CSCI 3104 Fall 2022
Instructors: Prof. Grochow and Nagesh

Problem Set 7

Due Date ..... October 24, 2022
Name ..... **Your Name**
Student ID ..... **Your Student ID**
Collaborators ..... **List Your Collaborators Here**

Contents

Instructions

- The solutions **must be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Useful links and references on LaTeXcan be found here on Canvas.
- You should submit your work through the **class Canvas page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You are welcome and encouraged to collaborate with your classmates, as well as consult outside resources. You must **cite your sources in this document. Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.

- You **must** virtually sign the Honor Code (see Section Honor Code). Failure to do so will result in your assignment not being graded.

Honor Code (Make Sure to Virtually Sign the Honor Pledge)

Problem HC. On my honor, my submission reflects the following:

- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

In the specified region below, clearly indicate that you have upheld the Honor Code. Then type your name.

Honor Pledge. □

19 Standard 19 - Solving Recurrences: Tree Method

Problem 19. Consider the recurrence $T(n)$ below. Using the **tree method**, determine a suitable function $f(n)$ such that $T(n) \in \Theta(f(n))$. Clearly show all steps. Note the following:

- You may assume, without loss of generality, that n is a power of 3 (i.e., $n = 3^k$ for some integer $k \geq 0$).
- You may hand-draw your tree and embed it, provided it is legible and we do not have to rotate our screens to read it. However, **all your calculations must be typed**.

$$T(n) = \begin{cases} 1, & n < 3 \\ 9T(n/3) + n^2, & \text{otherwise.} \end{cases}$$

Answer. **Consider a tree** whose root node is $T(n)$ and each node of the tree branches into 9 children until the leaf level is reached.

Find the depth of the tree. We reach our base case as we reach the leaf nodes which which are found at the level d of the tree, satisfying $n/3^d < 3$. So we have $d \geq \log_3 n - 1$ (ignore rounding errors).

Compute the total work. At layer i, there are 9^i vertices. Each vertex takes time $\left(n/3^i\right)^2 = n^2/3^{2i}$. Combining these we obtain the work done over each level other than the leaf level:

$$T^{\text{all but leaf}}(n) = \sum_{i=0}^{\log_3 n - 1} 9^i \cdot \frac{n^2}{9^i} = n^2 \log_3(n) \tag{1}$$

□

Next, we find the work done at the leaf level

$$T_{\text{leaf}}(n) = 9^{\log_3(n)-1} = \frac{1}{9} \cdot n^{\log_3 9} = \frac{n^2}{9} \tag{2}$$

Thus, the total cost is

$$\begin{aligned} T(n) &= T^{\text{all but leaf}}(n) + T_{\text{leaf}}(n) \\ T(n) &= n^2 \log_3(n) + \frac{n^2}{9} \\ T(n) &= \Theta\left(n^2 \log_3(n) + \frac{n^2}{9}\right) \\ T(n) &= \Theta(n^2 \log(n)) \end{aligned} \tag{3}$$

since $\Theta(n^2 \log_3(n))$ dominates $\Theta(n^2)$ and also $\Theta(\log_3 n) = \Theta(\log_2 n)$.

20 Standard 20 - QuickSort

20(a). Part (a)

Problem 20. (a) Write down a recurrence relation that models the **best case** running time of Quicksort, i.e. the case where PARTITION selects the **median** element at each iteration. What is the best-case running time of QuickSort? Be sure to write **both the recurrence relation and the runtime.**

Answer.

$$T(n) = \begin{cases} \Theta(1), & n \leq 1, \\ T(n) = 2T(n/2) + \Theta(n) & , n > 1. \end{cases}$$

$T(n) = 2T(n/2) + \Theta(n)$ follows from $T(n) = T(n/2) + T(n/2) + \Theta(n)$.

The best-case running time of QuickSort is $T(n) = \Theta(n \log n)$.

RECALL HOW WE SOLVED THE RECURRENCE RELATION FOR THE MERGESORT, WITH RUNNING TIME: $T(n) = 2T(n/2) + \Theta(n) \Rightarrow T(n) = \Theta(n \log n)$, BOTH VIA UNROLLING AND THE TREE METHOD. SAME METHODS CAN BE APPLIED HERE. NOTE THAT WE GET THE SAME RUNTIME COMPLEXITY AS MERGESORT. □

20(b). Part (b)

(b) Write down a recurrence relation that models the **worst case** running time of Quicksort, i.e. the case where PARTITION selects the **last** element at each iteration. What is the worst-case running time of QuickSort? Be sure to write **both the recurrence relation and the runtime.**

Answer.

$$T(n) = \begin{cases} \Theta(1), & n \leq 1, \\ T(n) = T(n-1) + \Theta(n), & n > 1. \end{cases}$$

$T(n) = T(n-1) + \Theta(n)$ follows from $T(n) = T(n-1) + T(1) + \Theta(n)$.

