CSC236 winter 2020, quiz week 4_1

first/given name: **last/family name:**

utorid:

Use the technique of unwinding to find a closed form for the following recurrence $T(n)$, assuming n is a power of 2:

$$T(n) = \begin{cases} 3 & \text{if } n = 1 \\ 2 + T(n/2) & \text{if } n > 1 \end{cases}$$

Your closed form should be exactly equal to $T(n)$ (do not modify any of the constants in the definition). It should not recursively refer to T, and ideally should not involve any summations (using Σ notation or "...").

You may use algebraic substitution and/or tree diagrams. You do not need to prove your closed form correct, but it should be reasonably clear from what you've written how you arrived at your closed form. It's recommended (but not required) to check your closed form on a small value of n to verify that it works.

Solution

$$\begin{aligned} T(n) &= 2 + T(n/2) \\ &= 2 + (2 + T(n/4)) \\ &= 2 + (2 + (2 + T(n/8))) \\ &= 2 + 2 + 2 + ...(2 + T(n/n)) \\ &= 2 + 2 + 2 + ...2 + 3 \end{aligned}$$

Based on this pattern, I can see that there will be $\log n$ '2's in my sum. The final term, $T(n/n)$ is equal to 3, by definition. So

$$T(n) = 2 \log n + 3$$

Tree method alternative: The call tree for this $T(n)$ should look like a straight line of nodes (like the binary search example in the exercises). Each node would be labelled as taking 2 steps of non-recursive work, except the leaf, which is 3 steps. Observing that the nodes follow a progression of dealing with input sizes n, $n/2$, $n/4$, ... I would subrmise that there are $\log n$ internal nodes (times 2 steps), plus one leaf node (representing 3 steps), for a total of $T(n) = 2 \log n + 3$.