Assignment 21: Cut Nodes
Due in class Monday, 3/11

March 7, 2019
CS: DS&A
Proof School

Let G be a connected graph. Recall that a vertex v is called a **cut node** if removing v and its edges results in a disconnected graph. Below, we describe an algorithm that finds all the cut nodes in linear ($\Theta(|V| + |E|)$) time. Your job is to (1) fill in the blanks in the algorithm, and (2) prove that the algorithm works.

The Algorithm. The basic idea is to do a DFS, but maintain an additional function which we'll call $L(v)$. By the time the algorithm ends, $L(v)$ will be the start time of the earliest-discovered node which is reachable from v via DFS-tree edges and at most one back edge. (Read through that once or twice!)

Here is the actual algorithm. (Note: Given a node a, a_s and a_f refer to the start and end times of a under DFS.)

Run DFS. Define $L(v)$ as follows:

- When a node v is discovered, set $L(v) = v_s$.
- When a back edge from v to w is discovered while processing v's neighbors, change $L(v)$ to ________, if ________ is less than the current value of $L(v)$. (Same thing goes in both blanks.)
- When v finishes, change $L(v)$ to $M =$ ____________________, if M is less than the current value of $L(v)$.
- When v finishes, determine whether v is a cut-node as follows:
  - If v is the start node, it is a cut node iff v has more than one child in the DFS tree.
  - If V is not the start node, it is a cut node iff _________________________.

The Verification. Here is some structure to help you along. In your proof, feel free to use the fact established in class that if there is a back edge from a to b, then a is an ancestor or a descendant of b in the DFS-tree. (From now on, "child", "ancestor", and "descendant" will always mean with respect to the DFS-tree.) The blanks here are symbolic; a 3-cm blank might take a whole paragraph to fill out!

First, suppose that v is the start node. If v has two children a and b, then when we remove v they cannot be connected, because: ______________ On the other hand, if v has one child a, when we remove v, a can be connected to any other node b because ______________.

Now for the main argument. Suppose that v is not the start node.

If v satisfies the condition you wrote above, then it must be a cut node. Indeed, when we remove v, I claim v has a child that cannot cannot be connected to the root R. This is because ______________.

Conversely, suppose that your condition fails. Let a and b be any two nodes in the graph that are not v. I claim that a and b can still be connected when we remove v. Here's why: ______________ (There are multiple cases to consider, based on where a and b lie in the DFS-tree.)