Assignment 1 — 2014 — solution

Theodore S. Norvell

6892 Due Oct 2 2014

Q0 [24] (Read all parts before attempting any.)
Suppose a is an array of numbers of length n, and x holds a number.
(a)[4] Write a contract (specification) for computing $\sum_{i\in\{0,..n\}} a(i) \times x^i$ into y.
Solution:

$\{true\}$
?
$\left\{y = \sum_{i\in\{0,..n\}} a(i) \times x^i\right\}$

(b)[4] Use the technique of replacing a constant by a variable to obtain an invariant.
Solution:
I'll replace the n with a natural variable k which ranges from 0 to n (inclusive)

$$I : k \leq n \wedge y = \sum_{i\in\{0,..k\}} a(i) \times x^i$$

(c)[4] Write a correct proof outline that solves the problem. (You may assume that computing x^i for any integer i is an operation in the programming language, although an expensive one.) Don't worry about efficiency at this point.
Solution:

$\{true\}$
$k := 0$
$y := 0$
$\left\{I : k \leq n \wedge y = \sum_{i\in\{0,..k\}} a(i) \times x^i\right\}$
while $k \neq n$ do
    $y := y + a(k) \times x^k$
    $k := k + 1$
end while
$\left\{y = \sum_{i\in\{0,..n\}} a(i) \times x^i\right\}$

(d)[4] What is the variant?

Solution: $n - k$

(e)[4] Introduce a tracking variable to improve the efficiency of the algorithm. State the linking invariant that relates the new variable to the rest of the state.

Solution: I'll introduce a number variable z with linking invariant $z = x^k$.

(f)[4] Rewrite the proof outline from part (c) to use the tracking variable.

Solution:

$\{true\}$
$k := 0$
$y := 0$
$z := 1$
$\left\{I' : k \leq n \wedge y = \sum_{i \in \{0,..k\}} a(i) \times x^i \wedge z = x^k\right\}$
while $k \neq n$ do
  $y := y + a(k) \times z$
  $z := z \times x$
  $k := k + 1$
end while
$\left\{y = \sum_{i \in \{0,..n\}} a(i) \times x^i\right\}$

[By the way, it is interesting to try a different invariant

$$I'' : k \leq n \wedge y = \sum_{i \in \{k,..n\}} a(i) \times x^{i-k}$$

(I rewrote x^i as x^{i-0} before replacing the two occurrences of 0 with k.) This invariant leads straight to a solution with only one multiplication in the loop: an algorithm known as Horner's rule.

$\{true\}$
$k := n$
$y := 0$
$\left\{I'' : k \leq n \wedge y = \sum_{i \in \{k,..n\}} a(i) \times x^{i-k}\right\}$
while $k \neq 0$ do
  $y := x \times y + a(k-1)$
  $k := k - 1$
end while
$\left\{y = \sum_{i \in \{0,..n\}} a(i) \times x^i\right\}$]

Q1 [16] (Read all parts before attempting any.)

(a)[4] Write a contract (specification) for computing the integer part of the square root of a natural number. (Integer part means floor, i.e., rounded down.)

Solution: Let s and p be variables of type $\mathbb{N}$. The contract is

$$\{\ \mathit{true}\ \}\ ?\ \{\ p = \lfloor\sqrt{s}\rfloor\ \} \qquad \text{or, equivalently,} \qquad \{\ \mathit{true}\ \}\ ?\ \{\ p^2 \leq s < (p+1)^2\}$$

(b)[4] Give a linking invariant that makes your specification equivalent to the specification

$$\{\neg A(m) \wedge A(n) \wedge m < n\}\ ?\ \{\neg A(p) \wedge A(p+1)\}$$

Solution: First I'm going to rewrite my postcondition as $p^2 \leq s < (p+1)^2$, which can be further rewritten as

$$\neg\left(p^2 > s\right) \wedge (p+1)^2 > s$$

This suggests a function $A(i) = \left(i^2 > s\right)$. We need to pick an m small enough that $m^2 > s$ is not possible. I'll pick $m = 0$. We also need n large enough that $n^2 > s$ is sure to be true. I'll pick $n = s + 1$. (Picking $n = s$ will not do, as $0^2 > 0$ is not true and nor is $1^2 > 1$.) Thus the liking invariant is

$$L : m = 0 \wedge n = (s+1) \wedge \forall i \in \mathbb{N} \cdot \left(A(i) = \left(i^2 > s\right)\right)$$

