(x+3)*y]`
- **Example 5**: `(y + b[x]) [x*3/x]` ≡ `y + b[x * 3]`
- **Example 6**: (**if** `x > 0` **then** `–x` **else** `0` **fi**)`[z+2/x]`
  ≡ **if** `z+2 > 0` **then** `–(z+2)` **else** `0` **fi**
- **Example 7**: `(b[x*(x+1)/2]) [y+4/x]` ≡ `b[(y+4)*((y+4)+1)/2 ]` ≡ `b[(y+4)*(y+4+1)/2 ]`.

- The technical definition of $e\,[\,e'/v\,]$ is done by cases on the structure of e. Briefly, we have constants and variables as base cases and expressions with subexpressions as recursive cases.

Definition of $e\,[\,e'/v\,]$, by Structural Induction

- **Case 1** (base cases)
  - $c\,[\,e'/v\,] \equiv c$ if c is a constant
  - $v\,[\,e'/v\,] \equiv (e')$
  - If $v \not\equiv w$, then $w\,[\,e'/v\,] \equiv w$
- **Case 2** (recursive cases): Consider the expressions that have subexpressions: function calls $f(e_1, e_2, \ldots)$, array indexing expressions $\mathtt{b}[\,e_1, e_2, \ldots\,]$, parenthesized expressions (e_1), unary operations $\oplus\, e_1$, binary operations $e_1 \oplus e_2$ and ternary operations e_1 **?** e_2 **:** e_3 (or **if** e_1 **then** e_2 **else** e_3 **fi**), we recursively process each subexpression.
  - Let $e_1{}' \equiv (e_1)\,[\,e'/v\,]$, $e_2{}' \equiv (e_2)\,[\,e'/v\,]$, etc.
  - Then $(f(e_1, e_2, \ldots))\,[\,e'/v\,] \equiv f(e_1{}', e_2{}', \ldots)$
  - And $(\mathtt{b}[\,e_1, e_2, \ldots\,])\,[\,e'/v\,] \equiv \mathtt{b}[\,e_1{}', e_2{}', \ldots]$
  - And $(e_1 \oplus e_2)\,[\,e'/v\,] \equiv e_1{}' \oplus e_2{}'$
  - And so on.

E. Substitution Into A Predicate

- **Notation**: $p\,[\,e/v\,]$ is pronounced "p with e (substituted) for v" and stands for the result of substituting e for each (free) occurrence of v in p. (Don't worry about free and bound occurrences of a variable until we get to quantified predicates.)
- Substitution into expressions and predicates is a syntactic operation. For example, $(\mathtt{x} > 0)\,[\,1/\mathtt{x}\,] \equiv 1 > 0$, which $\Leftrightarrow$ true, but $(\mathtt{x} > 0)\,[\,1/\mathtt{x}\,] \not\equiv \mathtt{T}$.
- Note: If p contains no occurrences at all of v, then $p\,[\,e/v\,] \equiv p$.

Substitution Into A Non-Quantified Predicate

- If p contains no quantifiers, then $p\,[\,e/v\,]$ is straightforward to calculate.
- **Case 1** (Non-quantified predicates):
  - For $p\,[\,e/v\,]$, go through the predicate p and replace each occurrence of v with (e); if the parentheses are redundant, we can omit them. Note for tests like $e_1 < e_2$, we substitute into the expressions e_1 and e_2: $(e_1 < e_2)\ [\,e/v\,] \equiv e_1\,[\,e/v\,] < e_2\,[\,e/v\,]$.
- **Example 8**: $(\mathtt{x} > 0 \rightarrow \mathtt{y} \ge \mathtt{x}/2)\,[\,\mathtt{z+1}/\mathtt{x}\,]$
  $\equiv (\mathtt{x} > 0)\,[\,\mathtt{z+1}/\mathtt{x}\,] \rightarrow (\mathtt{y} \ge \mathtt{x}/2)\,[\,\mathtt{z+1}/\mathtt{x}\,]$
  $\equiv (\mathtt{z+1} > 0 \rightarrow \mathtt{y} \ge (\mathtt{z+1})/2\,)$. (The parentheses around $\mathtt{z+1}$ are necessary)

Free and Bound Variables and Occurrences of Variables

- **Notation**: Q stands for a quantifier ($\forall$ or $\exists$).
- For the definition of $(Q\,x\,.\,q)\,[\,e/v\,]$, our natural instinct is to think that $(Q\,x\,.\,q)\,[\,e/v\,] \equiv (Q\,x\,.\,(q[\,e/v\,]))$, but in fact this isn't always true because of a distinction between "free" and "bound" occurrences of variables.

