Loop Invariants and Proof Outlines Practice

Stefan Muller, based partially on material by Jim Sasaki

CS 536: Science of Programming, Spring 2022
Lecture 14–16

1 Problems

1. What are the constants in the postcondition $x = max(b[0], b[1], ..., b[n-1])$? Using the technique "replace a constant by a variable," list the possible invariants for this postcondition.

2. Repeat, on the postcondition $x = n!$, where $n!$ is short for $product(1, n)$.

3. Repeat, on the postcondition $\forall i.0 \le i < n \to b[i] = 3$.

4. Repeat, on the postcondition $\forall i.\forall j.0 \le i < K \land K \le j < n \to b[i] < b[j]$ (that is, every value in $b[0 \ldots K-1]$ is less than every value in $b[K \ldots n-1]$).

5. Write a loop invariant for the factorial function below, then extend it into a full proof outline.

$$
\begin{array}{ll}
 & \{n > 0\} \\
m := \overline{1} & \\
i := \overline{0} & \\
\textsf{while } (i < n)\ \{ & \\
\quad i := i + \overline{1}; & \\
\quad m := m * i & \\
\} & \{m = n!\}
\end{array}
$$

6. Write a loop invariant for the sum function we saw in class that starts from n and decreases i:

```
s := 0;
var i := n;
while (i > 0)
{
    i := i - 1;
    s := s + i;
}
```

7. The following program (in Dafny syntax) finds the minimum value of an array/sequence. Write an appropriate postcondition (or `ensures` clause) that ensures m is the minimum value and write an appropriate loop invariant.

```
method findMin (a: seq<int>) returns (m: int)
requires |a| > 0
{
    m := a[0];
    var k := 1;
    while (k < |a|)
    {
```

```
        if (a[k] < m) {
            m := a[k];
        }
        k := k + 1;
    }
}
```

8. The following program (in Dafny syntax) returns true if and only if the array/sequence a has at least two different elements (the postcondition ensures this by saying that there exist two distinct values m and n which are both in the array). Write an appropriate loop invariant for the loop.

```
method hasTwoVals (a: seq<int>) returns (e: bool)
requires (forall i :: (i >= 0 && i < |a|) ==> a[i] >= 0)
ensures e ==> (exists m, n : int :: m != n
                && exists i, j :: i >= 0 && i < |a| && j >= 0 && j < |a|
                                  && (a[i] == m) && (a[j] == n))
{
    var i := 0;
    var m := -1;
    var n := -2;
    while (i < |a|)
    {
        if (m == -1) { m := a[i]; }
        if (n == -1 && a[i] != m) {n := a[i]; }
        i := i + 1;
    }
    e := m > -1 && n > -1;
}
```

For problems 9–11, you are given a minimal proof outline and should expand it to a full proof outline. Don't give the formal proof of partial correctness. Do list any predicate logic obligations.

9. $\{n > 1\}\ k := \overline{1}; s := \overline{0}\ \{0 \le k < n \wedge s = sum(0, k-1)\}$
   (a) Use wp on both assignments.
   (b) Use sp on both assignments.
   (c) Use sp on the left assignment and wp on the right assignment.

10. $\{T\}$ if $x \ge 0$ then $\{y := x\}$ else $\{y := -x\}$ $\{y = |x|\}$
    (a) Use sp on both branches and on the if as a whole.
    (b) Use $(P \wedge B)$ and $(P \wedge \neg B)$ (from the conditional rule) as overall preconditions for the two branches, and use wp on both branches.

11. Since the invariant of the loop below is a predicate function call, substitutions using it are easy.

$$
\begin{array}{l}
\{n \ge 0\} \\
x := \overline{0}; \\
y := \overline{1}; \\
\{\textbf{inv}\ P(a, b, x, y)\}; \\
\textsf{while}\ x < n\ \{; \\
\quad x := f(x, y) \\
\quad y := f(y, x) \\
\}\{a + x < b - y\}
\end{array}
$$

(a) Use wp as much as you can.

(b) Use sp as much as you can.

12. Expand the minimal proof outline below. The program has a bug; in the full proof outline, in what line(s) and in what form does the bug appear? Also, give two ways to fix the bug.

$$
\begin{array}{l}
\{\textbf{inv } 0 \le k \le n+1 \wedge s = sum(0, k-1)\}; \\
\mathsf{while}\ k \le n\ \{; \\
\quad k := k + 1; \\
\quad s := s + k \\
\} \\
\{s = sum(0, n)\}
\end{array}
$$

2 Solutions

1. The best candidates are 0, which gives us $x = max(b[j], b[1], ..., b[n-1]) \wedge 0 \leq j \leq n-1$, and $n-1$, which gives us $x = max(b[0], b[1], ..., b[j]) \wedge 0 \leq j \leq n-1$.
   You could also just consider n and 1 constants by themselves and get $x = max(b[0], b[1], ..., b[j-1]) \wedge 0 \leq j \leq n$ and $x = max(b[0], b[1], ..., b[n-j]) \wedge 0 \leq j \leq n$, respsectively. These invariants will probably be less useful.

