Coding Competitions Farewell Rounds - Round B

Analysis: Railroad Maintenance

Test Set 1

We can represent the railroad network as an undirected graph by creating a node for each station, and then creating an edge for each pair of adjacent stations in each train line, that is, for every $1 \leq i \leq \mathbf{L}$, and for every $1 \leq j \leq \mathbf{K_i} - 1$, add an edge between nodes $\mathbf{S_{i,j}}$ and $\mathbf{S_{i,j+1}}$.

Let's define an undirected graph to be *connected* if, for every pair of vertices x and y, there is a path between them. From that definition, we can say that a train line is *essential* if, by removing it, a *connected* graph stops being *connected*.

Given an undirected graph, to check if it's *connected*, it is enough to check if a single arbitrary vertex can reach all others. To prove it, let's assume that a vertex v can reach all other vertices. Then, because the graph is undirected, all vertices can also reach vertex v. Therefore, for any pair of vertices x and y, we can see that x can reach y because x can reach v and v can reach y.

Now we can check if a train line is *essential* by building the graph without the edges that belong to that train line, and checking if the graph is not *connected*. If we do that for each train line, we will be able to solve Test Set 1.

Let's define E as the number of edges of the entire graph, that is, $E = \mathbf{K_1} + \mathbf{K_2} + \cdots + \mathbf{K_L}$. The time complexity to build the graph is $O(\mathbf{N} + E)$. The time complexity to check if the graph is *connected* is $O(\mathbf{N} + E)$. These two operations have to be performed for every train line, therefore the overall time complexity of the algorithm will be $O(\mathbf{L} \times (\mathbf{N} + E))$.

Test Set 2

The algorithm described for Test Set 1 is too slow for the limits of Test Set 2.

Now let's introduce the concept of *articulation points* (or *cut vertices*), which are vertices that, when removed (along with its adjacent edges), make a *connected* undirected graph not be *connected* anymore.

Let's explore a different way to represent the railroad network. We will first create $\mathbf{L}$ special nodes to represent each train line, and then add edges between each train line's special node and the stations that such train line goes through. In other words, for every $1 \leq i \leq \mathbf{L}$, let's create a special node $\mathbf{P_i}$, and for every $1 \leq j \leq \mathbf{K_i}$, add an edge between nodes $\mathbf{S_{i,j}}$ and $\mathbf{P_i}$.

For this graph representation, the action of shutting down a train line can be seen as removing its special node and all its adjacent edges. Now notice that if a train line's special node is an *articulation point*, that means that such train line is *essential*.

To find all *articulation points* of an undirected graph we can use this modified Depth First Search (DFS) algorithm.

The time complexity to build the graph is $O(\mathbf{N} + E)$. The time complexity to run the modified DFS algorithm is $O(\mathbf{N} + E)$. Therefore, the overall time complexity of the algorithm will be $O(\mathbf{N} + E)$.