*Finding Invariants; Array Assignments**

CS 536: Science of Programming, Spring 2023

Due Wed Apr 26, 11:59 pm

2023-04-25 p.1, 2023-04-26 p.1, 2023-04-29: pp. 2,3

A. Why?

- The hardest part of programming is finding good loop invariants.
- There are heuristics for finding them but no algorithms that work in all cases.
- Array assignments aren't like assignments to plain variables because the actual item to change can't be determined until runtime.

B. Outcomes

After this homework, you should be able to

- Describe the strength connections among the conditions of $\{ p_0 \}\ S_0\ \{ \textbf{inv}\ p \}\ \textbf{while}\ B\ \textbf{do}\ S\ \textbf{od}\ \{ q \}$.
- Describe and use the invariant-finding heuristics "Replace a constant by a variable", "Drop a conjunct" and "Add a disjunct".
- Be able to perform textual substitution to replace an array element.
- Be able to calculate the *wp* of an array element assignment.

C. Problems [60 points total]

Classes 19 & 20: Finding Invariants [24 points]

1. [3 points] Say we have a postcondition q and are looking for a compatible loop invariant p and test B. Briefly, does p need to be weaker or stronger than q? How does B fit in? How about initialization?

2. [9 points] Take the postcondition $(0 \le x \ \wedge\ y < m \wedge (b \to x < m))$ and list all of the candidate invariant/while header combinations you can get using the technique Replace a Constant by a Variable? Assume b, x, and y are a variables and m is a constant. [2023-04-25]

3. [9 points] Using the same postcondition, List all of the candidate invariant/while header combinations you can get using the technique Delete a Conjunct.

---

* This is the last assignment! For the Final Exam, be sure to study the practices for Classes 22 & up.

4. [3 points] Take the candidate invariant/while headers of the previous problem and explain briefly why they can all also be viewed as instances of the technique Add a Disjunct.

Class 21: Array Assignments [36 points]

For these problems, simplify as you go (it will make life easier).

5. [9 points] Calculate $wp(b[x] := y, b[y] \geq b[n])$. Show your calculations.

6. [9 points] Calculate $wp(b[n] := b[x], b[y] > b[n])$. Show your calculations.

7. [18 points] Is the triple $\{b[m] < b[n]\}\ \ b[b[n]] := b[m]\ \{b[m] \leq b[n]\}$ valid? I.e., does $(b[m] < b[n]) \rightarrow wp(b[b[n]] := b[m],\ \ b[m] \leq b[n])$? Show your calculations.
   [2023-04-26]

Solution to Homework 9

Classes 19 & 20: Finding Invariants

1. p needs to be weaker than q because we need to satisfy p before entering the loop (and satisfying q is hard). We need initialization to establish p, preferably with some simple code that sets the loop variables. Obviously B needs to be testable but more generally, $p \wedge B$ needs to be stronger than q so that $p \wedge B \rightarrow q$.

2. $\{\textbf{inv}\ (z \leq x \wedge y < m \wedge (b \rightarrow x < m))\}$ **while** $z \neq 0$ [2023-04-29]
   $\{\textbf{inv}\ (0 \leq x \wedge y < z \wedge (b \rightarrow x < m))\}$ **while** $z \neq m$
   $\{\textbf{inv}\ (0 \leq x \wedge y < m \wedge (b \rightarrow x < z))\}$ **while** $z \neq m$

3. $\{\textbf{inv}\ (y < m \wedge (b \rightarrow x < m)))\}$ **while** $0 > x$
   $\{\textbf{inv}\ (0 \leq x \wedge (b \rightarrow x < m)))\}$ **while** $y \geq m$
   $\{\textbf{inv}\ (0 \leq x \wedge y < m)\}$ **while** $\neg(b \rightarrow x < m))$ or the equivalent while $b \wedge x \geq m$.

Class 21: Array Assignments

5. (Calculate the wp of an array assignment)
   $wp(b[x] := y,\ \ b[y] \geq b[n])$

$\equiv (b[y])[y/b[x]] \geq (b[n])[y/b[x]]$

$\equiv$ ***if*** $y = x$ ***then*** y ***else*** $b[y]$ ***fi*** $\geq$ ***if*** $n = x$ ***then*** y ***else*** $b[n]$ ***fi*** [2023-04-29]

Has no obviously good simplification [2023-04-29]

~~$\equiv$ ***if*** $y=x$ ***then*** $b[y] \geq b[y]$ ***else*** $b[y] \geq b[x]$ ***fi***~~

~~$\Leftrightarrow y=x \vee b[y] \geq b[x]$~~

6. (Calculate the wp of an array assignment)

$wp(b[n] := b[x],\ b[y] \geq b[n])$ [2023-04-29]

$\equiv (b[y])[b[x]/b[n]] \geq (b[n])[b[x]/b[n]]$

$\equiv$ ***if*** $y = n$ ***then*** $b[x]$ ***else*** $b[n]$ ***fi*** $\geq b[x]$

$\Leftrightarrow y = n \vee b[n] \geq b[x]$

~~$\Leftrightarrow b[x] \geq$ ***if*** $y=n$ ***then*** $b[x]$ ***else*** $b[y]$ ***fi***~~

~~$\Leftrightarrow$ ***if*** $y=n$ ***then*** $b[x] \geq b[x]$ ***else*** $b[x] \geq b[y]$ ***fi***~~

~~$\Leftrightarrow y=n \vee b[x] \geq b[y]$~~

7. (Is $\{b[m] < b[n]\}\ b[b[n]] := b[m]\ \{b[m] \leq b[n]\}$ valid?)

It's sufficient to show that the precondition implies the *wp* of the assignment and postcondition. I.e.,

$(b[m] < b[n]) \rightarrow wp(b[b[n]] := b[m], b[m] \leq b[m])$

First let's calculate the *wp* :

$wp(b[b[n]] := b[m],\ b[m] \leq b[n])$

$\equiv (b[m] \leq b[n])[b[m] / b[b[n]]]$

$\equiv (b[m])[b[m] / b[b[n]]] \leq (b[n])[b[m] / b[b[n]]]$

$\equiv$ ***if*** $m = b[n]$ ***then*** $b[m]$ ***else*** $b[m]$ ***fi***

$\leq$ ***if*** $n = b[n]$ ***then*** $b[m]$ ***else*** $b[n]$ ***fi***

$\Leftrightarrow b[m] \leq$ ***if*** $n = b[n]$ ***then*** $b[m]$ ***else*** $b[n]$ ***fi***

$\Leftrightarrow$ ***if*** $n = b[n]$ ***then*** $b[m] \leq b[m]$ ***else*** $b[m] \leq b[n]$ ***fi***

$\Leftrightarrow$ ***if*** $n = b[n]$ ***then*** T ***else*** $b[m] \leq b[n]$ ***fi***

$\Leftrightarrow$***if*** $(n = b[n]) \wedge T) \vee (n \neq b[n]) \wedge (b[m] \leq b[n])$ ***fi***

$\Leftrightarrow n = b[n] \vee b[m] \leq b[n]$ or

$\Leftrightarrow n \neq b[n] \rightarrow b[m] \leq b[n]$ (they're equivalent)

Now that we know the *wp*, we can our say triple is valid if

$(b[m] < b[n]) \rightarrow wp(b[b[n]] := b[m], b[m] \leq b[m])$

$\Leftrightarrow (b[m] < b[n]) \rightarrow (n = b[n] \vee b[m] \leq b[n])$

which is true.