2. Expanding using the product function gives two constants, 1 and n.
   Using 1 gives $x = product(i, n) \wedge 1 \leq i \leq n$.
   Using n gives $x = product(1, i) \wedge 1 \leq i \leq n$.

3. Using 0 gives $0 \leq j < n \wedge \forall i.j \leq i < n \rightarrow b[i] = 3$.
   Using n gives $0 \leq j < n \wedge \forall i.0 \leq i < j \rightarrow b[i] = 3$.
   We can also use 3, which gives $\forall i.0 \leq i < n \rightarrow b[i] = k$ (this is probably less useful).

4. We have 0, n and two occurrences of K.
   Using 0 gives $0 \leq k < K \wedge \forall i.\forall j.k \leq i < K \wedge K \leq j < n \rightarrow b[i] < b[j]$.
   Using n gives $K \leq k < n \wedge \forall i.\forall j.0 \leq i < K \wedge K \leq j < k \rightarrow b[i] < b[j]$.
   Using the first K gives $0 \leq k \leq K \wedge \forall i.\forall j.0 \leq i < k \wedge K \leq j < n \rightarrow b[i] < b[j]$.
   Using the second K gives $K \leq k < n \wedge \forall i.\forall j.0 \leq i < K \wedge l \leq j < n \rightarrow b[i] < b[j]$.

5.

	$\{n > 0\}$
$m := \overline{1}$	$\{n > 0 \wedge m = 1\}$
$i := \overline{0}$	$\{n > 0 \wedge m = 1 \wedge i = 0\}$
$\{\mathbf{inv}\ i \leq n \wedge m = i!\}$	
while $(i < n)$ {	$\{i \leq n \wedge m = i! \wedge i < n\} \Rightarrow \{i < n \wedge m = (i - 1 + 1)!\}$
$\quad i := i + \overline{1};$	$\{i \leq n \wedge m = (i-1)!\} \Rightarrow \{i \leq n \wedge m * i = i!\}$
$\quad m := m * i$	$\{i \leq n \wedge m = i!\}$
}	$\{i \leq n \wedge m = i! \wedge i \geq n\} \Rightarrow \{m = n!\}$

6. $i \geq 0 \wedge s = sum(i, n)$

7. Postcondition: $\exists m.(\exists i.i \geq 0 \wedge i < |a| \wedge a[i] = m) \wedge (\forall i \in [0, |a|-1].a[i] \geq m)$.
   Invariant: $k \leq |a| \wedge (\exists i.i \geq 0 \wedge i < |a| \wedge a[i] = m) \wedge (\forall i \in [0, k-1].a[i] \geq m)$

8. $(m > -1 \rightarrow \exists i.a[i] = m) \wedge (n > -1 \rightarrow \exists i.a[i] = n) \wedge m \neq n$

9. (a)

$$\begin{array}{ll} & \{n > 1\} \Rightarrow \{0 \leq 1 < n \wedge 0 = sum(0, k-1)\} \\ k := \overline{1}; & \{0 \leq k < n \wedge 0 = sum(0, k-1)\} \\ s := \overline{0} & \{0 \leq k < n \wedge s = sum(0, k-1)\} \end{array}$$

   Predicate logic obligation: $n > 1 \Rightarrow 0 \leq 1 < n \wedge 0 = sum(0, k-1)$

   (b)

$$\begin{array}{ll} & \{n > 1\} \\ k := \overline{1}; & \{n > 1 \wedge k = 1\} \\ s := \overline{0} & \{n > 1 \wedge k = 1 \wedge s = 0\} \Rightarrow \{0 \leq k < n \wedge s = sum(0, k-1)\} \end{array}$$

   Predicate logic obligation: $n > 1 \wedge k = 1 \wedge s = 0 \Rightarrow 0 \leq k < n \wedge s = sum(0, k-1)$

   (c)

$$\begin{array}{ll} & \{n > 1\} \\ k := \overline{1}; & \{n > 1 \wedge k = 1\} \Rightarrow \{0 \leq k < n \wedge 0 = sum(0, k-1)\} \\ s := \overline{0} & \{0 \leq k < n \wedge s = sum(0, k-1)\} \end{array}$$

   Predicate logic obligation: $n > 1 \wedge k = 1 \Rightarrow 0 \leq k < n \wedge 0 = sum(0, k-1)$

