College of Computer & Information Science — CS5800
Northeastern University — MS Algorithms
Fall 2017 — 4 October 2017

Problem set 3 (Due Wednesday, October 11, 11:59 pm)

- The assignment is due at the time and date specified. Late assignments will not be accepted.
- We encourage you to attempt and work out all of the problems on your own. You are permitted to study with friends and discuss the problems; however, *you must write up your own solutions, in your own words.*
- If you do collaborate with any of the other students on any problem, please do list all your collaborators in your submission for each problem.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class (or the class staff) is strictly prohibited.
- We require that all homework submissions be neat, organized, and *typeset.* You may use plain text or a word processor like Microsoft Word or LaTeX for your submissions. If you need to draw any diagrams, however, you may draw them with your hand.

1. (6 points) Determining the end of an array

You are given a long array A, in which the first n indices contain arbitrary integers, and the remaining entries (indices n and higher) are all ∞. Give an algorithm that determines n, while accessing at most $O(\log n)$ of the entries of A.

2. (4 + 4 = 8 points) Graph coloring

This exercise is primarily to help you reason about graphs.

Call an undirected graph k-*colorable* if one can assign each vertex a color drawn from $\{1, 2, \ldots, k\}$ such that no two adjacent vertices have the same color. We know that bipartite graphs are 2-colorable.

(a) Prove that if the degree of every vertex of G is at most Δ, then the graph is $(\Delta+1)$-colorable. Show how to contruct, for every integer $\Delta > 0$, a graph with maximum degree Δ that requires $\Delta + 1$ colors.

(b) Prove that if G has n vertices and $O(n)$ edges, then it can be colored with $O(\sqrt{n})$ colors.

3. (8 points) Number of shortest paths in social networks

Chapter 3, Exercise 10, page 110. (*Hint:* Use breadth-first search.)

4. (8 points) A walk through the entire graph

Give an algorithm that takes as input an undirected graph $G = (V, E)$ and returns a path that traverses every edges of G exactly once in each direction. Your algorithm should run in $\Theta(n + m)$ time in the worst-case, where m is the number of edges and n is the number of vertices.