CS2800 Design and Analysis of Algorithms
Problem Set #5 February 27, 2023

1. Given a graph G and a source vertex s, modify the Bellman-Ford algorithm to check if there is a negative weight cycle reachable from s, and if so, find one such cycle.

2. Given a graph G and two vertices s and t, the replacement path P_e is the shortest path from s to t in $G - e$. Suppose that the shortest path in G from s to t passes through every vertex in G, give an $O(E \log V)$-time algorithm to compute the distance of replacement paths P_e for every edge e. Assume that the graph has no negative weight edges.

3. For a weighted graph G and two vertices s and t, the *best path* between s and t is the path with minimum weight that contains the least number of edges. Assuming that the weights of the edges in the graph are non-negative, modify Dijkstra's algorithm to obtain the best paths from s to all the other vertices in the graph.

4. Given a directed graph G with non-negative edge weights, and two vertices s and t, design an algorithm to find the shortest *walk* from s to t (shortest in terms of sums of weights of edges in the walk) such that the number of edges in the walk is a multiple of 3.

   A *walk*, as opposed to a path, is a sequence of moves in the graph where vertices can repeat.

5. Consider a weighted digraph $G(V, E)$ whose edge weights change over time. You can think of your digraph as representing the road network of a city where an edge $e = (u, v)$ is the road connecting two junctions u and v. Naturally, the time to traverse that road depends on the traffic, and this varies over time.

   To model this situation, assume that for every edge e there is a function $f_e : \mathbb{N} \to \mathbb{N}$ (which is its weight that varies over time) such that the following conditions hold:

   - You don't own a time machine - when crossing $e = (u, v)$, you cannot reach v at a time before the time at which you were at u: More formally,
   $$\forall t, f_e(t) > 0.$$
   - You don't overtake within city limits - while crossing $e = (u, v)$, if your friend reaches u earlier than you, then you cannot reach v earlier than your friend. More formally,
   $$t_1 \leq t_2 \Rightarrow t_1 + f_e(t_1) \leq t_2 + f_e(t_2).$$

   Given such a digraph $G(V, E, f)$, and two vertices s and t, design an algorithm to find the path that takes shortest time to commute from s to t. Assume that you start from s at $t = 0$, and you don't care about the number of roads you take, but just the total commute time. Furthermore, assume that given t, you can compute $f_e(t)$ in $O(1)$-time for every $e \in E$.

6. Consider the greedy algorithm for the interval scheduling problem that we saw in class. Let $\mathcal{I}$ be an instance of the problem and let OPT be the maximum number of non-overlapping intervals in $\mathcal{I}$. Let k be the number of intervals returned by the shortest-interval first algorithm. Show that $k \geq \mathrm{OPT}/2$.