- **Definition**: If an occurrence of a variable v in a predicate is within the scope of a quantifier over v, then it is a **bound occurrence**, else it is a **free occurrence**. A variable v **is free in** (= **occurs free in**) p iff it has a free occurrence in p. Similarly, v **is bound in** (= **occurs bound in**) p iff it has a bound occurrence in p.
- For any variable v and predicate p, there are four possibilities:
  - v is neither free nor bound in p: v doesn't occur at all in p.
  - v is free but not bound in p: v occurs at least once in p, and all the occurrences of v are free.
  - v is not free but is bound in p: v occurs at least once in p, and all the occurrences of v are bound.
  - v is free and bound in p: v occurs at least twice in p with at least one occurrence being free and at least one occurrence being bound.
- **Example 9**: If $p \equiv \texttt{x} > \texttt{z} \wedge \exists \texttt{x}.\exists \texttt{y}.\texttt{y} \le \texttt{f}(\texttt{x}, \texttt{y})$, then
  - `x` is free and bound in p. (Its first occurrence is free; its second is bound.)
  - `y` is bound in p but not free in p.
  - `z` is free in p but not bound in p.
  - `w` is neither free nor bound in p.
- The reason we're interested in occurrences of variables being free or bound in a predicate is that we only substitute for free occurrences of a variable. In computer science terms, we're looking for non-local variables, not local variables.
- Taking polynomials as an example, $p(x) = x^2 + a\,x + y$. If we want to substitute 17 for y, that's fine: $p(x) = x^2 + a\,x + 17$; substituting expressions with variables that aren't bound in the definition is okay too: substituting $(z^3 + 1)$ for y gives us $p(x) = x^2 + a\,x + (z^3 + 1)$. But if we want to substitute something like $(x+3)$ for y (note: x is the defined parameter variable), we **don't** want $p(x) = x^2 + a\,x + (x+3)$. But if we had defined $p(w) = w^2 + a\,w + y$, then substituting $(x+3)$ for y gives us $p(w) = w^2 + a\,w + (x+3)$.

Substitution Into A Quantified Predicate, part 1

- If p has no quantifiers over v, then every occurrence of every variable in p is free, so for $p[e/v]$, we can just scan p looking for occurrences of v and replace them by e. This was case 1 of our definition of substitution.
- In the remaining cases, we substitute into a quantified predicate: $(Q\,x\,.\,q)[e/v]$.
- **Case 2**: $(Q\,v\,.\,q)[e/v]$: The quantified variable matches the variable we're substituting for. Then $(Q\,v\,.\,q)$ has no free occurrences of v because all the free occurrences of v in q are bound by the quantifier. Since there aren't any free occurrences of v, there's nothing to replace, and $(Q\,v\,.\,q)[e/v] \equiv (Q\,v\,.\,q)$.
  - **Example 10**: $(\texttt{x} > 0 \wedge \exists\,\texttt{x}.\texttt{x} \le \texttt{f}(\texttt{y}))[17/\texttt{x}] \equiv 17 > 0 \wedge \exists \texttt{x}.\texttt{x} \le \texttt{f}(\texttt{y})$. Here, the first occurrence of x (in $\texttt{x} > 0$) is free, so we replace it with `17`, but the second occurrence of `x` is bound, so we don't do any replacement.
- **Case 3**: If $x \not\equiv v$ and x does not occur in e, then $(Q\,x\,.\,q)[e/v] \equiv (Q\,x\,.\,(q[e/v]))$. Here, we go through q and replace its free occurrences of v with e.
  - **Example 11**: $(\texttt{y} \ge 0 \to \forall \texttt{x}\,.\,\texttt{x} > \texttt{y} \to \texttt{x}*\texttt{x} > \texttt{y} \wedge \exists\,\texttt{y}.\texttt{f}(\texttt{y}) > \texttt{x}))[17/\texttt{y}]$
    $\equiv 17 \ge 0 \to \forall \texttt{x}\,.\,(\texttt{x} > \texttt{y} \to \texttt{x}*\texttt{x} > \texttt{y} \wedge \exists\,\texttt{y}.\texttt{f}(\texttt{y}) > \texttt{x})[17/\texttt{y}]$
    $\equiv 17 \ge 0 \to \forall \texttt{x}\,.\,\texttt{x} > 17 \to \texttt{x}*\texttt{x} > 17 \wedge \exists\,\texttt{y}.\texttt{f}(\texttt{y}) > \texttt{x}$ (`y` in `f(y)` is bound, so no substituting for it)

