CSCI 3104 Fall 2021
Instructors: Profs. Grochow and Waggoner

Problem Set 7

Due Date .... October / 28th / 2021
Name .... **Michael Ghattas**
Student ID **109200649**
Collaborators .... **Me, myself & I**

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.

I agree to the above, Michael Ghattas. $\square$

1 Standard 19: Tree Method.

Problem 1. 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**.

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

Answer:

Given the provided information: $n = 3^k \rightarrow n^2 = (3^k)^2 = 3^{2k} = 9^k$

We identify the non-recursive work at each level: $9^i \cdot \frac{n}{3} = \frac{n9^i}{3} = n \cdot 3^i = n \cdot (\frac{3}{1})^i$

Finding the value of k when he reach the base case:

$\frac{n}{3^k} < 3 \rightarrow n < 3^{k+1} \rightarrow \log n < \log 3^{k+1} \rightarrow \log n < (k+1)\log 3 \rightarrow \frac{\log n}{\log 3} < (k+1) \rightarrow \log_3 < k+1 \rightarrow \log_3 n - 1 < k$

$k = \lceil \log_3(n) - 1 \rceil$

Knowing we are dealing with a geometric series, we proceed as follows:

$$\begin{aligned}
T(n) &= \sum_{i=0}^{\lceil \log_3(n)-1 \rceil} n \cdot (\frac{3}{1})^i \\
&= n \cdot \sum_{i=0}^{\lceil \log_3(n)-1 \rceil} (\frac{3}{1})^i \\
&= n \cdot \sum_{i=0}^{\lceil \log_3(n)-1 \rceil} [\frac{1 - 3^{[\lceil \log_3(n)-1 \rceil]+1}}{1-3}] \\
&= \frac{n}{2} \cdot [3^{\lceil \log_3(n) \rceil}] \\
&= \frac{n}{2} \cdot [n^{\left\lceil \frac{1}{\log_3(3)} \right\rceil}] \\
&= \frac{n}{2} \cdot [n^{\lceil 1 \rceil}] \\
&= \frac{n}{2} \cdot n \\
&= \frac{n^2}{2}
\end{aligned}$$

Thus: $T(n) \in \Theta(n^2)$. □

2 Standard 20: Quicksort.

2.1 Part (a)

Problem 2. (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.

Answer:

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

□

2.2 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.

Answer.

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

□

2.3 Part (c)

(c) Suppose that we modify PARTITION(A, s, e) so that it chooses the median element of $A[s..e]$ in calls that occur in nodes of even depth of the recursion tree of a call QUICKSORT$(A[1, \dots, n], 1, n)$, and it chooses the maximum element of $A[s..e]$ in calls that occur in nodes of odd 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 an even number), its children are at level 1, etc.

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.

Answer:

The modified version of *Quicksort* mentioned in the problem alternates between best-case & worst-case runtime complexity, both of which are defined in parts (a) and (b) above. Thus:

$$T(n) = \begin{cases} 2T(\frac{n}{2}) + \Theta(n) & : When \to even \\ T(n-1) + \Theta(n) & : When \to odd \end{cases}$$

We now consider the combined runtime complexity by considering the best-case & worst-case runtime.

When considering the best-case scenario, we can define it as:

- $f(n) = 2g(\frac{n}{2}) + \Theta(n)$
- $g(n) = 2f(n-1) + \Theta(n)$

Thus we can note that $T(n) = f(n) = g(n)$. Now we substitute the values from above and start combining the values into $f(n)$:

- $f(n) = 2f(\frac{n}{2-1}) + \frac{3}{2}\Theta(n)$
- $g(n) = 2g(\frac{n-1}{2}) + 2\Theta(n) - 1$
- $f(n) = g(n)$
- $f(n) = 2f(\frac{n}{2-1}) + 2\Theta(n) = 2f(\frac{n}{2}) + 2\Theta(n)$

Note that the runtime complexity is approximately double the same the best-case scenario, therefore the combined runtime complexity is:

$T(n) \in \Theta(n \log n)$. $\square$