CS240: Homework 4

Sahit Mandala

October 16, 2015

Problem 1

P(n) = $\forall$ good sequences generated from n applications of the constructor rules upon the foundational rule, the number of 0's in the string equals the number of 1's

Note that $P(0)$ considers only the empty sequence (e.g. the foundational rule construction).

We shall prove $\forall n \in \mathbb{N}, P(n)$ using strong structural induction.

Base case $P(0)$: At iteration 0, the only good sequence to consider is the empty sequence, which generated by the foundational rule. Note that the number of 0's is 0 and the number of 1's is 0, so clearly the number of 0's and 1's are equal. Hence, we have shown the base case $P(n)$ at $n = 0$.

Next, assume the strong inductive hypothesis: Given some $n \in \mathbb{N}$, $\forall k \in \mathbb{N}$ s.t. $x \leq n$, $P(k)$. That is, every good sequence generated by k iterations of the constructor rule, with $\forall k \in \mathbb{N}$ s.t. $x \leq n$, has the same number of 1's and 0's. Let us define the set S of all good sequences generated by some $m \in \mathbb{N}$ applications of the constructor rule, with $m \leq n$. Then for all sequences s in S, s has the same number of 1's and 0's by the strong inductive hypothesis, and S contains every good sequence generated by up to n applications of the constructor rule.

We want to now show that $P(n+1)$ follows. Let some good sequence generated from $n+1$ applications of the constructor rules upon the foundational rule be given, which we shall call s_{n+1}. Then s_{n+1} was generated using some 2 elements in S using one of the constructor rules. We shall consider every possible construction of s_{n+1}:

Case 1:

Suppose that s_{n+1} has the form $0s1t$ for some $s, t \in S$.

We know that s,t each have the same number of 0's, 1's. So lets say i_s = # of 0's in s = # of 1's in s and i_t = # of 0's in t = # of 1's in t.

Notice that the # of 0's in s_{n+1} = the # of 0's in $0s1t$ = 1 + # of 0's in s + 0 + # of 0s in t = $1 + i_s + i_t$, since the number of 0's is additive across subsequences and the sequences 0,1 clearly have 1,0 0's respectively. Furthermore, the # of 1's in s_{n+1} = the # of 1's in $0s1t$ = 0 + # of 1's in s + 1 + # of 1's in t = $i_s + 1 + i_t$, by similar logic. But clearly, $1 + i_s + i_t = i_s + 1 + i_t$, so s_{n+1} has the same number of 0's and 1's

Case 2:

Suppose that s_{n+1} has the form $1s0t$ for some $s, t \in S$.

Again, We know that s,t each have the same number of 0's, 1's. So lets say i_s = # of 0's in s = # of 1's in s and i_t = # of 0's in t = # of 1's in t.

Notice that the # of 0's in s_{n+1} = the # of 0's in $1s0t$ = 0 + # of 0's in s + 1 + # of 0s in t = $i_s + 1 + i_t$, again noting that the number of 0's is additive across subsequences. Furthermore, the #

of 1's in s_{n+1} = the # of 1's in $1s0t = 1 +$ # of 1's in $s + 0 +$ # of 1's in t $= 1 + i_s + i_t$, by similar logic. But clearly, $1 + i_s + i_t = i_s + 1 + i_t$, so s_{n+1} has the same number of 0's and 1's.

Overall, we have shown that s_{n+1} has the same number of 0's and 1's. Because s_{n+1} was any arbitrary good sequence generated from $n+1$ applications of the constructor rules upon the foundational rule, we have shown all good sequences generated by $n+1$ applications of the constructor rule have the same number of 0's and 1's. Whence, we have shown $P(n+1)$, from which it follows that $\forall n \in \mathbb{N}, P(n)$ holds.

Problem 2

Part A

Let our loop invariant be: P(m) = on the m^{th} iteration of the loop on line (2), $0 \leq i \leq n$ and $x \notin A[0,(i-1)]$.

First, lets consider the case that A is empty. Then $n = 0$. Then, during the 0th iteration, $i = 0$ and clearly, $x \notin A[0,-1] = []$. So then $i = n$, which means the loop at (2) is skipped entirely. Hence, the loop invariant holds (for the only iteration, 0).

Now we shall show induction on m in P(m), assuming that the array A is nonempty.

Base case $P(0)$: On the 0th iteration, the loop has iterated 0 times, so $i = 0$ from line (1). Since we assumed A is nonempty, $0 \leq n$, so $0 \leq i \leq n$. Also, $x \notin A[0, 0-1] = A[0,-1] = []$, the empty subarray. Hence, we have shown $P(0)$.

Inductive Hypothesis: Next, suppose that $P(m)$ for some $m \in \mathbb{N}$. That is, on the m^{th} iteration, $0 \leq i \leq n$ and $x \notin A[0,(i-1)]$. We want to show $P(m+1)$.