- In case 3, the restriction that the quantified variable not appear in e keeps us from having a "capture" problem, where occurrences of x in e are free, but when we we replace an occurrence of v by e in $Q\ \mathtt{x}\,.\,q\,[\,e/v\,]$, the occurrences of x in e become bound, which changes their meaning.
  - **Example 11**: $(\exists \mathtt{y}\,.\,\mathtt{y} = \mathtt{v}^2)[\mathtt{x{+}1}\,/\,\mathtt{v}\,] \equiv (\exists \mathtt{y}\,.\,\mathtt{y} = (\mathtt{x{+}1})^2)$. If we were to let $(\exists \mathtt{x}\,.\,\mathtt{x} = \mathtt{v}^2)[\mathtt{x{+}1}\,/\,\mathtt{v}\,]$ be $(\exists \mathtt{x}\,.\,\mathtt{x} = (\mathtt{x{+}1})^2)$, then the x in x+1 would become bound to the x in ∃x (= the x is "**captured**").
- The way out of this problem is to **rename the quantified variable** from x to something not in e; that way the quantifier can't capture occurrences of x.
- **Case 4**: (The painful case) If $x \not\equiv v$ and x occurs in e, then $(Q\,x\,.\,q)[\,e/v\,] \equiv (Q\,z\,.\,(\,q\,[\,z/x\,]\,[\,e/v\,]\,))$ where z is a **fresh variable** (one not used in e or q).
- **Example 12**: Using z as a fresh variable, we have

$(\mathtt{g}(\mathtt{x}, \mathtt{v}) < 0 \wedge (\exists\, \mathtt{x}\,.\,\mathtt{x} = \mathtt{v}^2) \wedge \mathtt{h}(\mathtt{x}, \mathtt{v}) > 0)[\,\mathtt{x{+}1}\,/\,\mathtt{v}\,]$

$\equiv \mathtt{g}(\mathtt{x}, \mathtt{x{+}1}) < 0 \wedge (\exists\, \mathtt{z}\,.\,((\mathtt{x} = \mathtt{v}^2)[\,\mathtt{z}/\mathtt{x}\,])[\,\mathtt{x{+}1}\,/\,\mathtt{v}\,]) \wedge \mathtt{h}(\mathtt{x}, \mathtt{x{+}1}) > 0$

// Pick fresh variable, quantify over it and then substitute for it in the body

$\equiv \mathtt{g}(\mathtt{x}, \mathtt{x{+}1}) < 0 \wedge (\exists\, \mathtt{z}\,.\,\mathtt{z} = \mathtt{v}^2)[\,\mathtt{x{+}1}\,/\,\mathtt{v}\,]$

$\equiv \mathtt{g}(\mathtt{x}, \mathtt{x{+}1}) < 0 \wedge \exists\, \mathtt{z}\,.\,\mathtt{z} = (\mathtt{x} + 1\,)^2$

- Note there's some ambiguity in the definition: Which "fresh" variable should we choose?

Syntactic Substitution

CS 536: Science of Programming

A. *Why*

- Syntactic substitution is used in the assignment rules to calculate the weakest precondition (and as we'll see, the strongest postcondition).

B. *Objectives*

At the end of this activity you should

- Be able to calculate a syntactic substitution on an expression or predicate.

C. *Questions*

1. Calculate $(\texttt{x+i*b+c} = 0)[\texttt{i+1}/\texttt{i}][\texttt{b+c}/\texttt{c}]$.
2. Let p be $\exists \texttt{x}.\, \texttt{x} < \texttt{y} \wedge \texttt{x}^2 \geq \texttt{y+k}$
   a. What is $p[5/\texttt{x}]$?
   b. What is $p[5/\texttt{y}]$?
   c. What is $p[5/\texttt{z}]$?
   d. What is $p[\texttt{y*2}/\texttt{y}]$?
   e. What is $p[\texttt{y*k}/\texttt{y}]$?
   f. What is $p[(\texttt{x} + \texttt{y}) \div 2/\texttt{y}]$?
3. Give an example where $(v * w)[e/v][e'/w]$ and $(v * w)[e'/w][e/v]$ are
   a. Syntactically equal ($\equiv$)
   b. Syntactically unequal ($\not\equiv$).
4. In the predicate $(\exists \texttt{x} .\, \texttt{x} < \texttt{y} \wedge \texttt{x}^2 \geq \texttt{y+k})$, $\texttt{x}$ is bound, but in $(\texttt{x} < \texttt{y} \wedge \texttt{x}^2 \geq \texttt{y+k})$, $\texttt{x}$ is free — is this a contradiction?
5. For substitution into a quantified predicate $(Q\, x \,.\, p)[e/v]$, we could just say "always rename x to something fresh." Why do you think we didn't do that?

