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, your homeworks, homework and test solutions I’ve distributed, 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 4 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. Consider a link/cut tree with \( n \) nodes. Let the number of children of the \( i \)th node be \( c_i \).

   (a) Given an arbitrary link/cut tree, explain how to compute the maximum number of solid edges possible in it using only \( n \) and the numbers of children \( \{c_i\} \).

   (b) Describe an algorithm that will take any link/cut tree and make as many edges solid as possible. (Do not use the link or cut operations in your solution for this problem – they are not necessary.)

2. (a) Describe how to implement \( \text{findnodecost}(v) \), a new operation which returns the cost of a given node in a link/cut tree. (HINT: Use \( \text{findcost} \) as a subroutine, along with whatever other tree operations you need.)

   (b) Assuming that link/cut trees are implemented with splay trees as described in RET 5, what is the worst-case (not amortized) complexity of \( \text{findnodecost} \)?

3. (a) Draw an actual tree corresponding to the virtual link/cut tree shown below.

   (b) Is there more than one actual tree that could correspond to this virtual tree?
4. FlowBot is a helpful robot who can solve one max-flow problem per day, of any size. However, if FlowBot attempts to solve more than one max-flow problem on any given day, it will explode. Explain in detail how you can use FlowBot to help you solve \( n \) distinct max-flow problems in a single day without making it explode, and without using any other max-flow solver.

Please be clear about:

- Any preprocessing you need to do.
- How you obtain solutions to the \( n \) original problems from FlowBot’s max-flow solution.

5. We define a raw flow, \( r \), to be a function over edges satisfying the following two conditions:

- Flow Conservation (raw): for all \( v \neq s, t \),
  \[
  \sum_{w \in V} r(v, w) = \sum_{w \in V} r(w, v)
  \]

- Capacity Constraint (raw): for all \( v, w \in V \),
  \[
  0 \leq r(v, w) \leq \text{cap}(v, w)
  \]

We further define the value of a raw flow, \( |r| \), to be the sum of flows out of the source:
\[
|r| = \sum_{v \in V} r(s, v).
\]
(We assume that the source has no incoming edges.)

For this problem, we use the term “network flow” to refer to our definition of a flow from class, which was somewhat different. This definition can also be found at the start of RET chapter 8.

(a) Let \( f(u, v) = r(u, v) - r(v, u) \). Prove that, for any raw flow \( r \), \( f \) is a valid network flow (as defined in class and the RET book).

(b) Prove that \( |f| = |r| \).