Introduction to the theory of computation
week 6 (chapter 2 of Course Notes)

21st June 2005

- Note (under web page announcements) that your A1 mark is increased by 6/59.
- A2 is due Thursday at 10 am, in drop box or electronic submission of a PDF file.

**Binary search example, continued...**

Last time we were in the midst of proving the following loop invariant, which we believe will help us to prove binary search correct with respect to its specifications:

Let $P(i)$ be “If the precondition of binSearch is satisfied and the loop as at least $i$ iterations, then $0 \leq f_i \leq l_i \leq A.length - 1$ and $(t_x \in [f_i, l_i]) \land x \not\in A$, where $t_x$ denotes the lowest index, if it exists, such that $A[t_x] == x$.”

**Claim:** $P(i)$ is true for all $i \in \mathbb{N}$.

**Proof (induction on $i$):** When $i = 0$, $P(i)$ states that $0 \leq f_0 \leq l_0 \leq A.length - 1$ and $(t_x \in [f_0, l_0]) \land x \not\in A$. Well, according to the program $f_0 = 0$ and $l_0 = A.length - 1$, so the claim holds for the base case.

**Induction Step:** Assume that $P(i)$ is true for some arbitrary natural number $i$. I need to show that this implies $P(i+1)$. If there is no $(i + 1)$th iteration of the loop, then $P(i + 1)$ is vacuously true. Otherwise, the loop did not exit after iteration $i$, which means that $f_i < l_i$, since by $P(i)$ $f_i \leq l_i$ and (by non-exiting) $f_i \neq l_i$. Recall that $m_{i+1} = (f_i + l_i)/2$. Thus

$$m_{i+1} = \frac{f_i + l_i}{2} \geq (f_i + f_i)/2 = f_i$$

on the other hand

$$m_{i+1} = \frac{f_i + l_i}{2} \leq \frac{(l_i - 1 + l_i)}{2} = \frac{|(2l_i - 1)/2|}{2} = \frac{|l_i - \frac{1}{2}|}{2}$$

so $f_i \leq m_{i+1} < l_i$.

(full solution here)

Now combine the loop invariant with the assumption that binSearch terminates:
CLAIM: The precondition plus termination imply the postcondition.$^2$

This proves partial correctness. To prove termination, you need to make precise your intuition that if the gap from $f$ to $l$ gets steadily smaller, but never less than zero, eventually you must have $f = l$. In symbols this says that if $g_i = l_i - f_i$, then $g_i$ is always non-negative, and if $g_{i+1}$ exists, then $g_i > g_{i+1}$. If you can prove these two things, then the sequence $(g_0, g_1, \ldots)$ is finite (by well-ordering it has a smallest element $g_k$, and hence no $g_{k+1}$), so there are finitely many loop iterations. Since $l_i$ and $f_i$ are integers, so is their difference, and (by the loop invariant proved above) $l_i \geq f_i$, so $g_i$ is a natural number. It remains to show that $(g_i)$ is strictly decreasing.

CLAIM: If the loop is executed at least $i + 1$ times, then $g_{i+1} < g_i$.$^3$

Notice that nowhere did we claim that the loop exit condition would be satisfied. In general, reasoning that some function of the loop index defines a strictly decreasing sequence of natural numbers is easier than directly showing that the loop condition is eventually satisfied.

CLAIM: Suppose the precondition is satisfied. Then $\text{binSearch}(A, x)$ terminates.$^4$

**Binary multiplication**

Multiply binary numbers $(m = 101) \times (n = 11)$ (using distributivity) is the same as $1 \times 11 + 00 \times 11 + 100 \times 11$ (the same product in base 10 translates into: $5 \times 3$ equals $1 \times 3 + 0 \times 3 + 4 \times 3$). This is the same as our usual algorithm for multiplying numbers in base 10.

\[
\begin{array}{c}
11 \\
\times 101 \\
\hline
- - - - - \\
\end{array}
\]

\[
\begin{array}{c}
z_0 = 0 \\
z_1 = 1 \times 11 = z_0 + 2^0 \times 1 \times 11 \\
z_2 = 01 \times 11 = z_1 + 2^1 \times 0 \times 11 \\
z_3 = 101 \times 11 = z_2 + 2^2 \times 1 \times 11 \\
\hline
1111
\end{array}
\]

The program $\text{mult}(m, n)$ assumes that you can quickly multiply and divide by 2 (perhaps using left-shift and right-shift), and then implements multiplication of natural number $m$ by integer $n$ (see program listing Example.java). (By the way, Java integer division / and mod operator % don’t match our definition from Chapter 1 for negative arguments).

