CSCI 3104 Spring 2023
Instructors: Chandra Kanth Nagesh and Prof. Ryan Layer

Quiz 19 Standard 19 – (Divide & Conquer) MergeSort

Due Date .... TODO
Name .... **Blake Raphael**
Student ID **109752312**
Quiz Code (enter in Canvas to get access to the LaTeX template) .... **IKJOP**

Contents

Instructions 1

Honor Code (Make Sure to Virtually Sign) 2

19 Standard 19 – (Divide & Conquer) MergeSort (4 points) 3

Instructions

- You may either type your work using this template, or you may handwrite your work and embed it as an image in this template. **If you choose to handwrite your work, the image must be legible, and oriented so that we do not have to rotate our screens to grade your work.** We have included some helpful LaTeX commands for including and rotating images commented out near the end of the LaTeX template.
- You should submit your work through the **class Gradescope 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.
- You **must** virtually sign the Honor Code (see Section). Failure to do so will result in your assignment not being graded.

Honor Code (Make Sure to Virtually Sign)

Problem HC.
- 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.

Agreed (I agree to the above, Blake Raphael). □

19 Standard 19 – (Divide & Conquer) MergeSort (4 points)

Problem 19. Consider a modified version of MergeSort, which divides the array into 6 (as equally-sized as possible) partitions (**Recall:** Standard MergeSort partitions the array into 2-equal sized sets). This variation of MergeSort is called the 'k-way' MergeSort, where $k = 6$. Instead of dividing the input array into two subarrays, '6-way' MergeSort divides the array into **six** subarrays, sorts them recursively, and then call a '6-way' Merge which combines six sorted arrays into one sorted array. (4 points)

Do the following **three** parts of the question.

1. **Write down** the number of comparisons that occur in the '6-way' Merge function, in the worst case. Assume we have six 'M-element' arrays. (1 point)
   (**Note:** In the worst case, a '2-way' Merge function performs $2M - 1$ comparisons, where we assume to have two 'M-element' arrays. **Donot** give a Big-O notation.)

   Answer. The 6-way Merge function would perform $6M - 1$ comparisons at the worst case. □

2. Now, with the the number of comparisons from part (1) **write down** the recurrence relation for the runtime of this modified version of '6-way' MergeSort. **Justify your recurrence relation in 1–2 sentences.** (1 point)

   Answer.

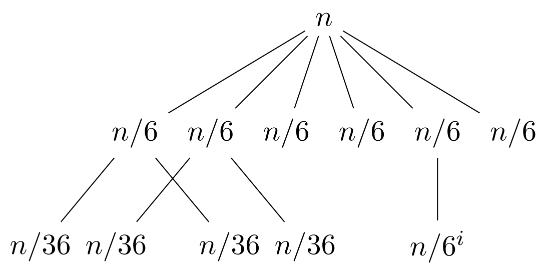
$$T(n) = \begin{cases} \Theta(1) & : n \leq 1, \\ 6T(n/6) + \Theta(n) & : n > 1. \end{cases}$$

   This is the recurrence relation because with each recursion, we break the array down 6 more times, which is $6 * n/6$.

   □

3. Solve your recurrence relation from part (1), by any method you choose. **Show your work.** Find a function $f(n)$ such that the runtime $T(n)$ is $T(n) = O(f(n))$. (2 points)
   (**Further:** You need not show this work, but would a '6-way' MergeSort sort elements faster than '2-way' MergeSort.)

   Answer. Here we will use the tree method to solve the recurrence relation.


n
$n/6$ $n/6$ $n/6$ $n/6$ $n/6$ $n/6$
$n/36$ $n/36$ $n/36$ $n/36$ $n/6^i$

So we see that the non-recursive work at each level is $(n/6^i) * 6^i$.

We also hit our base case at $n/6^k \leq 1$ or $n \leq 6^k$. Solving for k, we get $k \leq \log_6(n)$.

So the total work performed by our 6-way mergesort is $\displaystyle\sum_{i=0}^{\log_6(n)} n = n \cdot \log_6(n)$.

So we see that $T(n) \in \Theta(n \log(n))$.

A 6-way mergesort would not solve any faster than a 2-way mergesort. They have the same Θ runtimes. $\square$