Assuming this linking invariant L we have

$$\begin{array}{ll} & \neg A(m) \wedge A(n) \wedge m < n \\ = & \\ & \neg\left(m^2 > s\right) \wedge n^2 > s \wedge m < n \\ = & \\ & \neg\left(0^2 > s\right) \wedge (s+1)^2 > s \wedge 0 < s+1 \\ = & \\ & 0 \leq s \wedge (s+1)^2 > s \wedge 0 < s+1 \\ = & \\ & \mathit{true} \end{array}$$

For the postcondition we have

$$\begin{array}{ll} & \neg A(p) \wedge A(p+1) \\ = & \\ & \neg p^2 > s \wedge (p+1)^2 > s \\ = & \\ & p^2 \leq s \wedge (p+1)^2 > s \\ = & \\ & p = \lfloor\sqrt{s}\rfloor \end{array}$$

So given this linking invariant holds, the pre- and postconditions from part (a) are equivalent to those given in the question.

(c)[8] Use your linking invariant to derive a correct proof outline for the contract given in part (a) from the algorithm given in slide set 4 pages $\{5, .., 8\}$. Running time should roughly proportional to the number of bits required to represent the input. See slide set 5, for an exemplar.

I suggest doing this in three stages: first, introduce additional variables and the linking invariant; second, rewrite the algorithms so that A (at least) is no longer needed; third eliminate A and any other variables no longer needed.

Solution:

(i) Introduce additional variable s of type $\mathbb{N}$ using the following linking invariant from part (b)

$$L : m = 0 \wedge n = (s+1) \wedge \forall i \in \mathbb{N} \cdot \big(A(i) = \big(i^2 > s\big)\big)$$

Since m, n, and A are not changed by the algorithm, there is no need to add or modify any code to maintain this invariant.

$\{\ \neg A(m) \wedge A(n) \wedge m < n \ \underline{\wedge\ L}\ \}$
$p := m$
$q := n$
$\{\ I : m \leq p < q \leq n \wedge \neg A(p) \wedge A(q) \ \underline{\wedge\ L}\ \}$
// variant is $q - p$
while $q \neq p + 1$ do
  $\{\ I \wedge q \neq p+1 \ \underline{\wedge\ L}\ \}$
  $r := \left\lfloor \frac{p+q}{2} \right\rfloor$
  $\{\ p < r < q \wedge I \ \underline{\wedge\ L}\ \}$
  if $A(r)$ then $q := r$ else $p := r$ end if
end while
$\{\ \neg A(p) \wedge A(p+1) \ \underline{\wedge\ L}\ \}$

(ii) Rewrite so that A, m, and n are only mentioned in L.

$\{\ \underline{\neg\big(0^2 > s\big)} \wedge \underline{(s+1)^2 > s} \wedge \underline{0 < s+1} \wedge L\ \}$
$p := \underline{0}$
$q := \underline{s+1}$
$\{\ I' : \underline{0} \leq p < q \leq s+1 \wedge \underline{\neg\big(p^2 > s\big)} \wedge \underline{\big(q^2 > s\big)} \wedge L\ \}$
// variant is $q - p$
while $q \neq p + 1$ do
  $\{\ I' \wedge q \neq p+1\ \}$
  $r := \left\lfloor \frac{p+q}{2} \right\rfloor$
  $\{\ p < r < q \wedge I'\ \}$
  if $\underline{r^2 > s}$ then $q := r$ else $p := r$ end if
end while
$\{\ \underline{\neg\big(p^2 > s\big)} \wedge \underline{(p+1)^2} > s \wedge L\ \}$

(iii) Eliminate A, m, and n and clean up. At this point, I'm also dropping the requirement that $q \leq s+1$ from the invariant, as it is never used.

{ *true* }
$p := 0$
$q := s + 1$
{ $I'' : p < q \wedge p^2 \leq s < q^2$ }
// variant is $q - p$
while $q \neq p + 1$ do
  { $I'' \wedge q \neq p + 1$}
  $r := \left\lfloor \frac{p+q}{2} \right\rfloor$
  { $p < r < q \wedge I''$ }
  if $r^2 > s$ then $q := r$ else $p := r$ end if
end while
{ $p = \lfloor \sqrt{s} \rfloor$ }

Bonus [5]. Extend the solution from Q1 to find an outline that uses no multiplications.

Solution: Slide set 5 sets out a road map for this in the case that n is a power of 2 —i.e. s is one less than a power of 2. For the sake of finding out what would happen, I thought I would tackle the problem directly (i.e. without switching to the c-i representation from the p-q representation).

