Assignment 1 2012 Solution

Theodore S. Norvell

Solution Oct 10. 16:50.

Q0 [10] For integers X and Y design an efficient ($\log Y$ time) iterative algorithm to compute X^Y. (a) Complete the following proof outline

$$\{x = X \land y = Y \geq 0\} \; ? \; \left\{z = X^Y\right\}$$

You should assume that variables x, y, and z hold mathematical integers —thus there is no overflow– and that the operations assignment, +, -, $\times$, $\div$, 'odd', and 'even' each take 1 unit of time, as do any comparisons. Be sure to state the invariant of your loop.

(b) List the boolean expressions that must be shown universally true (according to the rules discussed in class)

(c) For each boolean expression from part (b), give a concise argument that it is universally true.

Solution

(a) For clarity I used $\lfloor y/2 \rfloor$ instead of $y \div 2$. This makes the rounding explicit and thus easier to deal with.

```
$\{x = X \land y = Y \geq 0\}$
$z := 1$
$\left\{I : z \times x^y = X^Y \land y \geq 0\right\}$
while $y > 0$ do
    $\left\{Q : z \times x^y = X^Y \land y > 0\right\}$
    if even($y$) then $x := x^2$    $y := \lfloor y/2 \rfloor$
    else $z := z \times x$    $y := y - 1$
    end if
end while
$\left\{R : z = X^Y\right\}$
```

(b)

0. $x = X \land y = Y \geq 0 \Rightarrow I[z : 1]$

1. $I \land y > 0 \Rightarrow Q$

2. $I \land y \leq 0 \Rightarrow R$

3. $Q \land \text{even}(y) \Rightarrow I[y : \lfloor y/2 \rfloor][x : x^2]$

4. $Q \land \text{odd}(y) \Rightarrow I[y : y - 1][z : z \times x]$

(c)

0. $I[z:1]$ is $1 \times x^y = X^Y \wedge y \geq 0$ or $x^y = X^Y \wedge y \geq 0$. The first conjunct follows from $x = X \wedge y = Y$ by the one-point law. The second conjunct is given by the precondition.

1. Given $y > 0$, I is easily rewritten to Q.

2. Given $y \leq 0$ we have

$$\begin{array}{ll} & I \\ = & \text{“since } y \leq 0\text{”} \\ & z \times x^y = X^Y \wedge y = 0 \\ = & \text{“one-point”} \\ & z \times x^0 = X^Y \wedge y = 0 \\ = & \text{“Since } x^0 = 1\text{”} \\ & x^0 = X^Y \wedge y = 0 \\ \Rightarrow & \text{“dropping the second conjunct”} \\ & R \end{array}$$

(Note that we need to assume that $0^0 = 1$. This is a reasonable assumption to make when dealing with integers.) This shows that $y \leq 0$ implies $I \Rightarrow R$. By the shunting law,[0] that's the same as $y \leq 0 \wedge I \Rightarrow R$.

3. Given that y is even we have

$$\begin{array}{ll} & I[y : \lfloor y/2 \rfloor][x : x^2] \\ = & \text{“doing the substitutions”} \\ & z \times \left(x^2\right)^{\lfloor y/2 \rfloor} = X^Y \wedge \lfloor y/2 \rfloor \geq 0 \\ = & \text{“}y\text{ is even”} \\ & z \times \left(x^2\right)^{y/2} = X^Y \wedge y/2 \geq 0 \\ = & \text{“simplifying”} \\ & z \times x^y = X^Y \wedge y/2 \geq 0 \\ \Leftarrow & \text{“if } y > 0 \text{ then } y/2 > 0\text{”} \\ & z \times x^y = X^Y \wedge y > 0 \ \ ; \end{array}$$

the last line is Q. This shows $\mathit{even}(y) \Rightarrow \big(Q \Rightarrow I[y : \lfloor y/2 \rfloor][x : x^2]\big)$; by shunting, this is equivalent to $\mathit{even}(y) \wedge Q \Rightarrow I[y : \lfloor y/2 \rfloor][x : x^2]$.

[0] The shunting law is

$$(P \Rightarrow (Q \Rightarrow R)) = (P \wedge Q \Rightarrow R)$$

It is analogous to the power law for real numbers

$$(r^q)^p = r^{p \times q} \ ,$$

which is universal (if you assume that $r^0 = 1$, for all real numbers r.) This is because the truth tables for $\wedge$ and $\Rightarrow$ are the same as the tables for the operators $\times$ and exponentiation, respectively, when true is treated as 1 and false is treated as 0.

