Homework #1: UF and PQ
Due Date: both written and program, by 5pm Friday 1/29

This is written part of hw1, the programming part is an edge percolation experiment described in share/hw1/. Both parts count for 50% of the grade.

Try to solve these problems, and turn in your work on paper. You do not have to write a lot, at most a half-page per problem. If you cannot turn this in during class or during office hours, then slide your work under my office door (W426).

Problem 1. Consider these two versions of the UF data structure (from Section 1.5): weighted quick-union (without path compression), and path compression (without weights). Give an advantage of each of these over the other. (Hint: time versus space.)

Problem 2. Let $f(M, N, p)$ be the edge percolation probability as defined in the programming part of this homework. Approximately plot (by hand or by computer) the two curves $f(5, 5, p)$ and $f(50, 50, p)$, as functions of $p$, for $0 \leq p \leq 1$. (You should see a “critical point” developing around $p = 1/2$.)

Problem 3. Suppose we are reading an input stream of $N$ numbers, and after reading the stream, we want to report the $M$ largest numbers. Suppose $M$ is big, but relatively small compared to $N$: $M \leq N/\lg N$. For both parts, describe how to solve the problem using a binary heap. (One of these is “TopM” in the book, for the other you need the linear-time heap construction method of Section 2.4, part of the discussion of heapsort.)

3(a). Solve the problem in $O(N)$ time and $O(N)$ space.

3(b). Solve the problem in $O(N \lg M)$ time and $O(M)$ space.

Remarks: In both parts, “space” counts words of memory. A word is large enough to store an input number, an index, or a pointer. These two solutions demonstrate a “trade-off” between time and space.

Problem 4. The “height” of a tree is the number of steps on a longest path from some leaf to the root; in particular, a single-node tree has height 0. Consider a tree of height $h$ constructed by the weighted quick-union algorithm (without path compression).

Argue that such a tree has at least $2^h$ nodes. (Try to use an induction on $h$, not on size like the book.)

Honor Policy This assignment is governed by the Emory Honor Code, it should be your own work. The programming part is governed by the department’s Statement of Policy on Computer Assignments.