**Weakest Preconditions pt. 2**

*CS 536: Science of Programming, Fall 2019*

9/20; 9/24 misc cleanup

A. Why

- Weakest liberal preconditions (wlp) and weakest preconditions (wp) are the most general requirements that a program must meet to be correct

B. Objectives

At the end of today you should understand

- How to add error domain predicates to the wlp of a loop-free program to obtain its wp.

C. Some Examples of Calculating wp/wlp:

- The programs in these examples don't end in “state” ⊥, so the wp and wlp are equivalent.

  - Example 2: wp(x := x+1, x ≥ 0) ≡ x+1 ≥ 0

  - Example 3: wp(y := y+x; x := x+1, x ≥ 0)
    ≡ wp(y := y+x, wp(x := x+1, x ≥ 0))
    ≡ wp(y := y+x, x+1 ≥ 0) ≡ x+1 ≥ 0

  - Example 4: wp(y := y+x; x := x+1, x ≥ y)
    ≡ wp(y := y+x, wp(x := x+1, x ≥ y))
    ≡ wp(y := y+x, x+1 ≥ y)
    ≡ x+1 ≥ y+x

  - If we were asked only to calculate the wp, we'd stop here. If we also wanted to logically simplify the wp then x+1 ≥ y+x ⇔ y ≤ 1.

  - Example 5: (Swap the two assignments in Example 4)
    wp(x := x+1; y := y+x, x ≥ y)
    ≡ wp(x := x+1, wp(y := y+x, x ≥ y))
    ≡ wp(x := x+1, x ≥ y+x)
    ≡ x+1 ≥ y+x+1 [⇔ y ≤ 0 if you want to logically simplify]

  - Example 6: wp(if y ≥ 0 then x := y fi, x ≥ 0)
    ≡ wp(if y ≥ 0 then x := y else skip fi, x ≥ 0)
    ≡ (y ≥ 0 ∧ wp(x := y, x ≥ 0)) ∨ (y < 0 ∧ wp(skip, x ≥ 0))
    ≡ (y ≥ 0 ∧ y ≥ 0) ∨ (y < 0 ∧ x ≥ 0).

    If we want to simplify logically, we can continue with
    ⇔ y ≥ 0 ∨ (y < 0 ∧ x ≥ 0)
    ⇔ (y ≥ 0 ∨ y < 0) ∧ (y ≥ 0 ∨ x ≥ 0)
    ⇔ y ≥ 0 ∨ x ≥ 0 (which is also ⇔ y < 0 → x ≥ 0, if you prefer)
D. Avoiding Runtime Errors in Expressions