4. Given that y is odd —although I won't use this fact— we have

$$\begin{array}{cl} & I[y : y-1][z : z \times x] \\ = & \text{"doing the substitutions"} \\ & z \times x \times x^{y-1} = X^Y \wedge y - 1 \geq 0 \\ = & \text{"simplifying"} \\ & z \times x^y = X^Y \wedge y > 0 \;\; ; \end{array}$$

the last line is Q. This shows that $odd(y) \Rightarrow (Q = I[y : y-1][z : z \times x])$, but this implies $odd(y) \Rightarrow (Q \Rightarrow I[y : y-1][z : z \times x])$ and, by shunting, this is equivalent to $odd(y) \wedge Q \Rightarrow I[y : y-1][z : z \times x]$.

Q1[10] An array a of n marbles (indexed from 0) potentially has marbles of 3 colours, green, white and pink. Unfortunately they are originally in an arbitrary order. Design an iterative algorithm (with a proof outline) that sorts the marbles so that, when it is done, all green marbles are to the left of any white and pink marbles, and all pink marbles are to the right of any white or green marbles. The only operation allowed on the array is to swap two items $a(i) :=: a(j)$.[1]

(a) Specify the problem by writing a precondition and a postcondition.

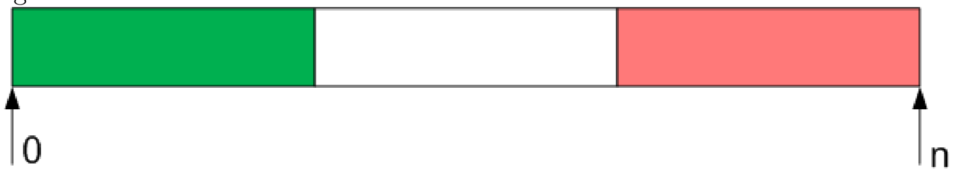
(b) State the loop invariant that you will use.

(c) State the variant that you will use.

(d) Write out a proof outline.

(e) Give an informal argument for why your initialization code establishes the loop invariant, why the loop preserves the invariant, why the postcondition is implied by the negation of the guard and the invariant, and why the loop body decreases the variant (though never below 0).

Solution: (a) In the end we want all the green on the left, all white in the middle and all pink on the right.



[1]Because I am not asking for a detailed proof of correctness, you shouldn't need a rule for showing that

$$\{P\}\ a(i) :=: a(j)\ \{Q\}$$

is correct. However, if you want one, here it is: We consider that $a(i) :=: a(j)$ is equivalent to $a := \mathsf{swap}(a, i, j)$ where swap is a function so that $\mathsf{swap}(a, i, j)$ is a sequence the same length as a and so that

$$\mathsf{swap}(a, i, j)(k) = \begin{cases} a(j) & \text{if } k = i \\ a(i) & \text{if } k = j \\ a(k) & \text{otherwise} \end{cases}$$

Now

$$\{P\}\ a(i) :=: a(j)\ \{Q\}$$

is correct if

$$\begin{array}{ccl} & & Q[a : \mathsf{swap}(a, i, j)] \\ P \Rightarrow & \wedge & 0 \leq i < a.\text{length} \\ & \wedge & 0 \leq j < a.\text{length} \end{array}$$

is universally true.

Define a Boolean function, *all*, so that, for all $i, j \in \{0, .., n\}$ and $c \in \{\text{green}, \text{white}, \text{pink}\}$,[2]

$$all(i, j, c) = (\forall k \in \{i, ..j\} \cdot a(i) = c)$$

Now let R be

$$\exists p, r \cdot 0 \le p \le r \le n \wedge all(0, p, \text{green}) \wedge all(p, r, \text{white}) \wedge all(r, n, \text{pink})$$

The specification is

$$\{true\} \ ? \ \{R\}$$

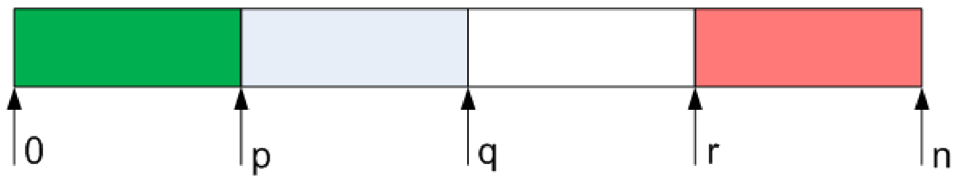
Technically we should also require that the number of each colour of marble remains unchanged, but, as the only operation on the array is to swap two marbles, this requirement will certainly be met by any program that obeys that restriction. Obviously there would be no marks off for specifying that the count of each colour remains the same. To do that I could conjoin the following to both the pre and postconditions

$$\begin{array}{rcl} GC & = & |\{i \in \{0, ..n\} \mid a(i) = \text{green}\}| \\ \wedge\, WC & = & |\{i \in \{0, ..n\} \mid a(i) = \text{white}\}| \\ \wedge\, PC & = & |\{i \in \{0, ..n\} \mid a(i) = \text{pink}\}| \end{array}$$

It would also be reasonable to put in the precondition that $GC + WC + PC = n$.

