CS 457, Data Structures and Algorithms I
Second Problem Set

October 8, 2019

Due on October 16. Collaboration is not allowed. Contact Daniel and me for questions.

1. (24 pts) Solve the following recurrence equation (tight upper *and* lower bounds!)

$$T(n) = \begin{cases} 1 & \text{if } n < 1 \\ 2T(n/2) + 23 & \text{otherwise.} \end{cases}$$

   a) Using the *master theorem*. Make sure to explain which case applies and why.

   b) Using the *recursion tree* method. Do not use asymptotic notation for the depth of the recursion tree; use exact upper and lower bounds instead.

   c) Using the *substitution* method. Make sure to show that the boundary conditions hold as well; choose the constants c and n_0 appropriately.

2. (30 pts) For the following algorithm calls, prove a *tight* asymptotic bound for their worst-case running time. Pay attention to the input in the algorithm *calls*!

   a) The call to RECURSIVE-ALGORITHM(n) for some $n > 1$.

   [H] RECURSIVE-ALGORITHM(a) $q = 0$ $a \geq 1$ $i = 1$ $\lfloor a \rfloor$q=q+1 RECURSIVE-ALGORITHM($a/2$) RECURSIVE-ALGORITHM($a/5$) RECURSIVE-ALGORITHM($a/9$)

   b) The call to RECURSIVE-ALGORITHM2(n, n) for some $n > 2$.

   [H] RECURSIVE-ALGORITHM2(a, b) $a \geq 2$ and $b \geq 2$ $u = a/3$ $v = b - 1$ RECURSIVE-ALGORITHM2(u, v)

3. (30 pts) In solving the following recurrence equations, first try to use the master theorem. If it does not apply, explain why this is the case.

   a) Solve the following recurrence equation.

$$T(n) = \begin{cases} 1 & \text{if } n \le 1 \\ T(2n/3) + T(n/4) + \Theta(n) & \text{otherwise.} \end{cases}$$

   b) Solve the following recurrence equation. one

$$T(n) = \begin{cases} 1 & \text{if } n \le 1 \\ 4T(n/2) + n^2/\log n & \text{otherwise.} \end{cases}$$

   c) For the following equation provide an exact (*not asymptotic*) closed form solution.

$$T(n) = \begin{cases} 1 & \text{if } n \le 1 \\ 5T(n/2) & \text{otherwise.} \end{cases}$$

4. (16 pts) Consider the following variation of Merge Sort: rather than dividing the array into two equal sized parts, recursively sorting each of them, and then merging them together, we divide the array into $\sqrt{n}$ equal sized parts instead. Once we recursively sort each one of these $\sqrt{n}$ parts of size $\sqrt{n}$ each, we need $\Theta(n \lg n)$ time in order to merge them together, leading to the following recurrence equation:

$$T(n) = \begin{cases} 1 & \text{if } n \le 1 \\ \sqrt{n}T(\sqrt{n}) + n \lg n & \text{otherwise.} \end{cases}$$

   Solve this recurrence equation, providing tight upper and lower bounds

   a) Using the *recursion tree* method.
   b) Using the *master theorem* after applying an appropriate change of variables.