- To avoid runtime failure of $\sigma(e)$, we'll take the context in which we're evaluating $e$ and augment it with a predicate that guarantees non-failure of $\sigma(e)$. For example, for $\{P(e)\} v := e \{P(v)\}$, we'll augment the precondition to guarantee that evaluation of $e$ won't fail.
- For each expression $e$, we will define a **domain predicate** $D(e)$ such that $\sigma \not\models D(e)$ implies $\sigma(e) \not\models \bot_c$.
  - This predicate has to be defined recursively, since we need to handle complex expressions like $D(b[b[i]]) \equiv 0 \leq i < \text{size}(b) \land 0 \leq b[i] < \text{size}(b)$.
  - As with $wp$ and $sp$, the domain predicate for an expression is unique only up to logical equivalence. For example, $D(x/y + u/v) \equiv y \neq 0 \land v \neq 0 \iff v \neq 0 \land y \neq 0$.
- **Definition** (Domain predicate $D(e)$ for expression $e$) We must define $D$ for each kind of expression that can cause a runtime error:
  - $D(c) \equiv D(v) \equiv T$ if where $c$ is a constant and $v$ is a variable.
    - Evaluation of a variable or constant doesn't cause failure.
  - $D(b[e]) \equiv D(e) \land 0 \leq e < \text{size}(b)$
  - $D(e_1 / e_2) \equiv D(e_1 \% e_2) \iff D(e_1) \land D(e_2) \land e_2 \neq 0$
  - $D(\sqrt{e}) \equiv D(e) \land e \geq 0$
    - And so on, depending on the datatypes and operations being used.
  - The various operations ($+,-,\text{etc.}$) and relations ($\leq,=,\text{etc.}$) don't cause errors but we still have to check their subexpressions:
    - $D(e_1 \text{ op } e_2) \equiv D(e_1) \land D(e_2)$, except for $\text{op} \equiv / \lor \%$
    - $D(\text{op } e) \equiv D(e)$, unless you add an operator that can cause runtime failure.
  - $D(\text{if } B \text{ then } e_1 \text{ else } e_2 \text{ fi}) \equiv D(B) \land (B \rightarrow D(e_1)) \land (\neg B \rightarrow D(e_2))$
    - (For a conditional expression, we only need safety of the one branch we execute.)
- **Example 7** $D(b[b[i]]) \equiv D(b[i]) \land 0 \leq b[i] < \text{size}(b)$
  \[
  \equiv D(i) \land 0 \leq i < \text{size}(b) \land 0 \leq b[i] < \text{size}(b) \\
  \iff 0 \leq i < \text{size}(b) \land 0 \leq b[i] < \text{size}(b)
  \]
- **Example 8** $D((-b + \sqrt{b*b - 4*a*c}))/2(a))$
  \[
  \equiv D(e) \land D(2*a) \land 2*a \neq 0 \\
  \equiv D(-b) \land D(\sqrt{b*b - 4*a*c}) \land D(2*a) \land 2*a \neq 0 \\
  \iff D(\sqrt{b*b - 4*a*c}) \land 2*a \neq 0 \quad \text{since } D(-b) \equiv D(2*a) \equiv T \\
  \equiv b*b - 4*a*c \land (b*b - 4*a*c \geq 0) \land 2*a \neq 0 \\
  \iff b*b - 4*a*c \geq 0 \land 2*a \neq 0
  \]
- **Example 9** $D(\text{if } 0 \leq i < \text{size}(b) \text{ then } b[i] \text{ else } 0 \text{ fi})$
  \[
  \equiv D(B) \land (B \rightarrow D(b[i])) \land (\neg B \rightarrow D(0)) \quad \text{where } B \equiv 0 \leq i < \text{size}(b) \\
  \equiv (B \rightarrow D(b[i])) \land (\neg B \rightarrow T) \quad \text{since } D(B) \text{ and } D(0) \equiv T \\
  \iff B \rightarrow D(b[i]) \\
  \equiv B \rightarrow D(i) \land 0 \leq i < \text{size}(b) \quad \text{since everything implies true} \\
  \equiv B \rightarrow D(b[i]) \quad \text{expanding } D(b[i])
Avoiding Runtime Errors in Programs

- Recall that we extended our notion of operational semantics to include \((S, \sigma) \rightarrow^\ast (E, \bot)\) to indicate that evaluation of \(S\) causes a runtime failure.

- We can avoid runtime failure of statements by adding domain predicates to the preconditions of statements. Though for loops we can’t in general calculate the \(wp/wp\) for loop-free programs, since we would also want to show termination of a loop.

**Definition:** For statement \(S\), the predicate \(D(S)\) gives a sufficient condition to avoid runtime errors.

- \(D(\text{skip})\) \(\equiv T\)
- \(D(v := e)\) \(\equiv D(e)\)
- \(D(b[e_1] := e_2)\) \(\equiv D(b[e_1]) \land D(e_2)\)
- \(D(S_1; S_2, q) \equiv D(S_1) \land D(S_2)\)
- \(D(\text{if } B \text{ then } S_1 \text{ else } S_2 \text{ fi, } q) \equiv D(B) \land (B \rightarrow D(S_1)) \land (\neg B \rightarrow D(S_2))\)
- \(D(\text{if } B_1 \rightarrow S_1 \sqcup B_2 \rightarrow S_2 \text{ fi, } q) \equiv D(B_1 \lor B_2) \land (B_1 \lor B_2) \land (B_1 \rightarrow D(S_1)) \land (B_2 \rightarrow D(S_2))\)
  - The condition \((B_1 \lor B_2)\) avoids failure of the nondeterministic **if-fi** due to none of the guards holding. This definition extends easily to **if-fi** with more than two guarded commands.
- \(D(\text{while } B \text{ do } S_1 \text{ od}) \equiv D(B) \land (B \rightarrow D(S_1))\)
- \(D(\text{do } B_1 \rightarrow S_1 \sqcup B_2 \rightarrow S_2 \text{ od}) \equiv D(B_1 \lor B_2) \land (B_1 \lor B_2) \land (B_1 \rightarrow D(S_1)) \land (B_2 \rightarrow D(S_2))\)
  - The domain predicate for nondeterministic **do-od** is like that for **if-fi** except that having none of the guards hold does not cause an error.

- With the domain predicates, it’s easy to extend \(wp\) for \(wp\) for loop-free programs, since we would also want to show termination of a loop.

