

Problem Set 1 for CS 170

Note

When asked for an algorithm you must give (1) a brief informal description of the algorithm, (2) a precise description using pseudo-code, (3) an argument for termination and correctness of the algorithm, and (4) an analysis of the running time of the algorithm. Be clear about what the input to the algorithm is, how you measure the size of the input, and what constitutes a “step” in your running-time analysis.

Problem 1. [Asymptotic notation] (6 points)

- (a) Give two functions f_1 and f_2 so that the following equation is true for f_1 , and false for f_2 :

$$\sum_{i=1}^n f(i) = \Theta(f(n))$$

Prove your answers.

- (b) Prove that

$$\log(n!) = \Theta(n \log n)$$

without appealing to Stirling’s approximation of the factorial function.

Problem 2. [Recurrence equations] (10 points)

Give asymptotically tight solutions for $T(n)$.

- (a) Use a recursion tree (CLRS Exercise 4.2-4) to guess a solution, and prove your guess by induction:

$$T(n) = T(n - d) + T(d) + cn$$

for constants $d \geq 1$ and $c > 0$. Assume that $T(n) = \Theta(1)$ for $n \leq d$.

- (b) Use a recursion tree (CLRS Exercise 4.2-5) to guess a solution, and prove your guess by induction:

$$T(n) = T(\alpha n) + T((1 - \alpha)n) + cn$$

for constants $0 < \alpha < 1$ and $c > 0$.

- (c) Use the master theorem:

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

for the three cases $a = 7, 8, 9$ and a constant $c > 0$.

- (d) Use a change of variable and the master theorem:

$$T(n) = 3T(\sqrt{n}) + c$$

for a constant $c > 0$.

Problem 3. [Lower bounds] (8 points)

- (a) You are given $n > 1$ coins. One of the coins is lighter than the others, but otherwise indistinguishable. You have a scale and can put, in a weighing, one coin on each side of the scale. How many weighings do you need to find the fake coin? Give an upper-bound and a lower-bound argument.
- (b) Same as part (a), only that now you can put any number of coins on each side of the scale in a single weighing.
- (c) Same as part (a), only that now you can put up to k coins on each side of the scale in one weighing, for some constant $k \geq 1$.

Problem 4. [Algorithm design] (6 points)

You are given an array $A[1..n]$ of pairwise distinct integers. Give an algorithm that finds the first and second largest elements in A using $n + \Theta(\log n)$ many comparisons between array elements.

Bonus question: With how many comparisons can you find the first, second, and third largest elements in A ?