Problem 9: Drop the Bass I

By Henry Tang

The intuitive idea that may come into mind is that this is a maxflow problem. However, trying to use maxflow algorithms presents two issues: maxflow algorithms tend to be fairly complex and slow, and you still need to find the edges that restrict the flow. Instead, we will use a different idea that solely relies on two DFS runs. Let's first consider the simpler problem of finding all vertices (instead of edges) that separate a and b upon being removed. Let's call these special nodes. To do this, first run a DFS from a to b in order to find any path from a to b. Label these nodes $1, \cdots, k$. All special nodes must be on this path, or else they wouldn't be special (a special node must prevent this path from existing if removed)!

Call this path the "central path", and call nodes on it "central nodes". Now, we will begin our second DFS. This DFS is a little different, in that we will process central nodes in order one by one, while keeping track of the highest central node that has been reached thus far in this DFS. For each central node, first check if the highest reached central node is less than or equal to itself. We claim that a node is a special node if and only this condition holds. The intuitive reason behind this is that if there is a higher central reached node when we start processing a node v, then there was a path that arrived at the higher node without passing through v. Thus, there is a path from 1 to n that skips over v.

No matter whether this is a special node or not, we then run a DFS from this node, while making sure not to recursively run DFS from already visited nodes or central nodes. Update the highest reached central node if applicable. Running this for each central node will generate the desired list of special nodes.

However, the problem is not just asking us for special nodes, it is asking for edges that break the graph into two sections, which we will call special edges. To do this, we can use the framework we have already built. Call edges along the central path "central edges". First, note that like with special nodes, only central edges can be special edges.

Now, we present a lemma. A central edge (u, v) (without loss of generality assume u comes before v in the central path) is a special edge if and only if both u and v are special nodes, and there is only one path from u to v (namely through the edge (u, v)). The intuitive reason why this is correct is because if both u and v are special nodes, then disconnecting u will create a connected section of the graph from 1 and u, while disconnecting v will create a connected section of the graph containing v and n. However, there can still be many paths from the first section to the second section, and we want there to only be 1 (namely from u to v directly).

How do we figure out if there is only one path from u to v? We can keep track of the number of times we reached v when it was the highest central node reached during the second DFS. If it was only reached once before we actually process v, then we know there is only one path from u to v. Note it must have been reached at least once because we process u before v, and since there is an edge (u, v), we have reached v at least once before we actually process it.

Time Complexity: $\mathcal{O}(m + n)$, as we only need to do two searches.

Plaudits:

- This was quite a difficult problem. We are impressed that Antonio Molina (grad) was able to solve it in 68 minutes!