**Definition:** If \(S\) is not a loop, then \(wp(S, q) \Leftrightarrow wp(S, D(S) \land q)\)

**Example 10:** If a program does a division, then the \(wp\) and \(wp\) can differ:

- \(D(x := y; z := v/x)\)
  \[\equiv D(x := y) \land D(z := v/x)\]
  \[\Leftrightarrow D(v/x) \quad \text{since } D(x := y) \equiv D(y) \equiv T\]
  \[\Leftrightarrow x \neq 0 \quad \text{Technically, } D(v/x) \equiv D(v) \land D(x) \land x \neq 0\]

- \(wp(x := y; z := v/x, z > x+2)\)
  \[\equiv wp(x := y, wp(z := v/x, z > x+2 \land D(x := y; z := v/x)))\]
  \[\Leftrightarrow wp(x := y, wp(z := v/x, z > x+2 \land x \neq 0)) \quad \text{Substituting from above}\]
  \[\equiv wp(x := y, v/x > x+2 \land x \neq 0)\]
  \[\equiv v/y > y+2 \land y \neq 0\]
Weakest Preconditions, pt. 1

CS 536: Science of Programming

A. Why
- The weakest precondition and weakest liberal preconditions are the most general preconditions that a program needs in order to run correctly.

B. Objectives
At the end of this activity you should be able to
- Describe the relationship between \(wp(S, q_1 \lor q_2), wp(S, q_1), \) and \(wp(S, q_2)\) and how it differs for deterministic and nondeterministic programs.
- Be able to calculate the \(wlp\) of a simple loop-free program.

C. Problems
1. How are \(wp(S, q_1 \lor q_2)\) and \(wp(S, q_1), \) and \(wp(S, q_2)\), related if \(S\) is deterministic? If \(S\) is nondeterministic?

2. Calculate the \(wlp\) in each of the following cases. Just syntactically calculate the \(wlp\); don't also logically simplify the result.)
   a. \(wlp(k := k - s, n = 3 \land k = 4 \land s = -7).\)
   b. \(wlp(n := n*(n-k), n = 3 \land k = 4 \land s = -7).\)
   c. \(wlp(n := n*(n-k); k := k - s, n > k + s)\)

3. Let \(Q(i, s) \equiv 0 \leq i \leq n \land s = \text{sum}(0, i)\) where \(\text{sum}(u, v)\) is the sum of \(u, u+1, ..., v\) (when \(u \leq v\)) or 0 (when \(u > v\)).
   a. Calculate \(wp(i := i+1; s := s+i, Q(i, s)).\)
   b. Calculate \(wp(s := s+i+1; i := i+1, Q(i, s)).\)
   c. Calculate \(wp(s := s+i; i := i+1, Q(i, s)).\) (This one isn't compatible with \(s = \text{sum}(0, i).\))

4. Calculate the \(wp\) below. (Again, just calculate the syntactically \(wp\) without logically simplifying the result.)
   a. \(wp(\text{if } x \geq 0 \text{ then } x := x*2 \text{ else } x := y \text{ if } x := c*x, a \leq x < y)\)

For Problems 5 and 6, be sure to include the domain predicates. If you want, logically simplify as you go.

5. Calculate \(p\) to be the \(wp\) of \(|p| x := y/b[i] \{ x > 0 \}.\)

6. Calculate \(p_1\) and \(p_2\) to be the \(wp\) of \(|p_1| y := \text{sqrt}(b[j]) \{ z < y \} \) and \(|p_2| j := x/j \{ p_1 \}.\)
**Solution to Activity 11 (Weakest Preconditions, pt. 2)**

1. For deterministic \( S \), \( wp(S, q_1 \lor q_2) \equiv wp(S, q_1) \lor wp(S, q_2) \). For nondeterministic \( S \), we have \( \Rightarrow \) instead of \( \Leftrightarrow \).

2. (Calculate \( wlp \))
   a. \( wlp(k := k - s, n = 3 \land k = 4 \land s = -7) \equiv n = 3 \land k-s = 4 \land s = -7 \)
   b. \( wlp(n := n*(n-k), n = 3 \land k = 4 \land s = -7) \equiv n*(n-k) = 3 \land k = 4 \land s = -7 \)
   c. \( wlp(n := n*(n-k); k := k-s, n > k+s) \)
      \( \equiv wlp(n := n*(n-k), wlp(k := k-s, n > k+s)) \)
      \( \equiv wlp(n := n*(n-k), n > k-s+s) \)
      \( \equiv n*(n-k) > k-s+s \)

