CS2040S: Data Structures and Algorithms

Discussion Group Problems for Week 11

For: March 31–April 4

In this week's tutorial, we will be focusing on:

- Shortest Paths
- Dijkstra's Algorithm
- Directed Acyclic Graphs

Problem 1. (Bad Dijkstra)

Problem 1.a. Give an example of a graph where Dijkstra's algorithm returns the wrong answer.

Problem 1.b. Give examples of graphs where if we used Dijkstra's algorithm, it would output the correct answer. And yet it would be unsuitable.

Problem 2. (Pizza Pirate Encounters)
Related Kattis Problems:

- https://open.kattis.com/problems/arbitrage
- https://open.kattis.com/problems/getshorty

Welcome to Mel's Pizzeria! We're here to deliver pizzas for them (given how tough the economy is, we're resorting to side hustles). We're given an undirected graph $G = (V, E)$ that represents the locations that we need to deliver to. We have source node s that represents Mel's Pizzeria, and we have a node t that represents our destination. We plan on delivering pizzas to places that we pass by on the way to home. Here's the catch! The city is rife with Pizza Pirates! With every edge e that we take, the pirates will leave us with $0 < f_e \leq 1$ fraction of whatever pizza we were carrying.

For example, if we started from node s, and took edges (s, a), (a, b), (b, t), where the edge (s, a) has fraction 0.25 the edge (a, b) has fraction 0.6 and the edge (b, t) has fraction 0.1, then the remaining pizza we have is $0.25 \times 0.6 \times 0.1 = 0.015$.

We want to maximise the amount of pizza remaining.

Problem 2.a. Design and analyze an algorithm to determine the path from a given start vertex s to a given target vertex t that maximises the fraction of pizza left. We want to run Dijkstra's. For this part, think about which parts of the algorithm we should change in order to make it work.

Problem 2.b. Are there any other ways of finding the safest path without modifying Dijkstra's algorithm? Can we modify the graph instead?

Problem 3. (Longest Path)
Given a directed acyclic graph G, compute the longest possible path that you can take in the graph.

Problem 3.a. For a node u, let $N(u)$ be the set of nodes that node u has edges to (think of $N(u)$ as the set of neighbours that node u can reach via a single edge). How do we find the longest possible path in graph G that starts at node u? Write this as a recurrence. In particular $lp(u)$ should be written in terms of $lp(v)$ for $v \in N(u)$.

Problem 3.b. For a given node u, how long does it take to compute $lp(u)$?

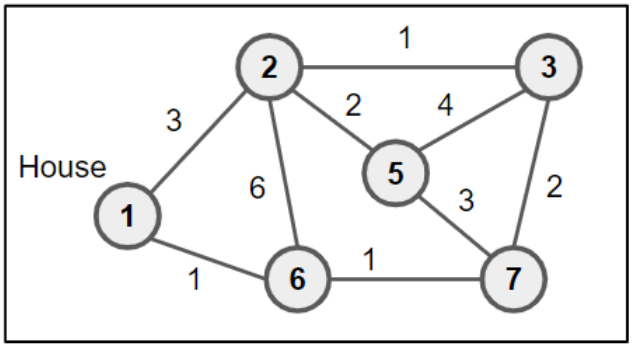
Hint: Can we say to compute this value, it is basically doing some kind of traversal on the graph? What kind?

Problem 3.c. We want to find the maximum possible, which means we wish to compute $\max_{u \in V} \{lp(u)\}$. However, whatever your answer to the previous part might be, I think we can all agree it is not $O(1)$. Which probably means naively running the solution to the previous part but v times for every node in the graph is potentially very expensive. Imagine a DAG like a linked list, and we ran the algorithm to compute $lp(u)$ for the tail node, then ran a fresh instance of the algorithm again to compute $lp(v)$ where v is $u's$ parent. And then another fresh instance to compute lp for its parent and so on. This likely does not only take $O(V + E)$ time.

Can we think of smart ordering in which we should compute the nodes so that we can compute the maximum possible path length in $O(V + E)$ time?

Hint: It's a DAG, what kind of ordering makes sense? Can we also store intermediate answers at each node to help us speed up our computation?

Problem 4. (Running Trails)



I want to go for a run. I want to go for a long run, starting from my home and ending at my home. And I want the first part of the run to be only uphill, and the second part of the run to be only downhill. I have a trail map of the nearby national park, where each location is represented as a node and each trail segment as an edge. For each node, I have the elevation (value shown in the node). Find me the longest possible run that goes first uphill and then downhill, with only one change of direction.