At every iteration of the loop, $z_i$ holds our result so far. Examining the algorithm, $z_i$ holds $n$ times the right-most $i$ bits of $m$ (write this out). In symbols:

\[
z_i = n \times (\text{right-most } i \text{ bits of } m)
\]

\[
= n \times (m - (m/2^i) \times 2^i)
\]

\[
= nm - (m/2^i)n2^i
\]

\[
= nm - x_i y_i
\]

We state (and prove) this invariant below. However (since it is easier) we prove termination of $\text{mult}(m, n)$ first.

CLAIM: If the loop is iterated at least $i + 1$ times, then $x_i > x_{i+1}$.$^5$
CLAIM: The loop in \( \text{mult}(m, n) \) terminates.\(^6\)

CLAIM: \( P(i) \): “If the loop has \( i \) iterations, then \( z_i = mn - x_iy_i \)” is true for all \( i \in \mathbb{N} \).\(^7\)

CLAIM (PARTIAL CORRECTNESS): Suppose the precondition holds and \( \text{mult}(m, n) \) terminates. Then, when it terminates, the postcondition holds.\(^8\)

Thus we’ve proved termination and then partial correctness of \( \text{mult}(m, n) \). The hard work was coming up with the appropriate loop invariant.
Notes

1 Proof (induction on i): P(0) states that if the precondition of binSearch is satisfied, and the loop has at least i iterations, then \(0 \leq f_0 \leq l_0 \leq A.length - 1\) and \((t_x \in [f_i, l_i]) \lor x \notin A\). Inspecting the program we see that \(f_0 = 0\) and \(l_0 = A.length - 1\), so the first part of the invariant is true, and either \(t_x \in [0, A.length - 1]\) or else \(x \notin A\), so the claim holds for the base case.

**Induction Step:** Assume that \(P(i)\) holds for some arbitrary natural number \(i\). If there is no \((i + 1)th\) iteration, then \(P(i + 1)\) holds vacuously (empty antecedent). Otherwise, \(f_i \neq l_i\), so (since \(f_i \leq l_i\)) we must have \(f_i < l_i\). This means that

\[
m_{i+1} = \frac{(f_i + l_i)}{2}
\]

integer division monotonic:

\[
\geq \frac{(f_i + f_i)}{2} = f_i
\]

... and you also

\[
m_{i+1} = \frac{(f_i + l_i)}{2}
\]

\[
\leq \frac{(l_i - 1 + l_i)}{2}
\]

integer division floors real division

\[
= \lfloor\frac{l_i - 1}{2}\rfloor
\]

\[
< l_i
\]

So \(f_i \leq m_{i+1} < l_i\), and we need to consider two cases.

1. If \(A[m_{i+1}] \geq x\), then you set \(f_{i+1} = f_i \leq m_{i+1} = l_{i+1}\), and so \(0 \leq f_{i+1} \leq l_{i+1} \leq A.length - 1\), as wanted. If \(t_x\) exists, we must have \(t_x \leq m_{i+1} = l_{i+1}\), since array \(A\) is sorted, and by \(P(i), t_x \in [f_i, l_i]\), so \(t_x \geq f_i = f_{i+1}\). Thus either \(t_x \in [f_{i+1}, l_{i+1}]\) or \(x \notin A\).

2. If \(A[m_{i+1}] < x\), then you set \(f_i < f_{i+1} = m_{i+1} + 1 \leq l_{i+1} = l_i\), so \(0 \leq f_{i+1} \leq l_{i+1} \leq A.length - 1\), as wanted. If \(t_x\) exists we must have \(t_x \geq m_{i+1} + 1 = f_{i+1}\), since the array \(A\) is sorted, and by \(P(i)\) \(t_x \in [f_i, l_i]\), so \(t_x \leq l_i = l_{i+1}\). Thus either \(t_x \in [f_{i+1}, l_{i+1}]\) or \(x \notin A\).

In both cases, the two invariants hold, so we have shown that \(P(i) \Rightarrow P(i + 1)\), and we conclude that \(P(i)\) holds for all \(i \in \mathbb{N}\). QED.