To start, let's look closely at r^2, which is the multiplication to eliminate

- When $p + q$ is even

$$\begin{aligned} r &= \frac{p+q}{2} \\ r^2 &= \frac{1}{4}\left(p^2 + 2 \times p \times q + q^2\right) \end{aligned}$$

- When $p + q$ is odd

$$\begin{aligned} r &= \frac{p+q-1}{2} \\ r^2 &= \frac{1}{4}\left(p^2 + 2 \times p \times q + q^2\right) - \frac{1}{2}(p+q) + \frac{1}{4} \end{aligned}$$

This suggests that we use a tracking variables for p^2, $p \times q$, and q^2. Let

$$L1 : pp = p^2 \wedge pq = p \times q \wedge qq = q^2$$

Data refining the solution to Q1, we get

$\{\ true\ \}$
$$\begin{bmatrix} p \\ q \\ \underline{pp} \\ \underline{pq} \\ \underline{qq} \end{bmatrix} := \begin{bmatrix} 0 \\ s+1 \\ \underline{0} \\ \underline{0} \\ \underline{(s+1)^2} \end{bmatrix}$$
$\{\ I''' : p < q \wedge p^2 \leq s < q^2 \ \underline{\wedge L1}\ \}$
// variant is $q - p$
while $q \neq p + 1$ do
  $\{\ I''' \wedge q \neq p+1\ \}$
  if $p + q$ is even then
    $$\begin{bmatrix} r \\ \underline{rr} \end{bmatrix} := \begin{bmatrix} \frac{1}{2}(p+q) \\ \underline{\frac{1}{4}(pp + 2 \times pq + qq)} \end{bmatrix}$$
  else
    $$\begin{bmatrix} r \\ \underline{rr} \end{bmatrix} := \begin{bmatrix} \frac{1}{2}(p+q-1) \\ \underline{\frac{1}{4}(pp + 2 \times pq + qq) - \frac{1}{2}(p+q) + \frac{1}{4}} \end{bmatrix}$$
  end if
  if $rr > s$ then
    $$\begin{bmatrix} q \\ \underline{qq} \\ \underline{pq} \end{bmatrix} := \begin{bmatrix} r \\ \underline{rr} \\ \underline{p \times r} \end{bmatrix}$$
  else
    $$\begin{bmatrix} p \\ \underline{pp} \\ \underline{pq} \end{bmatrix} := \begin{bmatrix} r \\ \underline{rr} \\ \underline{r \times q} \end{bmatrix}$$
  end if
end while
$\{\ p = \lfloor \sqrt{s} \rfloor\ \}$

Now we have multiplications $p \times r$ and $r \times q$ in the loop. So let's investigate them. When $p+q$ is even we have

$$\begin{aligned} r &= \frac{p+q}{2} \\ p \times r &= \frac{1}{2}\left(p^2 + p \times q\right) \\ r \times q &= \frac{1}{2}\left(p \times q + q^2\right) \end{aligned}$$

When $p + q$ is odd we have

$$\begin{aligned} r &= \frac{p+q}{2} - \frac{1}{2} \\ p \times r &= \frac{1}{2}\left(p^2 + p \times q - p\right) \\ r \times q &= \frac{1}{2}\left(p \times q + q^2 - q\right) \end{aligned}$$

Which leads to

$\{\ true\ \}$
$$\begin{bmatrix} p \\ q \\ pp \\ pq \\ qq \end{bmatrix} := \begin{bmatrix} 0 \\ s+1 \\ 0 \\ 0 \\ (s+1)^2 \end{bmatrix}$$
$\{\ I''' : p < q \wedge p^2 \leq s < q^2 \wedge L1\ \}$
// variant is $q - p$
while $q \neq p + 1$ do
$\{\ I''' \wedge q \neq p+1\}$
if $p + q$ is even then
$$\begin{bmatrix} r \\ rr \\ \underline{pr} \\ \underline{rq} \end{bmatrix} := \begin{bmatrix} \frac{1}{2}(p+q) \\ \frac{1}{4}(pp + 2 \times pq + qq) \\ \underline{\frac{1}{2}(pp+pq)} \\ \underline{\frac{1}{2}(pq+qq)} \end{bmatrix}$$
else
$$\begin{bmatrix} r \\ rr \\ \underline{pr} \\ \underline{rq} \end{bmatrix} := \begin{bmatrix} \frac{p+q-1}{2} \\ \frac{1}{4}(pp + 2 \times pq + qq) - \frac{1}{2}(p+q) + \frac{1}{4} \\ \underline{\frac{1}{2}(pp+pq-p)} \\ \underline{\frac{1}{2}(pq+qq-q)} \end{bmatrix}$$
end if
if $rr > s$ then
$$\begin{bmatrix} q \\ qq \\ pq \end{bmatrix} := \begin{bmatrix} r \\ rr \\ \underline{pr} \end{bmatrix}$$
else
$$\begin{bmatrix} p \\ pp \\ pq \end{bmatrix} := \begin{bmatrix} r \\ rr \\ \underline{rq} \end{bmatrix}$$
end if
end while
$\{\ p = \lfloor \sqrt{s} \rfloor\ \}$

