Due on April 2nd, 2018
65 points total

General Directions: If you are asked to provide an algorithm, you should clearly define each step of the procedure, establish its correctness, and then analyze its overall running time. There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice.

All the answers must be typed, preferably using LaTeX. If you are unfamiliar with LaTeX, you are strongly encouraged to learn it. However, answers typed in other text processing software and properly converted to a pdf file will also be accepted. Before submitting the pdf file, please make sure that it can be opened using any standard pdf reader (such as Acrobat Reader) and your entire answer is readable. **Handwritten answers or pdf files that cannot be opened will not be graded and will not receive any credit.**

Finally, please read the detailed collaboration policy given on the course website. You are **not** allowed to discuss homework problems in groups of more than 3 students. **Failure to adhere to these guidelines will be promptly reported to the relevant authority without exception.**

Problem 1 (15 points)
Let $G = (V, E)$ be a unit-capacity graph with n vertices and m edges. In this question, we will revisit the randomized edge contraction algorithm for the global min-cut problem we saw in class. In this setting, the contraction algorithm will do one additional contraction so that the final graph is just a single vertex (instead of being left with two vertices which define a cut).

(a) (5 points) Show that in any run of the edge contraction algorithm, the edges contracted form a spanning tree of G.

(b) (5 points) Using the analysis we saw in class for the contraction algorithm, argue that there are $O(n^2)$ *global min-cuts* in any graph.

(c) (5 points) Observe that part (c) implies nothing about the number of s-t *min-cuts* there can be for two vertices $s, t \in V$. Give an example of a graph where there are $\omega(n^2)$ s-t min-cuts for a particular pair of vertices s and t. Recall that a function $g(n) = \omega(n^2)$ if $g(n)$ is *strictly* greater than n^2 asymptotically , i.e., $g(n) = \Omega(n^2)$ but $g(n) \neq \Theta(n^2)$ (so functions like n^3, n^{10}, 2^n, etc.).

Note that this counterexample needs to be constructed based on a general parameter n since we are attempting to show an asymptotic lower bound. This means you should describe a graph with n vertices, define the edge set, and then argue why there are $\omega(n^2)$ min-cuts.

Problem 2 (25 points)
Consider the following randomized algorithm for computing a spanning tree S for an *unweighted* undirected graph $G = (V, E)$ where $|V| = n$ and $|E| = m$. Denote by E_v the set of edges incident on vertex $v \in V$:

1. Pick an arbitrary root $r \in V$.
2. Starting at r, begin traversing the graph randomly, i.e., when at a given vertex v pick an edge $e = (v, u)$ from E_v uniformly at random and then travel along e to u.
3. Whenever we travel along an edge e and reach a vertex for the first time, add e to S.

We continue this random traversal until we visit every vertex. Let p_e be the probability that edge e is added to S during the random process. The following questions ask you to analyze this algorithm.

(a) (5 points) Prove that the edges in S at the end of the algorithm form a spanning tree.

(b) (5 points) Show that $\sum_{e \in E} p_e = n - 1$.

(c) (15 points) For an edge $e = (u, v)$, define $d_e = \min(\deg(u), \deg(v))$, where $\deg(v) = |E_v|$ is the *degree* of a vertex v. Prove that $\sum_{e \in E}(p_e \cdot d_e) \leq 2m$.

Problem 3 (25 points)
Our friends Alice Algorithmix and Bob Bitfiddler are trying to learn randomized algorithms by playing a game. They start with n cards, all of which are initially placed in a single deck. The game proceeds in rounds. In each round, Alice chooses a deck containing at least 2 cards, secretly marks 1 card in the deck, and asks Bob to partition these cards into two decks of sizes chosen by her. For instance, if there are 10 cards in the deck chosen by her, she can ask Bob to partition the cards into two decks containing 8 and 2 cards respectively. (Bob does not know the marked card when partitioning the cards.) Once Bob has done the partitioning, his task is to record all the cards that were in the deck *not* containing the marked card. The game ends when every deck has exactly 1 card. The time that Bob takes to record the cards in a round is $\Theta(t)$, where t is the number of cards being recorded.

(a) (5 points) Show that if Bob uses a deterministic strategy for partitioning the cards, Alice can ensure that he will take a total time of $\Theta(n^2)$ over all rounds.

(b) (20 points) Instead, suppose Bob partitions the cards randomly. In other words, if he has to partition a deck of t cards into two decks of k and $t-k$ cards, then he shuffles the cards into a uniformly random order and chooses the first k cards in the first deck, and the last $t-k$ cards in the second deck. The random partitioning is done by Bob after Alice has marked a card from the overall deck. What is the expected time that Bob now takes over all rounds? (Give your answer using the $\Theta(.)$ notation.)