2 Proof: Suppose binSearch terminates at the end of the \(kth\) loop iteration. Examination of the loop condition implies that \(f_k = t_k\). The loop invariant, \(P(k)\), implies that either \(t_x \in [f_k, f_k]\), in which case binSearch returns \(t_x = f_k\), and \(A[t_x] = x\), or else \(x \notin A\), and (since \(0 \leq f_k \leq A.length - 1\) implies that \(f_k\) is a valid index for \(A\), \(A[f_k] \neq x\), so binSearch \((A, x)\) returns \(A.length\). In either case binSearch \((A, x)\) satisfies the postcondition, as claimed. QED.

3 Proof: Suppose the loop iterates at least \(i + 1\) times. Since it doesn’t terminate at the end of loop \(i\), we must have \(f_i < l_i\), so (by result in loop invariant) \(f_i \leq m_{i+1} < l_i\). If \(A[m_{i+1}] \geq x\) then (by the program) \(f_{i+1} = f_i\) and \(l_{i+1} = m_{i+1}\), and so \(g_{i+1} = l_{i+1} - f_{i+1} = m_{i+1} - f_i < l_i - f_i = g_i\), and the claim holds. If
Proof: The sequence \( \langle g_i \rangle \) is composed of natural numbers, since \( l_i \) and \( f_i \) are integers with \( l_i \geq f_i \) by the loop invariant. The set of values in \( \langle g_i \rangle \) form a non-empty subset of \( \mathbb{N} \) (containing at least \( l_0 - f_0 \)), and hence have a smallest element \( g_k \). Since (by the previous claim) \( \langle g_i \rangle \) is strictly decreasing, \( g_k \) is also the last element, hence there are no more than \( k \) loop iterations and \( \text{binSearch}(A, x) \) terminates. QED.

Proof: Since \( x_0 = m \) is assumed (by the precondition) to be a natural number, repeated integer division yields natural number quotients so (by a short induction proof omitted here), \( x_i \) is a natural number. If there is an \( (i + 1) \)th iteration of the loop, then \( x_i \neq 0 \) implies \( x_i > 0 \) (natural numbers are non-negative), so we have \( 2x_i > x_i \), (add \( x_i \) to both sides), which in turn implies \( x_i > x_i / 2.0 \geq \lfloor x_i / 2.0 \rfloor = x_{i+1} \). QED.

Proof: The \( i \)th iteration of the loop is associated with natural number \( x_i \). By the previous claim the sequence \( \langle x_i \rangle \) is strictly decreasing sequence in \( \mathbb{N} \), and hence (PWO) finite. Call the last element of the sequence \( x_k \), in other words, there is no element \( x_{k+1} \). Thus the loop does not iterate \( k + 1 \) times, so it must terminate. QED.

Proof (induction on \( i \)): If \( i = 0 \) then \( P(0) \) asserts that \( x_0 = 0 = mn - x_0 y_0 = mn - mn \), which is clearly true, so the base case holds.

**Induction Step:** Let \( i \) be an arbitrary natural number, and assume that \( P(i) \) holds. I must use this assumption to show that \( P(i + 1) \) holds. If the loop does not have \( i + 1 \) iterations, there is nothing to prove. Otherwise, there are two cases to consider, depending on whether \( x_i \) is even or odd:

**Case 1:** \( x_i \mod 2 = 0 \), so \( x_i = 2x_{i+1} \), and

\[
\begin{align*}
z_{i+1} &= z_i \\
by \text{program} & = & z_i \\
y_i &= y_{i+1}/2.0 \\
by \text{IH} & = & mn - x_i y_i \\
\end{align*}
\]

as claimed.

**Case 2:** \( x_i \mod 2 = 1 \), so \( x_i = 2x_{i+1} + 1 \), and

\[
\begin{align*}
z_{i+1} &= z_i + y_i \\
by \text{program} & = & z_i + y_i \\
y_i &= y_{i+1}/2.0 \\
by \text{IH} & = & mn - x_i y_i + y_i \\
\end{align*}
\]

as claimed.

in both cases \( P(i + 1) \) holds, so \( P(i) \Rightarrow P(i + 1) \). I conclude that \( P(i) \) holds for all \( i \in \mathbb{N} \).

Proof: Suppose the precondition holds and \( \text{mult}(m, n) \) terminates at the \( k \)th iteration of the loop. By the exit condition \( x_k = 0 \), and by \( P(k) \), \( x_k = mn - x_k y_k = mn \). The program returns the value \( z_k = mn \), which is what the postcondition claims. QED.