Problem Set 7 for CS 170

Formatting

Please use the following format for the top of the solution you turn in, with one line per item below (in the order shown below):

<your username on cory.eecs>
<your full name>
`CS170, Spring 2003`
`Homework #7`
`Section` <your section number>
`Partners:` <your list of partners>

(Remember to write your section number, not the name of your TA or the time of your section.) This will make it easier for us to sort and process your homeworks. Thank you!

Note

When asked for an algorithm you must give (1) a brief informal description of the algorithm, (2) a precise description using pseudo-code, (3) an informal argument for termination and correctness of the algorithm, and (4) an analysis of the running time of the algorithm. Be clear about what the input to the algorithm is, how you measure the size of the input, and what constitutes a "step" in your running-time analysis.

Problem 0. [Any questions?] (5 points)

What's the one thing you'd most like to see explained better in lecture or discussion sections? A one-line answer would be appreciated.

(Sometimes we botch the description of some concept, leaving people confused. Sometimes we omit things people would like to hear about. Sometimes the book is very confusing on some point. Here's your chance to tell us what those things were.)

Problem 1. [Union/Find] (20 points)

Consider a variation of the Union/Find data structure, with the Union-by-rank heuristic but without path compression. That is, we implement Makeset() and Union() as usual, but we do not reset the parent pointers in Find().

Show that there is some sequence of n calls to Makeset(), some number (at most n) of calls to Union(), and m calls to Find() that require $\Omega(m \log n)$ work from this suboptimal implementation.

Problem 2. [Most Likely Partition] (25 points)

Let $G = (V, E)$ be an undirected graph and $p : E \to [0, 1]$ be a weight function on the edges. G represents a communication network. For $(u, v) \in E$, $p(u, v)$ is the probability that the link between u and v fails on May 11, 2003. (May 11 is Mother's Day—the busiest day of the year for the public phone network!) Assume that the links fail independently.

We say that a set $S \subseteq E$ of edges *partitions* G if its removal disconnects G (i.e., if $G' = (V, E \setminus S)$ is disconnected). If $S \subseteq E$, let $p(S)$ be the probability that all edges in S fail on Mother's Day.

(a) Let $f(n)$ be the maximum number of partitions for a graph on n vertices. Is there some constant $c > 0$ such that $f(n) = O(n^c)$? Justify your answer informally.

(b) We say that a set $S \subseteq E$ that partitions G is a *most likely partition* for G if $p(S)$ is maximal among all sets that partition G. Give an $O(n^4)$ algorithm that computes a most likely partition for G.

Your algorithm may have a small chance of providing an erroneous answer, as long as the probability of erring is negligibly small.

Problem 3. [How Biased Is the Coin?] (25 points)

(a) You're given a biased coin: it comes up heads with probability p and tails with probability $1 - p$. Describe a randomized algorithm to estimate p, in the sense that your algorithm should output a guess for p that is in the range $[p - 0.01, p + 0.01]$ with probability at least 0.99.

HINT: You could flip the coin many times and then try to estimate p from the number of heads you've seen so far. How many times would you need to flip the coin to achieve the desired error bounds?

HINT: If you're not familiar with the variance of a random variable and Chebyshev's bound, you might want to read up on that. Lectures 22 (Variance) and 23 (Polling and voting) from CS 70 would be a good reference.

(b) Now you're given a communications network, i.e., an undirected graph $G = (V, E)$ and a link failure probability function $p : E \to [0, 1]$, like in Problem 2. We would like to determine the probability q that this network gets disconnected on Mother's Day. Describe a randomized algorithm to estimate q in the sense given above (i.e., you should output a guess so that at least 99% of the time your guess is in the range $[q - 0.01, q + 0.01]$).

WARNING: It turns out that computing q exactly is NP-hard, so don't try to do that. Just try to estimate it.

Problem 4. [A Faster Min-Cut] (25 points)

The randomized algorithm for min-cut that we gave in class ran in $O(n^4)$ time. In this problem, we will develop another min-cut algorithm that is based on the same ideas but has better asymptotic running time.

Consider the following algorithm:

`F`(G):
1: do $|V|/2$ times:
2:     pick $(u, v) \in E$ with probability proportional to $\mathrm{wt}(u, v)$
3:     contract (u, v), updating V and E
4: return `Trial`(G)

`Trial`(G):
1: if $|V| < 16$:
2:     *use the* $O(n^4)$ *algorithm given in class*
3: else:
4:     for $i := 1, 2, 3, 4$:
5:         $C_i :=$ `F`(G)
6:     return the C_i of least weight

(a) Write a recurrence relation for the worst-case running time of `Trial` in terms of n, where $n = |V|$.

(b) Solve your recurrence relation from (a) to get the asymptotic worst-case running time of `Trial`. Use $\Theta()$ notation.

(c) Let C be a min-cut of $G = (V, E)$. Consider the probability that steps 1–3 of `F`(G) never do a contraction on an edge crossing between C and $V \setminus C$. Show that this probability is at least $\frac{n-2}{4n-4}$.

(d) Let $p(n)$ denote the probability that `Trial`(G) finds the min-cut of G with n vertices. Show how to derive this recurrence relation:

$$p(n) \geq 1 - \left(1 - \frac{n-2}{4n-4} \cdot p(n/2)\right)^4 .$$

(e) It turns out that the solution to the recurrence relation in (d) is $p(n) = \Omega(\frac{1}{\lg n})$. (You do *not* need to show this!)

Using this fact, describe an efficient algorithm that returns a min-cut of G with probability at least $1 - \frac{1}{2^{100}}$. What is its running time?