

CS375 Weekly Assignment 4

Due Mar 10th 10PM ET

A general note for CS375: When writing up your homework, please write neatly and explain your answers clearly, giving all details needed to make your answers easy to understand. Graders may not award full credit to incomplete or illegible solutions.

You are encouraged to discuss the problems with your classmates, **but your submission should be your own**. When writing your solutions up, you should not be looking at any other student's submission.

Clear communication **is the point**, on every assignment. In general in CS375, unless explicitly specified otherwise, answers should be accompanied by explanations. Answers without explanations may not receive full credit. Please feel free to ask me any questions about explanations that might come up!

Necessary acknowledgements: many of these questions are reproductions or adaptations of questions of Eric Aaron's devising. Others are from our course texts as listed on the syllabus.

1. A pair of numbers in a list are *inverted* if one is bigger than the other and appears before it. For example, the list [3, 8, 2, 5, 1, 4, 7, 6] has the following 12 inversions: (3, 2), (3, 1), (8, 2), (8, 5), (8, 1), (8, 4), (8, 7), (8, 6), (2, 1), (5, 4), (5, 1), (7, 6). Design and analyze a divide and conquer algorithm for counting the number of inversions and show that its runtime is $\Theta(n \log n)$ (note that this is both a lower and upper bound). (**Hint:** Start with the pseudocode for MERGESORT and adjust it for this problem)
2. Recall the PARTITION algorithm from class:

```
def Partition(A[1, ..., n], r) -> int:
    """ Partitions the array A in such a way that all items smaller than A[r]
    appear before all other items. Note that this may result in A[r] being moved
    to a new location: the new location is returned. """

    Swap A[r] and A[n]

    bigs = 1 # This will mark the start of the indices which are at least as big
              # as the pivot (A[n])

    for i in range(1, n):
        if A[i] < A[n]:
            Swap A[i] and A[bigs]
            bigs += 1

    swap A[bigs] and A[n]

    return bigs
```

Using induction, show that this algorithm is correct. Namely, you're showing that It's a good exercise to come up with it yourself, so I'll sort of hide it, but the exact statement you should be trying to show is in the comments of the tex file if you'd like a hint.

3. For the following functions of T , pick 3 (I'll ask you to pick a few more next week) and determine their asymptotic complexity **by means of a recursion tree**. You can assume that $T(0) = T(1) = 1$ and that all relevant mathematical operations drop remainders (so $\frac{3}{2} = \sqrt{2} = 1$, etc). Note that I have purposely shuffled the list so that the order of these is not in increasing difficulty.

(a) $T(n) = 2T\left(\frac{n}{2}\right) + \frac{n}{\lg n}$

(b) $T(n) = 8T\left(\frac{n}{2}\right) + n^3\sqrt{n}$

(c) $T(n) = 4T\left(\frac{n}{2}\right) + n^2 \lg n$

(d) $T(n) = 5T\left(\frac{n}{3}\right) + \frac{n}{\lg n}$

(e) $T(n) = \sqrt{n}T(\sqrt{n}) + n$

(f) $T(n) = 2T\left(\frac{n}{2}\right) + n \lg n$

(g) $T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + T\left(\frac{n}{8}\right) + n$

(h) $T(n) = \sqrt{2n}T(\sqrt{2n}) + \sqrt{n}$

(i) $T(n) = T(n-2) + \frac{1}{\lg n}$

(j) $T(n) = T\left(\frac{8n}{11}\right) + n$

(k) $T(n) = 3T\left(\frac{n}{3}\right) + \frac{n}{\lg n}$

(l) $T(n) = T(n-1) + 1$

(m) $T(n) = 7T\left(\frac{n}{2}\right) + n^2 \lg n$

(n) $T(n) = T(n-2) + n^2$

(o) $T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{n}$

(p) $T(n) = 16T\left(\frac{n}{4}\right) + n^2$

(q) $T(n) = 2T\left(\frac{n}{2}\right) + n^3$

(r) $T(n) = 8T\left(\frac{n}{3}\right) + n^2$

(s) $T(n) = T(n-1) + n$

(t) $T(n) = T(n-1) + \lg n$

(u) $T(n) = 2T(n-1) + 1$

(v) $T(n) = T(n-1) + \frac{1}{n}$