COMS W3137
Solutions for Theory Homework 1

February 21, 2023

Problem 1

We want to show $4n^2 + 2n - 1 = \Theta(n^2)$. We must find positive constants c_1, c_2, and n_0, such that $c_1 n^2 \leq 4n^2 + 2n - 1 \leq c_2 n^2$ for all $n \geq n_0$. Dividing by n^2 yields

$$c_1 \leq 4 + \frac{2}{n} - \frac{1}{n^2} \leq c_2.$$

$\frac{2}{n} - \frac{1}{n^2}$ approaches 0 as n increases, so $\lim\limits_{n \to \infty} 4 + \frac{2}{n} - \frac{1}{n^2} = 4$. So we can set (for example) $c_2 = 5$ to satisfy the right side of the inequality. For the left side, we can set (again, for example) $c_1 = 1$ and $n_0 = 1$. To verify:

$$4 + \frac{2}{n_0} - \frac{1}{n_0^2} = 5.$$

Other proofs (and other constants) are possible.

Problem 2

Given a polynomial $p_d(n) = \sum_{i=0}^{d} a_i n^i$ and some integer $k \geq d$, We need to show that

$$\sum_{i=0}^{d} a_i n^i \leq cn^k \textit{ for all } n \geq n_0.$$

Because $k \geq d$, we know $n^k \geq n^i$ for each term n^i, therefore

$$\sum_{i=0}^{d} a_i n^i \leq \sum_{i=0}^{d} a_i n^k$$

factor out n^k:

$$\sum_{i=0}^{d} a_i n^i \leq n^k \sum_{i=0}^{d} a_i.$$

So we can set $c = \sum_{i=0}^{d} a_i$ and $n_0 = 1$ and satisfy the definition for big-O.
Other proofs are possible.

Problem 3

$2/N < 128 < \log N < \sqrt{N} < 3N < N \log N < N^2 < 7N^3 < 2^N = 2^{N+1} < 4^n < N!$

Note that $lim_{n\to\infty} \frac{2^{n+1}}{2^n} = 2$ and therefore $2^n = \Theta(2^{n+1})$, but $lim_{n\to\infty} \frac{4^n}{2^n} = 2^n$, so $2^n = o(4^n)$.

Problem 4

a)

$$D = 2^{2^{(N-1)}}$$

b) Solve the equation from a) for N.

$$D = 2^{2^{(N-1)}}$$
$$log_2 D = 2^{(N-1)}$$
$$log_2 \log_2 D = N - 1$$
$$N = \log_2 \log_2 D + 1$$

Therefore $N = \Theta(\log \log D)$.

Problem 5

a) The inner loop runs a constant number of times, so only the outer loop depends on n. The runtime is $O(n)$.

b) In the first iteration of the outer loop, the inner loop will not run. In the second iteration of the outer loop, the inner loop performs $O(n^2)$ steps, but now $k = n^2$, so the outer loop will not enter into another iteration. The the total runtime is $O(n^2)$.

c) The function will call itself recursively $log_c n$ times before the base case is reached. The runtime is $O(\log n)$