6. Let $p \equiv (\forall\, \texttt{x} \,.\, \exists\, \texttt{y} \,.\, \texttt{R}(\texttt{x}, \texttt{y}, \texttt{z})) \wedge (\exists\, \texttt{z} \,.\, \texttt{R}(\texttt{x}, \texttt{y}, \texttt{z}))$ where $\texttt{R}$ is a boolean function over three arguments.
   a. What is $p[17/\texttt{w}]$?
   b. What is $p[17/\texttt{x}]$?
   c. What is $p[\texttt{y*2}/\texttt{y}]$?
   d. What is $p[\texttt{y*2}/\texttt{z}]$?
   e. What is $p[\texttt{a*z}/\texttt{y}][\texttt{a+b}/\texttt{z}]$?

Solution to Activity 12 (Syntactic Substitution)

1. $(x+i*b+c = 0)[i+1/i][b+c/c] \equiv (x + (i+1)*b + c = 0)[b+c/c]$
   $\equiv x+(i+1)*b+(b+c) = 0$

2. Let $p \equiv \exists x . x < y \wedge x^2 \geq y+k$
   - 2a. $p[5/x] \equiv p$ unchanged
   - 2b. $p[5/y] \equiv (\exists x . x < y \wedge x^2 \geq y+k)[5/y] \equiv \exists x . x < 5 \wedge x^2 \geq 5+k$
   - 2c. $p[5/z] \equiv p$ unchanged because z doesn't occur in p
   - 2d. $p[y*2/y] \equiv (\exists x . x < y \wedge x^2 \geq y+k)[y*2/y] \equiv \exists x . x < y*2 \wedge x^2 \geq y*2+k$
   - 2e. $p[y*k/y] \equiv (\exists x . x < y \wedge x^2 \geq y+k)[y*k/y]$
     $\equiv \exists x . x < y*k \wedge x^2 \geq y*k+k$
   - 2f. $p[(x + y) \div 2/y] \equiv (\exists x . x < y \wedge x^2 \geq y+k)[(x + y) \div 2/y]$
     $\equiv \exists v.(x < y \wedge x^2 \geq y+k)[v/x][(x + y) \div 2/y]$ (note renaming of x to v)
     $\equiv \exists v .(v < y \wedge v^2 \geq y+k)[(x + y) \div 2/y]$
     $\equiv \exists v . v < (x + y) \div 2 \wedge v^2 \geq (x + y) \div 2 + k$

3. (Cases where $(v * w)[e/v][e'/w]$ and $(v * w)[e'/w][e/v]$ are $\equiv$ and $\not\equiv$.)
   - 3a. One case is when v doesn't occur in e' and w doesn't occur in e.

   Example: $(v * w)[v*2/v][a*w/w] \equiv (v*2 * w)[a*w/w]$
   $\equiv v*2 * (a*w) \equiv (v * (a*w))[v*2/v]$
   $\equiv (v * w)[a*w/w][v*2/v]$
   - 3b. One case is when w appears in e and v appears in e' (at least, for certain e and e').

   Example: $(v * w)[w-3/v][a*v/w] \equiv ((w-3) * w)[a*v/w] \equiv (w-3) * (a * v)$
   but $(v * w)[a*v/w][w-3/v] \equiv (v * (a*v))[w-3/v] \equiv (w-3) * (a* (w-3))$

4. No, this is exactly what a quantifier does: It captures the x's that are free in its body and makes them bound with respect to any context that includes the quantified predicate.

5. Because it's confusing/annoying to have to come up with fresh variables if we don't really need them.

6. (Substitutions with $p \equiv (\forall x . \exists y . R(x, y, z)) \wedge \exists z . R(x, y, z)$)
   - 6a. $p[17/w] \equiv p$ (because w doesn't occur in p)
   - 6b. $p[17/x] \equiv (\forall x . \exists y . R(x, y, z)) \wedge \exists z . R(17, y, z)$)
   - 6c. $p[y*2/y] \equiv (\forall x . \exists y . R(x, y, z)) \wedge \exists z . R(x, y*2, z)$)
   - 6d. $p[y*2/z] \equiv (\forall x . \exists v . R(x, y, z)[v/y][y*2/z]) \wedge \exists z . R(x, y, z)$)
     (using v as a fresh variable)
     $\equiv (\forall x . \exists v . R(x, v, y*2)) \wedge \exists z . R(x, y, z)$)

6e. p[a*z/y][a+b/z]

$\equiv$ (∀x.∃y.R(x, y, z)) ∧ ∃v.R(x, y, z)[v/z][a*z/y])[a+b/z]

(using v as a fresh variable)

$\equiv$ ((∀x.∃y.R(x, y, z)) ∧ ∃v.R(x, y, v)[a*z/y])[a+b/z]

$\equiv$ ((∀x.∃y.R(x, y, z)) ∧ ∃v.R(x, a*z, v))[a+b/z]

$\equiv$ ((∀x.∃y.R(x, y, a+b)) ∧ ∃v.R(x, a*(a+b), v))

(Note the parens around a+b are required)

