Homework 01

CS 624, 2022 Fall

Read the updated course homework policies before you start!

1. Consider the following algorithm for calculating the *cumulative sums* of an array. The input is an array of numbers, `A`. The output is a new array of number of the same length, `R`. (Array indexes start at 1.)

```
CumulativeSums(A) :=

R ← new array with length[A] elements
if length[A] > 0
  R[1] ← A[1]
end if
for j ← 2 to length[A]
  R[j] ← R[j-1] + A[j]
end for
return R
```

The correctness property for this algorithm is the following:

$$\texttt{R[n]} = \sum_{i=1}^{n} \texttt{A[i]} \qquad \text{for all } 1 \leq n \leq \texttt{length[A]}$$

   (a) Prove that this algorithm terminates.

   (b) State the loop invariant for the `for` loop.

   (c) Prove the correctness of the algorithm using the loop invariant.

   (d) What is the running time of this algorithm? Justify your answer.

2. Prove that if $f = O(g)$ and $g = O(h)$, then $f = O(h)$.

3. Problem 3-4 (a, b, c, d) in the textbook (page 62).

> Let f and g be asymptotically positive functions. Prove or disprove each of the following conjectures:
>
> (a) $f(n) = O(g(n))$ implies $g(n) = O(f(n))$.
>
> (b) $f(n) + g(n) = \Theta(\min(f(n), g(n)))$.
>
> (c) $f(n) = O(g(n))$ implies $\lg(f(n)) = O(\lg(g(n)))$, where $\lg(g(n)) \geq 1$ and $f(n) \geq 1$ for all sufficiently large n.
>
> (d) $f(n) = O(g(n))$ implies $2^{f(n)} \in O(2^{g(n)})$.

4. Problem 4-1 (a, b, f, g) in the textbook (page 107).

> Give asymptotic upper and lower bounds for $T(n)$ in each of the following recurrences. Assume that $T(n)$ is constant for $n \leq 2$. Make your bounds as tight as possible, and justify your answers.

(a) $T(n) = 2T(n/2) + n^4$.

(b) $T(n) = T(7n/10) + n$.

(f) $T(n) = 2T(n/4) + \sqrt{n}$.

(g) $T(n) = T(n-2) + n^2$.

5. Problem 4.2 in Lecture notes 1 (`[aux01]`), page 7.

If there are positive constants a and c such that

$$T(n) = \sum_{j=2}^{n} (a + (j-1)c)$$

then there are constants A, B, and C such that

$$T(n) = An^2 + Bn + C$$

Of course A, B, and C depend on a and c, but to not depend on n. You should also show that $A > 0$. That's an important fact.

6. Let a binary tree be either NIL or a node with left and right attributes whose values are also binary trees. Define the mindepth function as follows:

$$\text{mindepth}(t) = \begin{cases} 0 & \text{if } t = \text{NIL} \\ 1 + \min(\text{mindepth}(\text{left}(t)), \text{mindepth}(\text{right}(t))) & \text{otherwise} \end{cases}$$

and define the countnil function as follows:

$$\text{countnil}(t) = \begin{cases} 1 & \text{if } t = \text{NIL} \\ \text{countnil}(\text{left}(t)) + \text{countnil}(\text{right}(t)) & \text{otherwise} \end{cases}$$

Prove the following: If $\text{mindepth}(t) \geq n$, then $\text{countnil}(t) \geq 2^n$.

Hint: Use induction on n.