The worst-case running time of QuickSort is $\Theta\left(n^2\right)$.

TO SEE THIS, USE TAIL RECURSION AND UNROLL.

$$\begin{aligned} T(n) &= T(n-1) + \Theta(n) \\ &= \sum_{k=1}^{n} \Theta(k) \\ &= \Theta\left(\sum_{k=1}^{n} k\right) \\ &= \Theta\left(n^2\right) \end{aligned}$$

□

20(c). Part (c) (Also credit towards S17 or S19)

(c) Suppose that we modify PARTITION so that it chooses the median element as the pivot in calls that occur in nodes of the recursion tree of a call to QUICKSORT whose depth in the recursion tree is divisible by 3, and it chooses the maximum element as the pivot in calls that occur in nodes **all** other depth of this recursion tree.

Assume that the running time of this modified PARTITION is still $\Theta(n)$ on any subarray of length n. You may assume that the root of a recursion tree starts at level 0 (which is divisible by 3), its children are at level 1, etc. For example, the modified PARTITION chooses the median element at the root of the recursion tree, in the next two layers of the recursion tree it chooses the max, and in level 3 of the recursion tree it chooses the median again, and so on.

Your job is to write down a recurrence relation for the running time of this version of QUICKSORT given an array n distinct elements and solve it asymptotically, i.e. give your answer as $\Theta(f(n))$ for some function $f(n)$. Show your work.

(If you solve your recurrence using unrolling, you can get credit towards S17. If you solve your recurrence using the tree method you can get credit towards S19. In either case, this problem also counts towards credit for S20.)

Answer. Note that there are three different kinds of layers: (1) layers in which the algorithm chooses the best possible pivot (median), and (2) *subsequent* two layers in which the algorithm chooses the worst possible pivot (maximum element). We define three functions functions $T_1(n)$, $T_2(n)$, $T_3(n)$, where $T_i(n)$ denotes the running time of the algorithm that starts in a layer of type i (for $i = 1, 2, 3$). We obtain a recurrence relation for each T_i that involves the next T_i:

$$T_1(n) = 2T_2(n/2) + \Theta(n) \tag{4}$$
$$T_2(n) = T_3(n-1) + \Theta(n) \tag{5}$$
$$T_3(n) = T_1(n-1) + \Theta(n) \tag{6}$$

We use substitution to unroll – first we write

$$T_1(n) = 2T_2(n/2) + cn$$

Then inserting Eqn. 5 (write $T_2(n)$ in terms of $T_3(n)$) and Eqn. 6 (write $T_3(n)$ in terms of $T_1(n)$) successively gives

$$T_1(n) = 2T_1(n/2 - 1) + 3cn$$
$$T_1(n) \leq 2T_1(n/2) + \Theta(n)$$

We can solve this recurrence relation by using the tree method or unrolling.

Tree method. Consider a tree whose root note is $T(n)$ and each node of the tree branches into 2 children until the leaf level is reached. We hit the leaf level when $n/2^k \leq 1$. Solving for k, we obtain $k = \lceil \log_2(n) \rceil$. At layer i of this tree there are 2^i nodes. Each node at level i takes time $\frac{n}{2^i} \cdot cn$. So our total work at level i is cn. Summing over all levels gives $T(n) = cn \cdot \lceil \log_2(n) \rceil$. So $T(n) \in \Theta(n \log(n))$.

Unrolling.

$$\begin{aligned} T(n) &= 2T(n/2) + cn \\ &= 2(2T(n/4) + n/2) + cn \\ &= 4T(n/4) + 2cn \\ &= 8T(n/8) + 3cn = \cdots \end{aligned} \tag{7}$$

□

$T(n)$ satisfies the recurrence $T(n) = 2^k T\left(n/2^k\right) + kn$ for any positive integer k (can be verified by induction). The recurrence becomes trivial when $n/2^k = 1$. This is when $k = \log_2 n$, so we substitute

$$T(n) = nT(1) + n\log_2 n = n\log_2 n + n.$$

Thus $T(n) = \Theta(n \log n)$.