CSCI 3104 Spring 2022
Instructors: Profs. Chen and Layer

Quiz 17 - Balanced versus Unbalanced Partitioning

Due Date ..... March 18
Name ..... **Julia Troni**
Student ID **109280095**

Contents

1 Instructions

- The solutions **should be typed**, using proper mathematical notation. We cannot accept hand-written solutions. Here's a short intro to LaTeX.
- 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 **may not collaborate with other students**. **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, Discord, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.

2 Standard 17 - Balanced versus Unbalanced Partitioning

Problem 1. Suppose that we modify the Merge-Sort algorithm so that the algorithm chooses a $(2, n-2)$ split for two levels of recursion and then a $(n/2,\ n/2)$ split at the next level, and this repeats so a $(2, n-2)$ split is chosen for the next two levels, then a $(n/2,\ n/2)$ split for one level, etc.

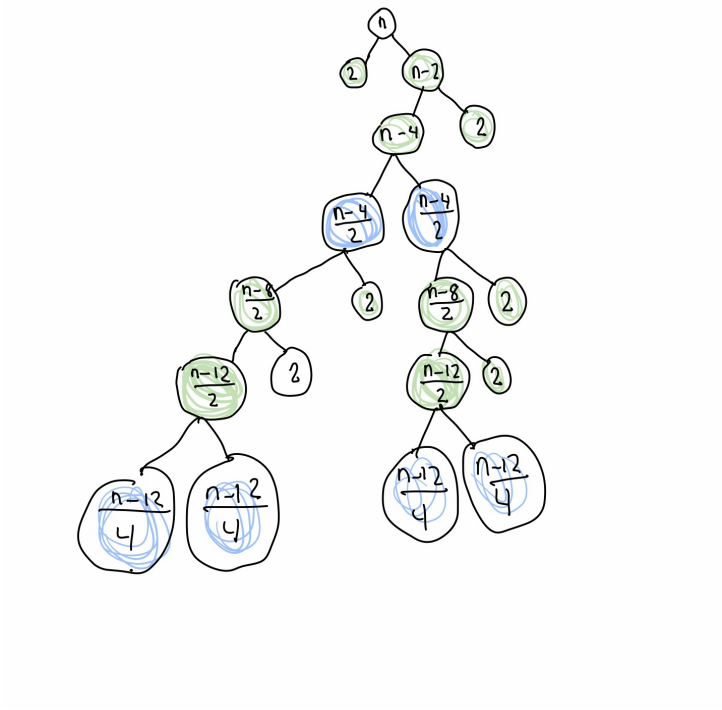
Your job is to write down a recurrence relation for this modified Merge-Sort algorithm and give its asymptotic solution, i.e., give your answer as $\Theta(f(n))$ for some function f. Show your work.

Answer. Answer. The recurrences are

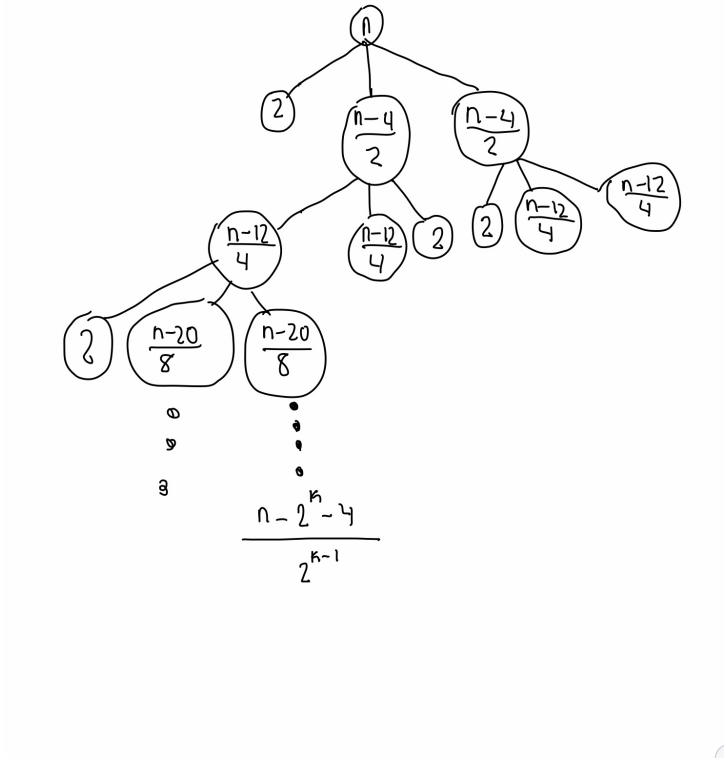
$$T_1(n) = \begin{cases} \Theta(1) & : n \leq 2, \\ T_2(n-2) + 2 + \Theta(n) & : n > 2. \end{cases}$$

$$T_2(n) = \begin{cases} \Theta(1) & : n \leq 2, \\ 2T_1(\frac{n}{2}) + \Theta(n) & : n > 2. \end{cases}$$

The tree is below:



I will now examine every other level of the tree to identify the base case



Observe that at level i of the tree the amount of nonrecursive work is $2 \cdot (n - 2^i - 4) = 2(n-4) - 2 \cdot 2^i$

Let us now determine the number of levels of the tree. Let k denote the number of levels of the tree. We will solve for k. The leaf nodes correspond to the base cases. We hit base cases when $\frac{n-2^k-4}{2^{k-1}} \leq 2$
Solving for k we get $log_2(n-4) - 1 \leq k$
So the number of levels of the tree is $k = \lceil log_2(n-4) - 1 \rceil$

So the total work is

$$\begin{aligned} T(n) &\leq \sum_{i=0}^{\lceil log_2(n-4)-1 \rceil} (2(n-4) - 2 \cdot 2^i) \\ &\leq 2(n-4) \sum_{i=0}^{\lceil log_2(n-4)-1 \rceil} (1) - 2 \cdot \sum_{i=0}^{\lceil log_2(n-4)-1 \rceil} (2^i) \\ &\leq 2(n-4) \cdot (\lceil log_2(n-4) - 1 \rceil) - 2 \cdot \frac{(n-4)}{2} \\ &\leq 2(n-4) \cdot (\lceil log_2(n-4) - 1 \rceil) - (n-4) \end{aligned}$$

Thus, we have that $T(n) \in O(nlogn)$. Thus, this merge sort algorithm does not change asymptotic the running time of merge sort, as traditional merge sort is $T(n) \in \Theta(nlogn)$ □

□