CIS 413/513 Final
DUE: 5:00pm on Thursday, June 10th

**Instructions:** You must complete this exam independently, with no outside resources of any kind except for: the Tarjan book, CLRS, the Fibonacci heap article, the “Scapegoat and Splay Trees” notes referenced in a previous homework, the skip lists article, and any notes you took in class. If you find a question confusing or suspect that it is inaccurate, please contact me ASAP by email. **You may not discuss any of these problems with any other person until after Thursday at 5:00pm.**

**Grad students:** Please answer all problems.

**Undergrads:** Your grade will be based on your 6 highest scoring problems. Feel free to answer as many as you like to maximize your chance of a good score.

*All problems are worth the same number of points.*

1. Explain what’s wrong with the Fibonacci heap in the figure below. (NOTE: Gray nodes represent marked nodes, and the min pointer has been omitted for simplicity. Children of each node are intentionally drawn as undirected edges, although they would typically be implemented as a circular linked list with a pointer from the parent.)

![Fibonacci Heap Diagram](image)

2. Explain what’s wrong with the link/cut tree in the figure below. (This figure represents the “actual tree,” not the virtual tree used to implement the actual tree.)

![Link/Cut Tree Diagram](image)
3. Consider the flow problem below, with source $s$, sink $t$, and edge capacities as shown.

(a) Find a maximum flow. Is it unique?
(b) Find a minimum cut. Is it unique?
(c) Explain why Dinic’s algorithm is guaranteed to terminate in one iteration (that is, after finding a single blocking flow).

4. Using the potential function from class, $\Phi = \sum_x r(x) = \sum_x \lceil \lg(\text{size}(x)) \rceil$, prove that the potential of a splay tree with $n$ nodes must always be less than or equal to $\lg(n!)$, where $n! = n \times (n - 1) \times (n - 2) \times \ldots 2 \times 1$. (Hint: Recall that $\lg(ab) = \lg(a) + \lg(b)$.)

5. Describe an algorithm for converting all solid edges into dashed edges in a link/cut tree, using only expose operations. (In other words, do not directly call the link or cut operations.)

6. Describe how to implement the function `add-single-cost(v, c)`, which adds $c$ to the cost of a single vertex $v$ in a link/cut tree and runs in amortized $O(\lg(n))$ time. (Note that this is different from `addcost(v, c)`, which adds the cost $c$ to every node from $v$ to the root.) You may use path operations and other tree operations as subroutines.

7. Let $f$ be a maximum flow on $G$, and let $(u, v)$ be an edge saturated by $f$. Show by counterexample that increasing $\text{cap}(u, v)$ does not necessarily increase the value of a maximum flow on $G$. In other words, show that increasing the capacity of a single saturated edge may not increase the amount of flow that can be pushed through the flow graph.

8. Let $f$ be a maximum flow on $G$, and let $(u, v)$ be an edge not saturated by $f$. Use the max-flow/min-cut theorem to prove that increasing $\text{cap}(u, v)$ never increases the value of a maximum flow on $G$. 