Hint: You might want to turn this into a kind of graph where the previous question can *just be called* on your graph to help compute the correct answer.

Problem 5. (A Random Problem with a dude called Dan)

Problem 5.a. Dan is on his way home from work. The city he lives in is made up of N locations, labelled from 0 to $(N-1)$. His workplace is at location 0 and his home is at location $(N-1)$. These locations are connected by M **directed** roads, each with an associated *non-negative* cost. To go through a road, Dan will need to pay the cost associated with that road. Usually, Dan would try to take the cheapest path home.

The thing is, Dan has just received his salary! For reasons unknown, he wants to flaunt his wealth by going through a *really expensive road.* However, he still needs to be able to make it back home with the money he has. Given that Dan can afford to spend up to D dollars on transportation, help him find **the cost of the most expensive road that he can afford to go through** on

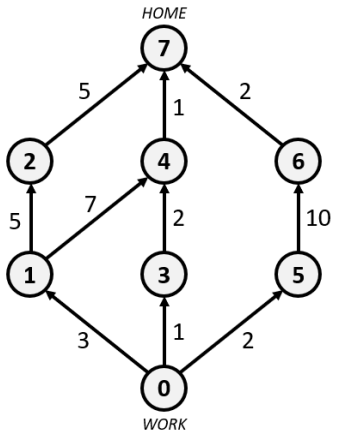


Figure 1: Example city 1

his journey back home.

Take note than Dan only cares about the most expensive road in his journey; the rest of the journey can be really cheap, or just as expensive, so long as the entire journey fits within his budget of D dollars. He is also completely focused on this goal and does not mind visiting the same location multiple times, or going through the same road multiple times.

For example, suppose Dan's budget is $D = 13$ dollars. Consider the city given in Figure 1, consisting of $N = 8$ locations and $M = 10$ roads.

The path that Dan will take is $0 \to 1 \to 4 \to 7$. In this journey, the total cost is 11 dollars and the most expensive road has a cost of 7 dollars - the road from locations 1 to 4. Therefore, the expected output for this example would be "7".

Note that this path is neither the cheapest path ($0 \to 3 \to 4 \to 7$), nor is it the most expensive path that fits within his budget of 13 dollars ($0 \to 1 \to 2 \to 7$).

There is also a more expensive road within this city - the road from locations 5 to 6 with a cost of 10 dollars. However, the only path that goes through this road, $0 \to 5 \to 6 \to 7$, has a total cost of 14 dollars which exceeds Dan's budget.

Problem 5.b. *(Optional)* Another month, another salary for Dan to flaunt. The situation is similar to that of the previous part.

This time, however, instead of maximizing the cost of the *most expensive road* in his journey, he wants to maximize the cost of *the second most expensive road* in his journey. In other words, he

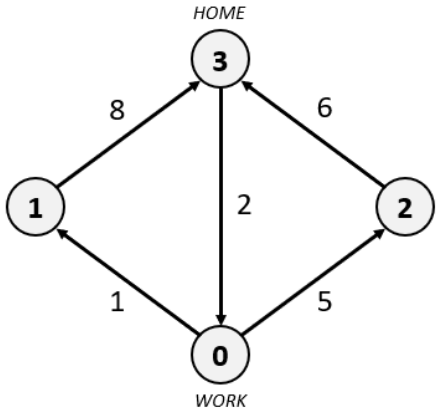


Figure 2: Example city 2

no longer cares about the most expensive road in his journey; that road can be 100 times more expensive than the second most expensive road in his journey for all he cares.

For example, consider the city in Figure 2 with $N = 4$ locations and $M = 5$ roads.

If Dan's budget is $D = 12$ dollars, the path that he will take is $0 \to 2 \to 3$. In this journey, the total cost is 11 dollars and the second most expensive road has a cost of 5 dollars, the road from locations 0 to 2. Therefore, the expected output for this example would be "5".

Notice that while he can afford to go through the path $0 \to 1 \to 3$ with an expensive 8 dollar road, the second most expensive road in that journey only costs 1 dollar.

If Dan's budget is $D = 20$ dollars, then the path he will take is $0 \to 1 \to 3 \to 0 \to 1 \to 3$. As irrational as this 20 dollar journey is, it allows him to go through the road from locations 1 to 3 twice, thus making the second most expensive road in his journey cost 8 dollars.