Exercise 11.2 *Shortest paths with cheating* **(1 point)**.

Let $G = (V, E)$ be a weighted, directed graph with weights $c : E \to \mathbb{R}_{\geq 0}$. We consider a variation of the shortest path problem in G, where we are allowed to 'cheat' by setting a certain number of weights to 0. Formally, for $k \in \mathbb{N}$, we write C_k for the set of all weight functions $\gamma : E \to \mathbb{R}_{\geq 0}$ on G with $\gamma(e) \neq c(e)$ for at most k edges $e \in E$.[2]

Given $s, t \in V$, we wish to find a path $P = (v_1 = s, v_2, \ldots, v_\ell = t)$ in G which minimizes:

$$c_k(P) := \min_{\gamma \in C_k} \gamma(P), \text{ where } \gamma(P) := \sum_{i=1}^{\ell-1} \gamma\big((v_i, v_{i+1})\big).$$

We call such a path a 'shortest path from s to t with k cheats.'

Recall that a naive implementation of Dijkstra's algorithm finds the length of a shortest path in a weighted graph (without cheating) in time $O(|V|^2)$.

$G = (V, E)$, directed / weighted. weight function $c: E \to \mathbb{R}_{\geq 0}$.

C_k is set of all weight functions $\gamma: E \to \mathbb{R}_{\geq 0}$ on G with $\gamma(e) \neq c(e)$ for at most k edges.

Given: $s, t \in V$

We want: path $P = (s = v_1, \ldots, v_\ell = t)$ which minimizes $c_k(P) := \min_{\gamma \in C_k} \gamma(P)$ where $\gamma(P)$ is sum of all edge weights in P.

(a) Describe an algorithm which finds the length of a shortest path from s to t with k cheats in time $O(|E|^k \cdot |V|^2)$. Prove that your algorithm is correct, and achieves the desired runtime.

We try all possible ways of setting k edges to 0. There are $\binom{|E|}{k}$ ways of picking k edges from $|E|$. Thus we run Dijkstra $\binom{|E|}{k}$ times, yielding

$$\binom{|E|}{k} |V|^2 \leq O(|E|^k |V|^2)$$

- k combinations of $|E|$ edges: $\binom{|E|}{k}$
- permutation with repetition: $|E|^k$

(b) Describe an algorithm which finds the length of a shortest path from s to t with k cheats in time $O\big((k|V|)^2\big)$. Prove that your algorithm is correct, and achieves the desired runtime.

Hint: *Construct a new graph $G' = (V', E')$ whose vertex set V' consists of $k+1$ copies of V. Choose the edges E' and weights c' in a clever way, and apply Dijkstra's algorithm to G'.*

We set $V' = \{v^{(\ell)} : v \in V,\ \ell \in \{0, 1, 2, \ldots, k\}\}$, which has size $|V'| = (k+1) \cdot |V|$. We define E' and $c' : E' \to \mathbb{R}_{\geq 0}$ by:

$$\forall\, (v,w) \in E,\ 0 \leq \ell \leq k: \quad (v^{(\ell)}, w^{(\ell)}) \in E' \text{ with } c'((v^{(\ell)}, w^{(\ell)})) = c((v,w)), \tag{T1}$$

$$\forall\, (v,w) \in E,\ 0 \leq \ell \leq k-1: \quad (v^{(\ell)}, w^{(\ell+1)}) \in E' \text{ with } c'((v^{(\ell)}, w^{(\ell+1)})) = 0, \tag{T2}$$

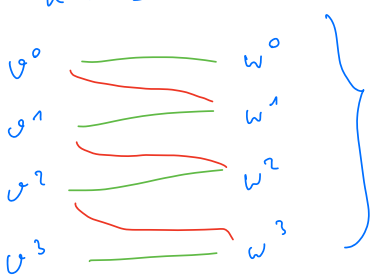
$$\forall\, 0 \leq \ell \leq k-1: \quad (t^{(\ell)}, t^{(\ell+1)}) \in E' \text{ with } c'((t^{(\ell)}, t^{(\ell+1)})) = 0. \tag{T3}$$

(T1) normal edge traversal in G

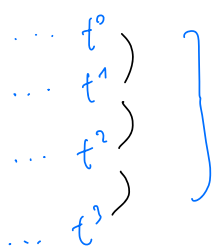
(T2) use one cheat

(T3) connects all $t^{(\ell)}$ with each other

Suppose $k = 3$



When we cheat we go one layer down. The graph construction ensures max. k layers down (i.e. max. k cheats)



we can always end up at $t^{(k)}$

We run Dijkstra's algorithm on (G', c') to find the length of a shortest path between $s^{(0)}$ and $t^{(k)}$ in time $O(|V'|^2) \leq O\big((k|V|)^2\big)$.

