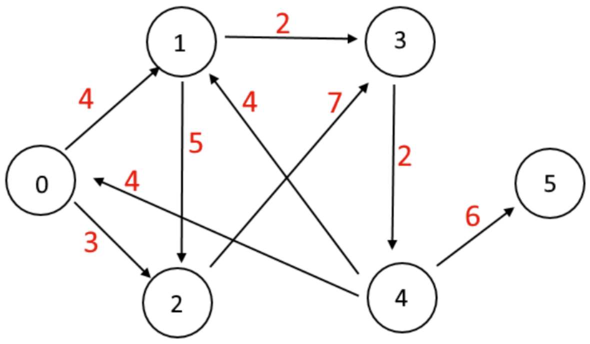
# EL-GY 9343 Homework 12

Yihao Wang
yw7486@nyu.edu

1. Consider the following graph, with node 0 as the source node.



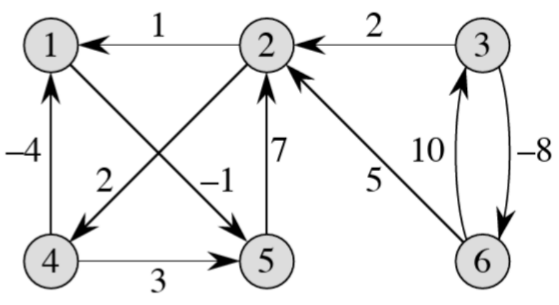
(a) Run Dijkstra's algorithm. Write down the $d$ array before each EXTRACT-MIN, also the final array.

| iteration $i$ | $d$ array (before iteration $i$) | | | | | | |
|---|---|---|---|---|---|---|---|
| | index | 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | | 0 | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ |
| 2 | | 0 | 4 | 3 | $\infty$ | $\infty$ | $\infty$ |
| 3 | | 0 | 4 | 3 | 10 | $\infty$ | $\infty$ |
| 4 | | 0 | 4 | 3 | 6 | $\infty$ | $\infty$ |
| 5 | | 0 | 4 | 3 | 6 | 8 | $\infty$ |
| 6 | | 0 | 4 | 3 | 6 | 8 | 14 |

(b) Run Bellman-Ford algorithm. Write down the $d$ distance array after each pass.

| vertex index | $d$ array (after pass $i$) | | | | | | |
|---|---|---|---|---|---|---|---|
| | index | 0 | 1 | 2 | 3 | 4 | 5 |
| 1 | | 0 | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ |
| 2 | | 0 | 4 | 3 | $\infty$ | $\infty$ | $\infty$ |
| 3 | | 0 | 4 | 3 | 6 | $\infty$ | $\infty$ |
| 4 | | 0 | 4 | 3 | 6 | 8 | $\infty$ |
| 5 | | 0 | 4 | 3 | 6 | 8 | 14 |

2. Run the Floyd-Warshall algorithm on the following graph. Show the matrix $D^{(k)}$ that results for each iteration of the outer loop.



$$D^{(0)} = \begin{pmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ 1 & 0 & \infty & 2 & \infty & \infty \\ \infty & 2 & 0 & \infty & \infty & -8 \\ -4 & \infty & \infty & 0 & 3 & \infty \\ \infty & 7 & \infty & \infty & 0 & \infty \\ \infty & 5 & 10 & \infty & \infty & 0 \end{pmatrix} \qquad D^{(1)} = \begin{pmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ 1 & 0 & \infty & 2 & 0 & \infty \\ 3 & 2 & 0 & 4 & 2 & -8 \\ -4 & \infty & \infty & 0 & -5 & \infty \\ 8 & 7 & \infty & 9 & 0 & \infty \\ 6 & 5 & 10 & 7 & 5 & 0 \end{pmatrix}$$

$$D^{(2)} = \begin{pmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ 1 & 0 & \infty & 2 & 0 & \infty \\ 3 & 2 & 0 & 4 & 2 & -8 \\ -4 & \infty & \infty & 0 & -5 & \infty \\ 8 & 7 & \infty & 9 & 0 & \infty \\ 6 & 5 & 10 & 7 & 5 & 0 \end{pmatrix} \qquad D^{(3)} = \begin{pmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ 1 & 0 & \infty & 2 & 0 & \infty \\ 3 & 2 & 0 & 4 & 2 & -8 \\ -4 & \infty & \infty & 0 & -5 & \infty \\ 8 & 7 & \infty & 9 & 0 & \infty \\ 6 & 5 & 10 & 7 & 5 & 0 \end{pmatrix}$$

