Leetcode, $\cdots$

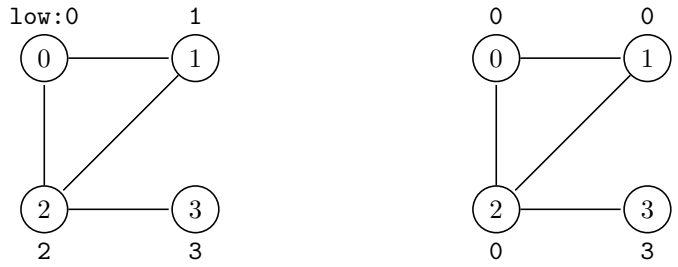
Critical Connections in a Network

id: 1192 tags: graph, dfs

Par	`ids[node]`	keep tracking ids of nodes in dfs ordering
	`low[node]`	smallest id which current node can reach
	`graph[node]`	adjacency list

Alg

1. build graph in form of adjacency list, `graph[node]`
2. tranverse graph dfsly. If neighbor node is not visited, dfs next node, update `low[node]` by `min(low[node], low[neighbor])` by callback.
3. Check if `ids[node] < low[neighbor]` is true, then we find one critical connection.
4. If neighbor node is visited and it is not the node visited right before current node, update `low[node]` by the same as in 2.



You can see that `ids[2] < low[3]`.

Sample

id: tags:
Par `i` par1
`j` par2
`k` par3
Alg

1. todo
2. todo
3. todo