Travelling Merchant – Editorial

Keenan Gugeler / Riolku

May 14, 2021

The Problem

In essence, we have a directed graph. Each edge has two associated values, a requirement r to use it (although it does not subtract from our current balance) and a prize p.

For each node, we want to know the minimum money we need in order to keep travelling forever.

Subtask 1

For each node, wildly underestimate and say the answer is 0.

Now for each node v, consider all edges (v, u, r, p). Now for each edge, if we choose to take it, we need to have at least $\max(r, \text{ans}[u] - p)$ units of money.

Our answer for v will be at *least* as large as the minimum across all edges considered.

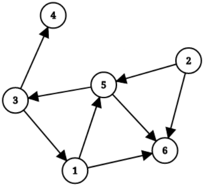
Subtask 1 (cont.)

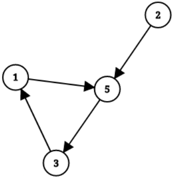
Note that if a node has no viable neighbours to travel to, it should be marked as ∞ to avoid a loop.

We can run this loop as many times as we need, getting closer and closer to the true answer each time. We can prove that we need only run this loop N times.

Subtask 2

Prune the graph.





Remove all nodes with no outdegree until all nodes have outdegree at least one.

Subtask 2 (cont.)

For any remaining nodes, there exists an r that is sufficient (clearly, the maximum r will work).

Put the remaining edges into a priority queue by requirement. If we remove the last edge for a node v, then our answer for v has to be r. Then we can push a new edge, for any edges going into v.

This edge has requirement $\max(\text{req}[v], r)$.

Full Solution

The full solution is very similar to the subtask 2 solution, except when we push the new edge, the requirement is less because of p, so our requirement is $\max(\text{req}[v] - p, r)$.

To prune the graph, we can use a queue. We can prune the graph in $\mathcal{O}(N + M)$ time. Note that we will have at most $2M$ elements in our priority queue at any time, so our final complexity is:

$$\mathcal{O}(N + M \log M)$$