$$D^{(4)} = \begin{pmatrix} 0 & \infty & \infty & \infty & -1 & \infty \\ -2 & 0 & \infty & 2 & -3 & \infty \\ 0 & 2 & 0 & 4 & -1 & -8 \\ -4 & \infty & \infty & 0 & -5 & \infty \\ 5 & 7 & \infty & 9 & 0 & \infty \\ 3 & 5 & 10 & 7 & 2 & 0 \end{pmatrix} \qquad D^{(5)} = \begin{pmatrix} 0 & 6 & \infty & 8 & -1 & \infty \\ -2 & 0 & \infty & 2 & -3 & \infty \\ 0 & 2 & 0 & 4 & -1 & -8 \\ -4 & 2 & \infty & 0 & -5 & \infty \\ 5 & 7 & \infty & 9 & 0 & \infty \\ 3 & 5 & 10 & 7 & 2 & 0 \end{pmatrix}$$

$$D^{(6)} = \begin{pmatrix} 0 & 6 & \infty & 8 & -1 & \infty \\ -2 & 0 & \infty & 2 & -3 & \infty \\ -5 & -3 & 0 & -1 & -6 & -8 \\ -4 & 2 & \infty & 0 & -5 & \infty \\ 5 & 7 & \infty & 9 & 0 & \infty \\ 3 & 5 & 10 & 7 & 2 & 0 \end{pmatrix}$$

3. Let $G(V, E)$ be a directed, weighted graph. The goal is to find the shortest path from a source node $s$ to every nodes in the graph.

(a) If all edges have unit weight, is there an algorithm faster than Dijkstra? Please just state the algorithm and its running time.

Directly using BFS is faster, with a running time of $O(|V| + |E|)$.

(b) Now suppose the weight function has the form $w(\cdot) : E \to \{1, 2\}$, i.e. every edge has weight either 1 or 2. Design $O(|V| + |E|)$-time algorithm for the problem. Please show why your algorithm works and how to get the running time bound.

We can manage to transform this problem into that in (a) by inserting some auxiliary vertices to all edges that have weight of 2. Now we can run BFS on the newly constructed graph, which yields a running time of $O(2|V| + 2|E|) = O(|V| + |E|)$.

4. The goal of this problem is to travel from home to a store, purchase a gift, and then get back home, at minimal cost.

Let us model this problem using a directed graph. Let $G(V, E)$ be a directed, weighted graph, with non-negative edge weights $w : E \to \mathbb{R}^+$. The weight of an edge represents the cost of traversing that edge. Each vertex $v \in V$ also has an associated cost $c(v) \in \mathbb{R}^+$ which represents the cost of purchasing the desired gift at that location.

Starting from "home base" $h \in V$, the goal is to find a location $v \in V$ where the gift can be purchased, along with a path $p$ from $h$ to $v$ and back from $v$ to $h$. The cost of such a solution is the cost $c(v)$ of the location $v$ plus the weight $w(p)$ of the path $p$ (i.e., the sum of edge weights along the path $p$).

Design an algorithm that on input $G(V, E)$, including edge weights $w(\cdot)$ and costs $c(\cdot)$, and home base $h \in V$ , finds a minimal cost solution. Assuming $G$ is represented using adjacency lists, and Dijkstra's algorithm runs in $O(|E| \log |V|)$. Your algorithm should run Dijkstra **exactly once** and in time $O((|V| + |E|) \log |V|)$. Please show why your algorithm works and analyze the time complexity.

**Hints**: If you try to solve with the original graph $G$, then most likely you will have to run multiple times of Dijkstra. You can try to extend the graph $G$ into two layers. What is the number of nodes and edges?

1. Split all vertices into two pieces, one named $v_i^{in}$, which are connected with all in-edges, and another named $v_i^{out}$, which are connected with all out-edges.
2. For all $i$, connect $v_i^{in}$ to $v_i^{out}$ with a one-way edge with weight of $c(v_i)$. Then assign $\infty$ to the edge $(h^{in}, h^{out})$.
3. Run Dijkstra's algorithm on the newly constructed graph $G'$ once. Then we can get the minimum cost from $h^{in}$ to $h^{out}$.

The newly constructed graph $G'$ has no more than $2|V|$ nodes and no more than $|E| + |V|$ edges. Therefore, the running time on $G'$ is $O((|E| + |V|) \log(2|V|)) = O((|V| + |E|) \log |V|)$.