(b) I will use 3 index variables (p, q, and r) to split the array into 4 segments. The segment $a[0, ..p]$ is all green; the segment $a[p, ..q]$ may contain any colour; the segment $a[q, ..r]$ is all white; the segment $a[r, ..n]$ is all pink.

$$\begin{array}{rcl} I & : & 0 \le p \le q \le r \le n \\ & & \wedge\ all(0, p, \text{green}) \\ & & \wedge\ all(q, r, \text{white}) \\ & & \wedge\ all(r, n, \text{pink}) \end{array}$$



(c) My variant is $r + q - p$. Other answers are acceptable. It really depends on the details of your code in part (d).

[2] Using our indexing with a set notation, one could also define

$$all(i, j, c) = (a\{i, ..j\} \subseteq \{c\})$$

(d) [As was pointed out in tutorial an arguably neater solution is obtained by switching on the colour of $a[q-1]$.]

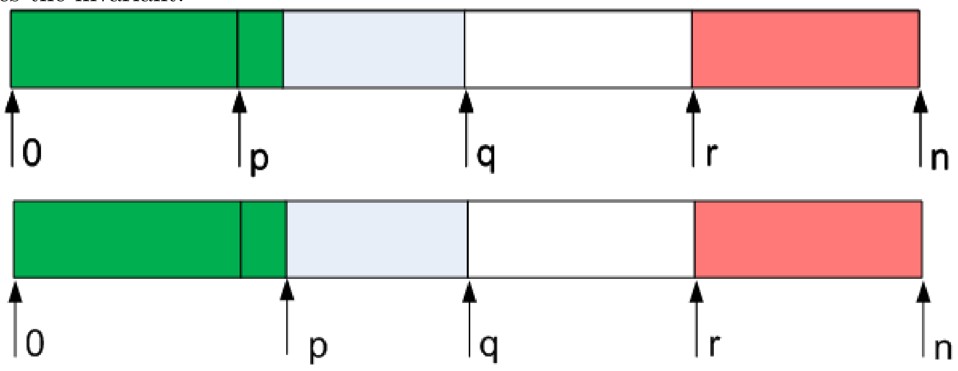
```
var $p := 0$
var $q := n$
var $r := n$
$\{I\}$
while $p < q$ do
    if $a(p) =$ green then $p := p + 1$
    elseif $a(p) =$ white then
        $a(p) :=: a(q-1)$
        $q := q - 1$
    elseif $q = r$ then
        $a(p) :=: a(r-1)$
        $q := q - 1$    $r := r - 1$
    else
        $a(p) :=: a(r-1)$
        $r := r - 1$
    end if
end while
$\{ R \}$
```

(There are several variations that will work. I chose this one because it seems to me the simplest, even though it requires a more complicated variant than some other possibilities.)

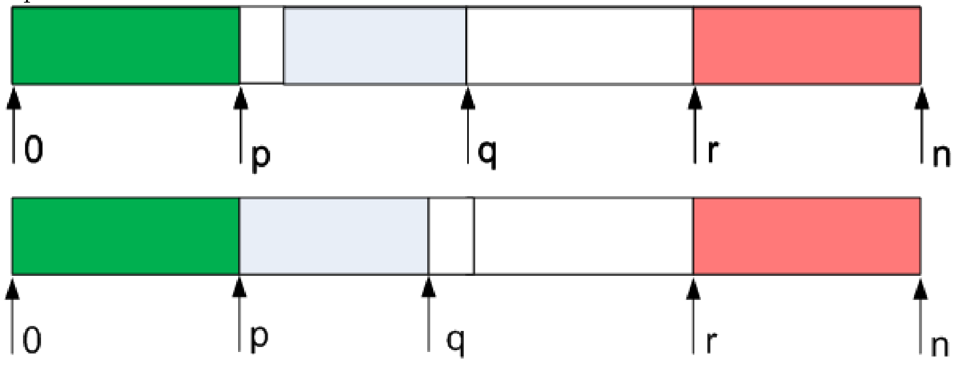
(d)

- The initialization code establishes the invariant because $p = 0 \wedge q = r = n$ implies that $\{0, ..p\} = \{q, ..r\} = \{r, ..n\} = \emptyset$ and therefore all items in those three intervals are the appropriate colour and that $0 \le p \le q \le r \le n$.
- To see that the invariant is preserved in all three cases, it is useful to look at the four cases individually