Correctness:

Let L be the length of a shortest path from s to t in (G, c) with k cheats.

Let L' be ... $s^{(0)}$ to $t^{(k)}$ ... .

We want to show $L = L'$.

(i) $L' \geq L$

because then $L' = \infty$

We show first that $L' \geq L$. If there is no path from $s^{(0)}$ to $t^{(k)}$ in (G'), then surely $L' \geq L$, so we may assume there is a shortest path $P' = (v_1^{(a_1)}, \ldots, v_\ell^{(a_\ell)})$ from $s^{(0)}$ to $t^{(k)}$ in (G', c'). By definition of E', we know that $P = (v_1, v_2, \ldots, v_\ell)$ is a path in G from s to t (after possibly removing some subsequent copies of t from the path, corresponding to edges of type (T3), which have weight 0). Since there are at most k edges of type (T2) in P', we conclude that there is a weight function $\gamma \in C_k$ such that $\gamma(P) = c'(P')$. But that means $L \leq c_k(P) \leq L'$.

since $c_k(P)$ is minimal.

(ii) $L \geq L'$

Next we show $L \geq L'$. Let $P = (v_1, v_2, \ldots, v_\ell)$ be a shortest path from s to t in G with k cheats. Again, we may assume such a path exists. Let $\gamma \in C_k$ be the weight function for which $\gamma(P) = c_k(P) = L$. Now, consider the path $P' = (v_1^{(a_1)}, \ldots, v_\ell^{(a_\ell)})$ in G', where $a_1, \ldots, a_\ell$ are defined by $a_1 = 0$, and, for $1 \leq i \leq \ell$,

$$a_i = \begin{cases} a_{i-1} + 1 & \text{if } c((v_{i-1}, v_i)) \neq \gamma((v_{i-1}, v_i)), \\ a_{i-1} & \text{if } c((v_{i-1}, v_i)) = \gamma((v_{i-1}, v_i)). \end{cases}$$

This path is well-defined because $c((v_{i-1}, v_i)) \neq \gamma((v_{i-1}, v_i))$ for at most k different i. By definition of c', the length of P' in (G', c') is at most L. We know that $v_1^{(a_1)} = s^{(0)}$, and $v_\ell^{(a_\ell)} = t^{(i)}$, for some $0 \leq i \leq k$. We can thus extend P' to be a path P'' from $s^{(0)}$ to $t^{(k)}$ using edges of type (T3) (whose cost is zero). We conclude that $L' \leq c'(P'') \leq \gamma(P) \leq L$.

(I) (II) (III)

(I) since L' is the shortest path in G'

(II) since γ has the same edge weights as c except for k edges. if we encounter a different edge weight ($a_{i-1} + 1$ if $c((v_{i-1}, v_i)) \neq \gamma((v_{i-1}, v_i))$,) we change layers, i.e. we use a (T2) edge, which has weight zero. γ can't be lower than zero.

(III) $\gamma(P) = L$.

Exercise 11.4 *Driving from Zurich to Geneva* **(1 point).**

(Slides)

(a) Model the problem as a graph problem such that you can directly apply one of the algorithms in the lecture, without modifications to the algorithm:

(1) Describe your graph. What are the vertices, what are the edges and the weights of the edges?

The graph $G = (V, E, w)$ is defined as follows: V is the set of cities on his map of Europe. There are directed edges between the cities (in both directions) if they are connected by a highway. The weight of any edge is the difference of the cost he needs to pay for the fuel for this highway and the money that a passenger would pay him (if available, otherwise it is just the cost he needs pay).