If the $m + 1^{th}$ iteration does not occur, then we are done.

So assume that the $m + 1^{th}$ iteration occurs. Because this iteration occurs, we know that the equality on (2) was satisfied, so $i < n$. If the conditional on (3) was triggered, then the return statement is triggered, so the loop does not complete. Because we are considering the case that this iteration completes, we can infer that this conditional is not satisfied, so $x \neq A[i]$. Since $x \notin A[0,(i-1)]$ and $x \notin A[i]$, then $x \notin A[0,i]$. Furthermore, after (4), i is incremented to $i+1$. This completes the iteration of the loop. Since we know $i < n$ and both are integral values, then $i + 1 \leq n$; and clearly, $0 \leq i + 1$ since $0 \leq i$. So overall, after the $m + 1^{th}$ iteration, $0 \leq i + 1 \leq n$ and $x \notin A[0,i] = A[0,(i+1)-1]$. Hence, we have shown $P(m+1)$ and thus, overall, have shown that $\forall m \in \mathbb{N}$, the loop invariant $P(m)$ holds.

Part B

Partial correctness:

We assume that the program terminates and go on to show that whenever the program terminates, we get a correct solution. There are 2 possible case of termination:

Case 1:

Suppose the program terminates at (5). Because we return -1, we want to show that $x \notin A[0,n-1]$ (that is, x is not in the array A). Reaching (5) implies that the loop conditional was not satisfied after some k^{th} iteration of the loop, with $k \in \mathbb{N}$. By our loop invariant, we know after that iteration, $0 \leq i \leq n$ and $x \notin A[0,i-1]$. Because the loop conditional was not satisfied, $i \geq n$ (since $i \not< n$). This implies that $i = n$, from which it follows that $x \notin A[0,i-1] = A[0,n-1]$. So x is in fact not in the entire array. Whence, we have shown correctness when terminating at line (5).

Case 2: Suppose that the program terminates at line (3). That is, we return i. We want to show that i is the smallest index such that $x = A[i]$. Reaching (3) implies that the if conditional was satisfied

after some k^{th} iteration of the loop, with $k \in \mathbb{N}$ (so we reach (3) on the $k+1^{th}$ [partial] iteration of the loop). After this k^{th}iteration, we know that $0 \leq i \leq n$ and $x \notin A[0, i-1]$. Again, because the if conditional was satisfied, we know that $x = A[i]$, which clearly shows $x \in A[0, n-1]$. Because $x \notin A[0, i-1]$, we know that $\forall j \in \mathbb{N}$ with $0 \leq j \leq i-1$, $x \neq A[j]$. That is, for all indices j smaller than i , $A[j] \neq x$. So clearly, i is the smallest index satisfying $x = A[i]$.

Overall, we have shown that when the program terminates, we have a correct solution in either case.

Termination:

We know that $i = 0$ and are given some $n \in \mathbb{N}$, A, x. To prove termination, we can show that this loop always terminates in at most n iterations of the loop. I claim that after the m^{th} iteration of the loop for any $m \in \mathbb{N}$, the program terminated or $i = m$. We shall use induction on m in $P(m) := (i = m)$

Base case $m = 0$: On the 0th iteration, $i = 0 = m$.

Inductive hypothesis: Let some $k \in \mathbb{N}$ be given. Assume $P(k)$. That is after the k^{th} iteration, $i = k$ or the program terminated.

We now want to show that $P(k+1)$ holds. That is, the value of i, say i', after the $k+1^{th}$ iteration satisfies $i' = k+1$

If the $k+1^{th}$ iteration does not occur (including if the k^{th} iteration terminated) or the program terminates on (3) of the $k+1^{th}$ loop, we are done.

Now suppose that the $k+1^{th}$ iteration does occur but does not terminate on (3). Then we know $i = k$ from the $P(k)$ (after all, the k^{th} iteration did not terminate). During the $k+1^{th}$ iteration, line (3) does not change the value of i, so we can ignore it. Note that we will not trigger (3) by our assumption that the $k+1$ iteration does not terminate here. Then, in line (4), i is incremented by i, and since $i = k$, $i' = i+1 = k+1$ is the value after the $k+1^{th}$ iteration, as expected. Hence, we have shown by induction that after the k^{th} iteration, $i = k$ or the program terminated.

To utilize this lengthy theorem, first note that if the loop terminates on the m^{th} iteration for some $m \leq n$ (here, we expect termination on (3), but that doesn't explicitly matter), we are done. So consider the case that $\forall m \in \mathbb{N}$ with $m \leq n$, the m^{th} iteration does not terminate. Then, after the n^{th} iteration, we know that $i = n$ after the loop completes. So then the loop conditional, $i = n \not< n$ is not satisfied, and the loop is completed, causing the program to terminate on (5). Ennumerating all cases, this shows that the program always terminates in at most n (a finite number) iterations.