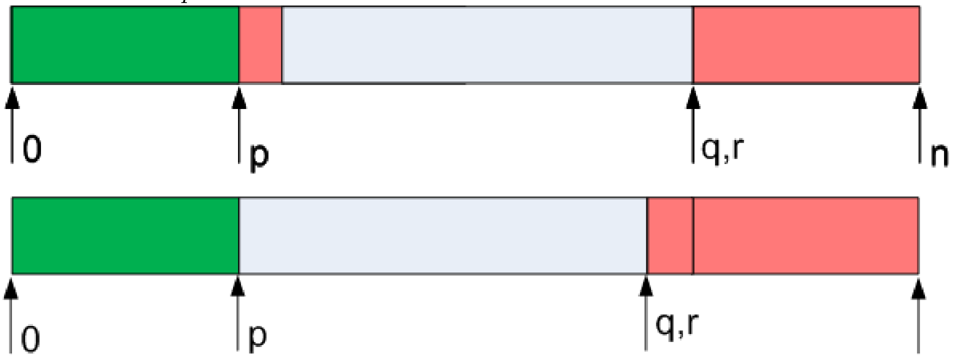
  0. When $p < q$ and $a(p) =$ green. p is incremented. The following picture shows that this preserves the invariant.



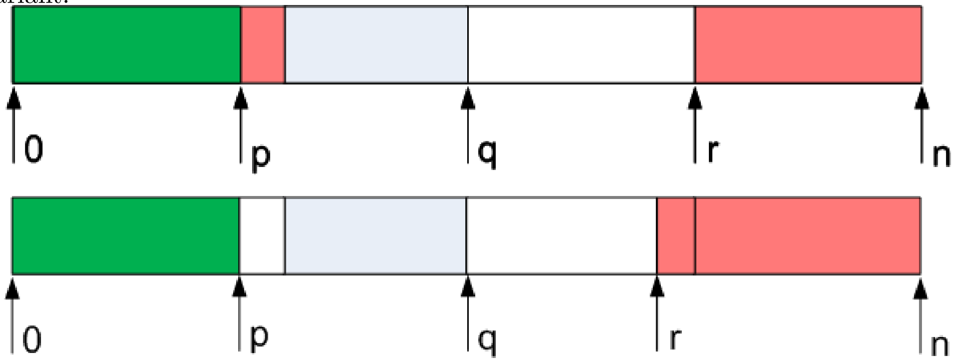
1. When $p < q$ and $a(p) =$ white, $a(p)$ is swapped with $a(q-1)$ (note that this may mean $a(p)$ is swapped with itself). Then q can be decremented. The following diagram shows that this preserves the invariant.



2. When $p < q$, $a(p) =$ pink, and $q = r$, $a(p)$ is swapped with $a(r-1)$, then both q and r are decremented. The following diagram shows that this preserves the invariant. Note that this works even if $p = r - 1$.



3. When $p < q$, $a(p) =$ pink, and $q \neq r$, $a(p)$ is swapped with $a(r-1)$ (which must be white), then r may be decremented. The following diagram shows that this preserves the invariant.



- When the loop exits we have $p = q$ and I.

$$\begin{array}{ll}
 & p = q \wedge I \\
= & \text{“one-point law} \\
 & p = q \wedge I[q : p] \\
= & \text{“making the substitution”} \\
 & 0 \le p = q \le r \le n \wedge \mathit{all}(0, p, \text{green}) \wedge \mathit{all}(p, r, \text{white}) \wedge \mathit{all}(r, n, \text{pink}) \\
\Rightarrow & \text{“dropping } q \text{ and all constraints on it”} \\
 & 0 \le p \le r \le n \wedge \mathit{all}(0, p, \text{green}) \wedge \mathit{all}(p, r, \text{white}) \wedge \mathit{all}(r, n, \text{pink}) \\
\Rightarrow & \text{“generalization”} \\
 & \exists p, r \cdot 0 \le p \le r \le n \wedge \mathit{all}(0, p, \text{green}) \wedge \mathit{all}(p, r, \text{white}) \wedge \mathit{all}(r, n, \text{pink}) \\
= & \text{“definition of } R\text{”} \\
 & R
\end{array}$$

  The last step is an example of the principle that if something is a true of a particular value, then there must exist a value for which it is true.

- The variant is $r + q - p$. Note that the invariant $0 \le p \le q \le r$ implies this is nonnegative.

  0. When $p < q$ and $a(p) = \text{green}$. p is incremented, which decreases the variant by 1.
  1. When $p < q$ and $a(p)$ is white, q is decremented, which decreases the variant be 1.
  2. When $p < q$, $a(p) = \text{pink}$, and $q = r$, both q and r are decremented. This decreases the variant by 2.
  3. When $p < q$, $a(p) = \text{pink}$, and $q \ne r$, r is decremented. This decreases the variant by 1.