There is only one multiplication left and that is the initialization of qq. However we've now removed all multiplications (that can't be represented by shifts in binary) out of the loop. And I'd be perfectly happy to stop here, having accomplished that much.

But let's eliminate that last multiplication anyway. The only requirement on the initial value of q is that it is big enough that $s < q^2$. So we could replace the initialization code with

$\{\ true\ \}$
$\begin{bmatrix} p \\ q \\ pp \\ pq \\ qq \end{bmatrix} := \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$
$\{\ 0 = p < q \wedge \underline{\text{pow2}(q)} \wedge L1\ \}$
while $s \geq qq$ do $\underline{[q, qq]} := [2 \times q, 4 \times qq]$ end while
$\{\ I''' : p < q \wedge p^2 \leq s < q^2 \wedge L1\ \}$

Mission accomplished. No multiplications are left.

However. It is worth noticing that, if we initialize q like this, then $q - p$ is a power of two after the first loop. And $q - p$ being a power of 2 is preserved by the body of the second loop, as the following argument shows: If $q - p$ is a power of two and $q - p > 1$, then $q - p$ is even and so is $p + q$. Thus r is half way between p and q, so $r - p = q - r = \frac{q-p}{2}$ and is also power of two.

Since powers of 2 greater than 1 are all even, we can ignore the case where $p + q$ is odd.

Using the notation pow2(x) to mean that x is a power of 2 we have

$\{\ true\ \}$
$\begin{bmatrix} p \\ q \\ pp \\ pq \\ qq \end{bmatrix} := \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$
$\{\ 0 = p < q \wedge \underline{\text{pow2}(q)} \wedge L1\ \}$
$\underline{\text{while } s \geq qq \text{ do } [q, qq] := [2 \times q, 4 \times qq] \text{ end while}}$
$\{\ I'''' : p < q \wedge p^2 \leq s < q^2 \wedge \underline{\text{pow2}(q - p)} \wedge L1\ \}$
while $q \neq p + 1$ do // variant is $q - p$
$\{\ I'''' \wedge q \neq p + 1\}$
$\begin{bmatrix} r \\ rr \\ pr \\ rq \end{bmatrix} := \begin{bmatrix} \frac{1}{2}(p + q) \\ \frac{1}{4}(pp + 2 \times pq + qq) \\ \frac{1}{2}(pp + pq) \\ \frac{1}{2}(pq + qq) \end{bmatrix}$
if $rr > s$ then $\begin{bmatrix} q \\ qq \\ pq \end{bmatrix} := \begin{bmatrix} r \\ rr \\ pr \end{bmatrix}$ else $\begin{bmatrix} p \\ pp \\ pq \end{bmatrix} := \begin{bmatrix} r \\ rr \\ rq \end{bmatrix}$ end if
end while
$\{\ p = \lfloor \sqrt{s} \rfloor\ \}$

We might worry about the divisions by 2 and 4. Do we need to worry about nonintegral values being assigned to rr, pr or rq? (Earlier we saw that $p + q$ is even, so no need to worry about r.) Since we designed the algorithm so that p, q, and r are natural numbers, we can be sure that r^2, $p \times r$, and $r \times q$ are all integers and thus, by $L1$, so are rr, pr, and rq. But we can also show this directly, without reference to the way the algorithm was developed: We can see that $q - p > 1 \Rightarrow \text{even}(p) \wedge \text{even}(q)$ is an invariant of both loops; thus pp, pq, and qq are all divisible by 4; thus the expressions $\frac{1}{4}(pp + 2 \times pq + qq)$, $\frac{1}{2}(pp + pq)$, and $\frac{1}{2}(pq + qq)$ are all integers.