3. (wp involving sums) We have \( Q(i, s) \equiv 0 \leq i \leq n \land s = \text{sum}(0, i) \).
   a. \( wp(i := i+1; s := s+i, Q(i, s)) \)
      \( \equiv wp(i := i+1, wp(s := s+i.Q(i, s)) \)
      \( \equiv wp(i := i+1, Q(i, s+i)) \)
      \( \equiv wp(i := i+1, 0 \leq i \leq n \land s+i = \text{sum}(0, i)) \)
      \( \equiv 0 \leq i+1 \leq n \land s+i = \text{sum}(0, i+1) \)
   b. \( wp(s := s+i+1; i := i+1, Q(i, s)) \)
      \( \equiv wp(s := s+i+1, wp(i := i+1.Q(i, s)) \)
      \( \equiv wp(s := s+i+1, Q(i+1, s)) \)
      \( \equiv wp(s := s+i+1, 0 \leq i+1 \leq n \land s = \text{sum}(0, i+1)) \)
      \( \equiv 0 \leq i+1 \leq n \land s+i+1 = \text{sum}(0, i+1) \)
   c. \( wp(s := s+i; i := i+1, Q(i, s)) \)
      \( \equiv wp(s := s+i, wp(i := i+1, Q(i, s)) \)
      \( \equiv wp(s := s+i, Q(i+1, s)) \)
      \( \equiv wp(s := s+i, 0 \leq i+1 \leq n \land s = \text{sum}(0, i+1)) \)
      \( \equiv 0 \leq i+1 \leq n \land s+i = \text{sum}(0, i+1) \) [which isn’t compatible with \( s = \text{sum}(0, i) \)]

4. (wp of if-then)
   a. \( wp(\text{if } B \text{ then } x := x/2 \text{ fi}; y := x, x = 5 \land y = z) \)
      \( \equiv wp(\text{if } B \text{ then } x := x/2 \text{ fi}, wp(y := x, x = 5 \land y = z)) \)
      \( \equiv wp(\text{if } B \text{ then } x := x/2 \text{ fi}, x = 5 \land x = z) \)
      \( \equiv (B \rightarrow wp(x := x/2, x = 5 \land x = z)) \land (\neg B \rightarrow wp(\text{skip}, x = 5 \land x = z)) \)
      \( \equiv (B \rightarrow x/2 = 5 \land x/2 = z) \land (\neg B \rightarrow x = 5 \land x = z) \)
   b. \( wp(\text{if } x \geq 0 \text{ then } x := x*2 \text{ else } x := y \text{ fi}; x := c*x, a \leq x < y) \)
      \( \equiv wp(S, wp(x := c*x, a \leq x < y)) \) where \( S \) is the if statement
      \( \equiv wp(S, a \leq c*x < y) \)
      \( \equiv wp(\text{if } x \geq 0 \text{ then } x := x*2 \text{ else } x := y \text{ fi}, a \leq c*x < y) \)
\[\equiv (x \geq 0 \rightarrow wp(x := x \times 2, a \leq c \times x < y)) \land (x < 0 \rightarrow wp(x := y, a \leq c \times x < y))\]
\[\equiv (x \geq 0 \rightarrow a \leq c \times (x \times 2) < y) \land (x < 0 \rightarrow a \leq c \times y < y)\]

5. For \(\{p\} x := y/b[i] \{x > 0\}\), let \(p \iff wp(x := y/b[i], x > 0)\)
\[\iff D(y/b[i]) \land (y/b[i] > 0)\]
\[\iff (0 \leq i < size(b) \land b[i] \neq 0) \land (y/b[i] > 0)\]

6. For \(\{p_1\} y := \sqrt{b[j]} \{z < y\}\), let \(p_1 \iff wp(y := \sqrt{b[j]}, z < y)\)
\[\iff D(\sqrt{b[j]}) \land wp(y := \sqrt{b[j]}, z < y)\]
\[\iff 0 \leq j < size(b) \land b[j] \geq 0 \land z < \sqrt{b[j]}\]

For \(\{p_2\} j := x/j\; \{p_1\}\), let \(p_2 \iff wp(j := x/j, p_1)\)
\[\iff D(x/j) \land wp(j := x/j, p_1)\]
\[\iff j \neq 0 \land (0 \leq j < size(b) \land b[j] \geq 0 \land z < \sqrt{b[j]}[x/j])\]
\[\iff j \neq 0 \land 0 \leq x/j < size(b) \land b[x/j] \geq 0 \land z < \sqrt{b[x/j]}\]