$V :=$ cities in Europe (on his map)

$E :=$ contains $e = (v, w)$ if v is connected to w via highway. (directed) and (w, v)

$c : E \to \mathbb{Z}$

$c(e = (v, w)) =$ (fuel cost) $-$ (passenger pay)

if available from v to w.

(2) What is the graph problem that we are trying to solve?

Solution:

We are trying to find the shortest path from the vertex v_{Zurich} corresponding to Zurich and the vertex v_{Geneva} corresponding to Geneva in the graph G. Note that since from no city he has a roundtrip that gains him money, the graph G does not have negative cycles, so we are indeed looking for a path and not a walk.

single-source shortest path.

source: v_{Zurich}, destination: v_{Geneva}

(3) Solve the problem using an algorithm discussed in the lecture (without modification).

Solution:

We can apply the Bellman-Ford algorithm to (G, v_{Zurich}) to find the shortest path from v_{Zurich} to v_{Geneva} (we have no negative cycles, so Bellman-Ford works). Note that the edge weights might be negative, so we cannot apply Dijkstra's algorithm.

(slides)

(b) Now we change the problem slightly. Bob got a list from his friend of certain highways that are in a bad condition. To not damage his car, he decided that he want to use at most one of these highways. Again, model the problem as a graph problem such that you can directly apply one of the algorithms in the lecture, without modifications to the algorithm:

(1) Describe your graph. What are the vertices, what are the edges and the weights of the edges?

We define the graph $G' = (V', E', w')$ as follows: The set of vertices is $V' = V \times \{0, 1\}$. For every edge $e = (u, v) \in E$, if it is a normal highway, we have the edges $((u, 0), (v, 0))$ and $((u, 1), (v, 1))$ in E'. If it is a highway in bad condition, then we have the edge $((u, 0), (v, 1))$ in E'. The weight function w' is as before: The weight of any edge is the difference of the cost he needs to pay for the fuel for this highway and the money that a passenger would pay him (if available, otherwise it is just the cost he needs pay).

G from (a). $G' = (V', E', w')$

- $V' = V \times \{0,1\}$ ← again, layers idea!
- E' contains (u^0, v^0) and (u^1, v^1) if $e = (u, v)$ is normal, and (u^0, v^1) if e is bad.
- w' stays the same

This construction can be interpreted as follows: We have two levels in the graph G', namely $V \times \{0\}$ and $V \times \{1\}$ and they are connected in such a way that it matches the problem description. Highways in bad condition connect layer 0 to layer 1 (and only in this direction). All other highways are present in both layer 0 and layer 1 (and do not cross between layers). Consider a path in this modified graph G'. Since we have no edges from layer 1 to layer 0 and whenever a highway in bad condition is used, the path moves from layer 0 to layer 1, this path can use at most one highway in bad condition.

(2) What is the graph problem that we are trying to solve?

Solution:

We are trying to find the shortest path between $(v_{\text{Zurich}}, 0)$ and $(v_{\text{Geneva}}, 0)$ or $(v_{\text{Geneva}}, 1)$ (whichever is shorter). Again, since the graph has no negative cycles, we are indeed looking for a path and not just a walk. Since we argued above that a path in G' uses at most one highway in bad condition, this is indeed what we are looking for (paths between $(v_{\text{Zurich}}, 0)$ and $(v_{\text{Geneva}}, 0)$ use no highway in bad condition and paths between $(v_{\text{Zurich}}, 0)$ and $(v_{\text{Geneva}}, 1)$ use exactly one).

Remark: If we wanted to model the problem as finding a single shortest path, we could add a vertex t to the graph and add the two edges $((v_{Geneva}, 0), t)$ and $((v_{Geneva}, 1), t)$. The weight of these two edges can be set to 0. Then the goal would to find a shortest path between $(v_{Zurich}, 0)$ and t. The shortest path from Zurich to Geneva would then be this shortest path without the last edge $((v_{Geneva}, i), t)$.

(3) Solve the problem using an algorithm discussed in the lecture (without modification).

Solution:

We can again apply the Bellman-Ford algorithm to the modified graph $(G', (v_{\text{Zurich}}, 0))$ to find the shortest paths from $(v_{\text{Zurich}}, 0)$ to $(v_{\text{Geneva}}, 0)$ and $(v_{\text{Geneva}}, 1)$ (we again have no negative cycles, so Bellman-Ford works).