10. (a)

$$\begin{array}{ll}
 & \{T\} \\
\mathsf{if} x \geq 0 \ \mathsf{then}\ \{ & \{x \geq 0\} \\
\quad y := x & \{x \geq 0 \wedge y = x\} \\
\}\ \mathsf{else}\ \{ & \{x < 0\} \\
\quad y := -x & \{x < 0 \wedge y = -x\} \\
\} & \{(x \geq 0 \wedge y = x) \vee (x < 0 \wedge y = -x)\} \Rightarrow \{y = |x|\}
\end{array}$$

Obligation: $(x \geq 0 \wedge y = x) \vee (x < 0 \wedge y = -x) \Rightarrow y = |x|$

(b)

$$\begin{array}{ll}
 & \{T\} \\
\mathsf{if} x \geq 0 \ \mathsf{then}\ \{ & \{x \geq 0\} \Rightarrow \{x = |x|\} \\
\quad y := x & \{y = |x|\} \\
\}\ \mathsf{else}\ \{ & \{x < 0\} \Rightarrow \{-x = |x|\} \\
\quad y := -x & \{y = |x|\} \\
\} & \{y = |x|\}
\end{array}$$

Obligations: $x \geq 0 \Rightarrow x = |x|$ and $x < 0 \Rightarrow -x = |x|$.

11. (a)

$$\begin{array}{ll}
 & \{n \geq 0\} \Rightarrow \{P(a,b,0,1)\} \\
x := \overline{0}; & \{P(a,b,x,1)\} \\
y := \overline{1}; & \{P(a,b,x,y)\} \\
\{\mathbf{inv}\ P(a,b,x,y)\}; & \\
\mathsf{while}\ x < n\ \{; \{x < n \wedge P(a,b,x,y)\} \Rightarrow \{P(a,b,f(x,y),f(y,x)\} & \\
\quad x := f(x,y) & \{P(a,b,x,f(y,x)\} \\
\quad y := f(y,x) & \{P(a,b,x,y)\} \\
\} & \{x \geq n \wedge P(a,b,x,y)\} \Rightarrow \{a + x < b - y\}
\end{array}$$

Obligations: 1) $n \geq 0 \Rightarrow P(a,b,0,1)$, 2) $x < n \wedge P(a,b,x,y) \Rightarrow P(a,b,f(x,y),f(y,x)$ and 3) $x \geq n \wedge P(a,b,x,y) \Rightarrow a + x < b - y$

(b)

$$\begin{array}{ll}
 & \{n \geq 0\} \\
x := \overline{0}; & \{n \geq 0 \wedge x = 0\} \\
y := \overline{1}; & \{n \geq 0 \wedge x = 0 \wedge y = 1\} \\
\{\mathbf{inv}\ P(a,b,x,y)\}; & \\
\mathsf{while}\ x < n\ \{; \{x < n \wedge P(a,b,x,y)\} & \\
\quad x := f(x,y) & \{x_0 < n \wedge P(a,b,x_0,y) \wedge x = f(x_0,y)\} \\
\quad y := f(y,x) & \{x_0 < n \wedge P(a,b,x_0,y_0) \wedge x = f(x_0,y_0) \wedge y = f(y_0,x)\} \\
\} & \{x \geq n \wedge P(a,b,x,y)\} \Rightarrow \{a + x < b - y\}
\end{array}$$

Obligations: 1) $n \geq 0 \wedge x = 0 \wedge y = 1 \Rightarrow P(a,b,0,1)$, 2) $x_0 < n \wedge P(a,b,x_0,y_0) \wedge x = f(x_0,y_0) \wedge y = f(y_0,x) \Rightarrow P(a,b,x,y)$ 3) $x \geq n \wedge P(a,b,x,y) \Rightarrow a + x < b - y$

12.

$$\begin{array}{ll}
\{\mathbf{inv}\ 0 \leq k \leq n+1 \wedge s = \mathit{sum}(0,k-1)\} & \\
\mathsf{while}\ k \leq n\ \{; & \{0 \leq k \leq n+1 \wedge s = \mathit{sum}(0,k-1)\} \\
 & \Rightarrow \{0 \leq k+1 \leq n+1 \wedge s+k+1 = \mathit{sum}(0,k)\} \\
\quad k := k + \overline{1}; & \{0 \leq k \leq n+1 \wedge s+k = \mathit{sum}(0,k-1)\} \\
\quad s := s + k & \{0 \leq k \leq n+1 \wedge s = \mathit{sum}(0,k-1)\} \\
\} & \{k \geq n \wedge 0 \leq k \leq n+1 \wedge s = \mathit{sum}(0,k-1)\} \Rightarrow \{s = \mathit{sum}(0,n)\}
\end{array}$$

The bug is in this proof obligation: $0 \leq k \leq n+1 \wedge s = \mathit{sum}(0,k-1) \Rightarrow 0 \leq k+1 \leq n+1 \wedge s+k+1 = \mathit{sum}(0,k)$

This isn't true: the value of s is off by one.