Computer Science 311
Recitation 2

Weisi Fan

Alvin Chon

02-08-2020

1 Review Notes

Please review and make sure you know the definitions of the three asymptotic notations (Big O, Big Omega, Big Theta) as well as using them in a rigorous proof (as done in the book and in lecture).

Reminder that a recurrence tree is NOT a proof. You can use it to get a recurrence relation which you can then use in your proof.

And of course, make sure you're familiar with the Master Theorem and can use it.

2 Asymptotic Notation

In each of the following situations, indicate whether $f(n) = O(g(n))$,or $f(n) = \Omega(g(n))$, or both(in which case $f(n) = \Theta(g(n))$).

2.1 Big O, Big Omega, Big Theta

$$
\begin{array}{ll}
f(n) = n^2 & g(n) = (\log n)^3 \\
f(n) = n^2 & g(n) = (\log n)^{\log n} \\
f(n) = n^3 & g(n) = 10^{\log_3 n} \\
f(n) = \sqrt[3]{\log n} & g(n) = \sqrt[3]{2}^{\log n} \\
f(n) = (\log n)^{\log n} & g(n) = n^{\log (\log n)} \\
f(n) = n! & g(n) = (n+1)! \\
f(n) = (\log n)^3 & g(n) = (\log n)! \\
f(n) = 2^{2^n} & g(n) = 2^{2^{n+1}} \\
f(n) = 3^n & g(n) = n^{\log (\log n)} \\
f(n) = \log (n!) & g(n) = n \log n \\
f(n) = \sum_{i=1}^{n} i^k & g(n) = n^k
\end{array}
$$

2.2 Proofs

As you can already see in the homework, you will need to use the definitions of the notations in your proofs.

O-notation:
$f(n)$ is $O(g(n))$ if and only if there exist positive constants c and n_0 such that

$$f(n) \leq c * g(n) \text{ for all } n > n_0$$

Prove $n = O(n)$
Prove $n = \Omega(n)$
Prove $n = \Theta(n)$

Disprove $n^2 = O(n)$
Disprove $n = \Omega(n^2)$
Disprove $n = \Theta(1)$

3 Divide and Conquer

Divide: Break the problem into a number of subproblems that are smaller instances of the same problem.
Conquer: Solve subproblems recursively. If a subproblem is sufficiently small, solve it directly, without recursion, in $\Theta(1)$ time.
Combine: Merge the solutions to the subproblems into the solution for the original problem.

Examples: Merge Sort and Maximum-Subarray Problem(Check our slides)

`https://visualgo.net/en/sorting`

Question for practice:

1. Let S be a set of two-dimensional points. Assume that all x-coordinates are distinct and all y-coordinates are distinct. A point $\langle x, y \rangle \in S$ is *acceptable* if there exists a point $\langle p, q \rangle$ in S such that $x < p$ and $y < q$. Give a divide and conquer algorithm that gets a set of points as input and outputs all acceptable points.(Summer 2020 HW)

2. Give an $O(\log m + \log n)$-time algorithm that takes two sorted lists of sizes m and n, respectively, as input and returns the ith smallest element in the union of the two lists.(Spring 2020 HW)