College of Computer & Information Science
Northeastern University
Fall 2017

CS5800
Algorithms
29 September 2017

Sample Solution to Quiz 1

1. Let $f(n)$ and $g(n)$ be positive non-decreasing functions such that $f(n) = O(g(n))$. Prove or disprove: $f(n) = O(g(n)^{1/3})$.

Answer: False. Take $f(n) = g(n) = n$. Then, $f(n) = O(g(n))$ since $n = O(n)$. However, $n \neq O(n^{1/3})$ since $\lim_{n\to\infty} n/n^{1/3} = \infty$.

2. Let $a = (a_1, a_2, \ldots, a_n)$ be an array of n distinct integers. Given array a and an integer k, describe a deterministic algorithm that determines whether there exist indices i and j such that $a_i + a_j = k$. State the worst-case running time of your algorithm.

For instance, if $a = (2, -1, 3, 5, 8)$ and $k = 2$, your algorithm should return YES since 3 + -1 = 2, while if $a = (2, -1, 3, 5, 8)$ and $k = 6$, your algorithm should return NO.

Your grade for this question will be determined on the basis of the correctness of your algorithm and its efficiency, given by its worst-case running time. Partial credit may be given for non-optimal algorithms provided they are correct and well explained. In your solution, you may use any algorithm we have covered in class, but you may not use data structures (e.g., maps, hashmap, hash tables) that have not yet been discussed in the course.

Answer: A brute-force algorithm takes $\Theta(n^2)$ time.

Here are two $\Theta(n \log n)$ time algorithms.

1. Sort a using Mergesort.
2. For each element x in a:
    - Use binary search to check if $k - x$ is in (sorted) a.

The first step takes $\Theta(n \log n)$ time. Since binary search takes $\Theta(\log n)$ time, the running time of the second step is $\Theta(n \log n)$. So the total time is $\Theta(n \log n)$.

Here is another algorithm:

1. Sort a using Mergesort.
2. Construct array b such that $b_i = k - a_i$.
3. Compare the elements of the arrays similarly to the Merge algorithm to determine whether 2 elements are equal.

Running time = $\Theta(n \log n)$.