CS109A Midterm Solutions

Problem 1: (20 points) Prove by induction on n that for all $n \geq 1$, the sum

$$1 \times 2 + 2 \times 3 + 3 \times 4 + \cdots + n \times (n+1)$$

(that is, $\sum_{i=1}^{n} i(i+1)$) equals $n(n+1)(n+2)/3$. To organize your proof, answer the following questions.

a) What is the statement $S(n)$ to be proved? *Ans.*: $\sum_{i=1}^{n} i(i+1) = n(n+1)(n+2)/3$. *Note*: Many people added or preceded this answer by "for all n." That is technically wrong. This phrase, in logic, has the effect of making n a local variable, different from the n in "$S(n)$," just as the sum over i makes i local to the sum and not defined outside it. We'll discuss this issue in Chapter 14. I didn't deduct for this.

b) What is a suitable basis case? *Ans.*: $n = 1$.

c) Prove the basis case. *Ans.*: $\sum_{i=1}^{1} i(i+1) = 2$. Also, $(1)(1+1)(1+2)/3 = 2$. Thus, both sides of the equation $S(1)$ have the same value, and so $S(1)$ is true.

d) When we prove the inductive step, we assume $S(n)$. What must we prove, using this assumption? *Ans.*: $\sum_{i=1}^{n+1} i(i+1) = (n+1)(n+2)(n+3)/3$.

e) Prove the inductive step. *Ans.*:

$$\begin{array}{lcll} \sum_{i=1}^{n+1} i(i+1) & = & \sum_{i=1}^{n} i(i+1) + (n+1)(n+2) & \\ & = & n(n+1)(n+2)/3 + (n+1)(n+2) & \text{By Ind. Hyp.} \\ & = & (n+1)(n+2)(n+3)/3 & \text{By algebra} \end{array}$$

Notes: Not a few people treated $S(n)$ as if it were the sum $\sum_{i=1}^{n} i(i+1)$, rather than a statement about n, as the problem clearly defines it. You typically lost 6 points if you did that more than once. Also, I deducted 3 points for failing to state where you were using the inductive hypothesis. I know many people knew quite well what they were doing, but if you're going to explain your proof properly, you really need to point out when you invoke the inductive hypothesis.

Problem 2: (15 points) A certain class of binary trees $T_0, T_1, T_2, \ldots$ is defined recursively by the following:

BASIS: T_0 is an empty tree, and T_1 is a tree with a single node.

INDUCTION: For all $n \geq 2$, T_n consists of a root node whose left subtree is a copy of T_{n-1}, and whose right subtree is a copy of T_{n-2}.

a) In the space below, draw T_2, T_3, and T_4.

T_2 T_3 T_4

b) Let $N(n)$ be the number of nodes in binary tree T_n. Write a recurrence relation for $N(n)$. You need not solve the recurrence. *Ans.*:

$N(0) = 0$
$N(1) = 1$
$N(n) = 1 + N(n-1) + N(n-2)$ for $n \geq 2$

Notes: Some people forgot the "+1" in the recursion and lost 3 points. I deducted one point if you neglected to explain that the recursion only applied for $n \geq 2$. If you omitted the basis cases, you lost 3 points.

Problem 3: (15 points) Write an ML function `getith(i,L)` that takes an integer i and a list L, and returns the ith element on list L. To simplify the problem, you may assume that i is at least 1, and that i is no larger than the length of the list L, so that there *is* an ith element on the list. *Ans.*:

```
fun getith(1,x::xs)  = x
|   getith(i,x::xs)  = getith(i-1,xs);
```

Notes: Some people erred by including a rule like `getith(i,[])=[]`. That will cause a type error if the type of elements is not some sort of list type. I was disappointed by, but did not take off for, circumlocutions like `let val j = i-1 in getith(j,L)` in place of the simpler `getith(i-1,L)`.

Problem 4: (20 points) Consider the recurrence

BASIS: $T(1) = 0$.

INDUCTION: $T(n) = \frac{1}{2}T(n-1) + 1$ for integers $n > 1$.

Note that since there are no symbolic constants, this is not a "big-oh" problem; we can solve for $T(n)$ exactly.

a) To begin, what are the initial values of $T(n)$? Ans: $T(1) = 0$; $T(2) = 1$; $T(3) = 1.5$; $T(4) = 1.75$; $T(5) = 1.875$.

b) Expand the inductive rule so $T(n)$ is expressed in terms of $T(n-2)$. *Ans.*: $T(n) = T(n-2)/4 + 3/2$.

c) Express $T(n)$ in terms of $T(n-3)$. *Ans.*: $T(n) = T(n-3)/8 + 7/4$.

d) What is the general pattern? That is, express $T(n)$ in terms of $T(n-i)$. *Ans.*: $T(n) = T(n-i)/2^i + 2 - 2^{-(i-1)}$.

e) For what value of i can we eliminate $T(n-i)$ from the expression? *Ans.*: $i = n - 1$.

f) Use your answer to (e) to express $T(n)$ as a function of n alone (i.e., without any terms involving the function T).

$T(n) = 2 - 2^{-(n-2)}$.

Note: A number of people failed to expand their answers to (d) and/or (f); some even did not simplify (b) and (c). You lost 3 points for each offense. Leaving your answer as a sum is really no better than not solving it at all, since I can get the answer by applying the recurrence n times just as easily as I can sum n terms (more or less).

Problem 5: (15 points) We say n_0 and c are *witnesses* to the fact that $f(n)$ is $O(g(n))$ if for all $n \geq n_0$ it is true that $f(n) \leq cg(n)$ (i.e., a big-oh proof "works" with n_0 and c as the constants).

a) If $n_0 = 1$, what is the smallest value of c such that n_0 and c are witnesses to the fact that $(n+2)^2$ is $O(n^2)$? *Ans.*: $c = 9$.

b) If $c = 5$, what is the smallest nonnegative integer n_0 such that n_0 and c are witnesses to the fact that $(n+2)^2$ is $O(n^2)$? *Ans.*: $n_0 = 2$.

c) If $n_0 = 0$, for what values of c are n_0 and c witnesses to the fact that $(n+2)^2$ is $O(n^2)$? *Ans.*: There are no such values of c.

Problem 6: (15 points) Let us consider three-node binary trees whose nodes are each labeled by 3, 5, or 8, and no two nodes have the same label. For the questions below, the "priority" of a label is the same as the value of that label.

a) How many of these trees are balanced, partially-ordered trees? *Ans.*: 2. Draw them in the space below.

8 8

3 5 5 3

b) How many such trees have the partially-ordered tree property (POT property), but are *not* balanced? *Ans.*: 4. Draw them in the space below.

8 8 8 8

5 5 5 5

3 3 3 3

Note: A number of people regarded the first of the four above as balanced. That is false, since the middle row is not complete; the right child of the root is missing.