Problem Set 2

This problem set is due at **10:00 am** on **Tuesday, February 7th**.

Problem 2- 1: Pedantic Sandy

Sandy would like to travel on a graph $G = (V, E, w : E \to \mathbb{R}^{\geq 0})$, i.e. a graph with vertex set V, edge set E and non-negative edge weights, which are determined by w. She starts with her left foot on the first vertex. She can only place one foot on any given vertex and alternates feet (i.e. if she starts with a left foot on the first vertex, the next vertex she's on will have her right foot on it). Sandy wants to know what the shortest path to every vertex is, such that the foot which ends on the final vertex is the left foot. Write an algorithm to help her do this. (HINT: Consider the graph G' with vertex set $V' = V \times \{\textit{left}\} \cup V \times \{\textit{right}\}$ edge set $E' = \{((u, \textit{left}), (v, \textit{right}))|(u, v) \in E\} \cup \{((u, \textit{right}), (v, \textit{left}))|(u, v) \in E\}$ and weight function $w'((u, \cdot), (v, \cdot)) = w((u, v))$. Use G' to solve the problem)
Write a paragraph or a sentence summarizing the problem you are solving and what your results are, then describe your algorithm in English and, if helpful, pseudocode. Prove your algorithm is correct and analyze the running time of the algorithm.

Proof. *Note:* This solution follows the hint given in the problem with slightly different notation.

Construct a graph $G' = (V', E')$ such that for every vertex $v_i \in V$, there exist vertices v_i^0 and v_i^1 in V'. For every edge $(u, v) \in E'$, there exists edges (u^0, v^1) and (u^1, v^0) in E'. Weight the edges in G' with w' such that $w'(u^0, v^1) = w'(u^1, v^0) = w(u, v)$. Run Dijkstra's algorithm on G' to find the shortest paths of G' starting from v_s^0. The length of shortest path to any vertex v_t in the original graph G, with the constraint of ending on a left foot, is equivalent to the shortest path from v_s^0 to v_t^0 in G'.

0.1 Correctness

Lemma 1 *Every path from v_s^0 to v_t^0 in G' has an odd number of vertices.*

Proof. Suppose for the sake of contradiction there exists some path from v_s^0 to v_t^0 that has an even number of vertices. Let p_e be the shortest such path in G'. Note that because of the way G' is constructed, there only exist edges of the form (u^0, v^1) or (u^1, v^0) in G'. If p_e only has the two vertices (v_s^0, v_t^0), then there must exist an edge from v_s^0 to v_t^0, which is a contradiction. If $p_e = (v_s^0, v_2, ..., v_{n-1}, v_t^0)$ has more than 2 edges, the edge (v_{n-1}, v_t^0) must be in the form of (v_i^1, v_t^0). Consequently, the edge (v_{n-2}, v_{n-1}) must be in the form

of (v_j^0, v_i^1). Consider the path $p'_e = (v_s^0, v_2, ..., v_{n-2})$. v_{n-2} must be in the form of v_j^0, and p'_e is an even length path, which contradicts our supposition that p_e is the shortest path that exhibits this property.

Lemma 2 *Any odd length path $p' = (v_0^0, v_1^1, v_2^0, \ldots, v_n^0)$ in G' corresponds to an odd length path $p = (v_0, v_1, v_2, \ldots, v_n)$ in G such that $w'(p') = w(p)$.*

Proof. First, note that p' has n vertices, and p has n vertices, p' is odd iff p is also odd.

$$\begin{aligned} w'(p') &= \sum_{i=0}^{n/2-1} w'(v_{2i}^0, v_{2i+1}^1) + w'(v_{2i+1}^1, v_{2i+2}^0) \\ &= \sum_{i=0}^{n/2-1} w(v_{2i}, v_{2i+1}) + w(v_{2i+1}, v_{2i+2}) \\ &= \sum_{i=0}^{n-1} w(v_i, v_{i+1}) \\ &= w(p) \end{aligned}$$

Theorem 3 *The length of the shortest path from v_s^0 to v_t^0 in G' is the length of the shortest odd length path from v_s to v_t in G.*

Proof. Let p' be the shortest path from v_s^0 to v_t^0 in G'. By the first lemma, this path is odd. By the second lemma, there exists a p in G from v_s to v_t in G which is also odd, such that $w'(p') = w(p)$. Suppose for the sake of contradiction there exists a odd length path q from v_s to v_t in G that that has lower weight $w(q) < w(p)$. By the second lemma, there exists a corresponding path q' from v_s^0 to v_t^0 in G' such that $w'(q') = w(q)$. This implies $w'(q') = w(q) < w(p) = w'(p')$, which contradicts that p' is the shortest path.

If Sandy wishes to end on her left foot, then she must step on an odd number of vertices. The algorithm will therefore find the shortest paths to each vertex such that she ends on her left foot.

0.2 Runtime

The algorithm first constructs G', then runs Dijsktra's algorithm of G'. Constructing G' involves duplicating every vertex and every edge once, and thus takes $O(|V| + |E|)$ time if using an adjacency list. G' will have $2|V|$ vertices and $2|E|$ edges, so the second part of the algorithm will run in $O(2|E| \log(2|V|)) = O(|E| \log |V|)$ time. The whole algorithm thus runs in $O(|E| \log |V| + |V| + |E|) = O(|E| \log |V|)$ time (assuming $|V| = O(|E|)$).

Problem 2- 2: OCD

You have to store L gallons of oil where L is a natural number. You own a factory that can make as many 1-gallon, 2-gallon, 2^2-gallon, ..., 2^{1000}-gallon containers as you would like. However, you would like to store the L gallons of oil in as few different containers as possible while ensuring that every container you store the oil in is full. Consider the following algorithm -

Input: L
Result: Smallest number of containers you can fill L gallons of oil in, while ensuring that no container has any space left and the list the capacities of these containers
$L' = L$;
$count = 0$;
$Arr = []$;
while $L' > 0$ **do**
  $A = \max\{2^i \leq L' | i \in [1000]\}$;
  $count = count + 1$;
  $Arr = Arr.append(A)$;
  $L' = L' - A$;
end
Output: $(count, Arr)$

Does this algorithm terminate and give you a correct answer? If yes, write a proof. If no, give a counterexample.

Proof.

Proof that the algorithm terminates:
Direct proof:
For $L > 0$, the algorithm terminates because at every iteration $L' = L' - A$ and $A \geq 2^0 = 1$. At the very least 1 is being subtracted from L' at every iteration of the while loop. When $L' = 0$ the algorithm will exit the while loop and stop. After at most L iterations, the algorithm would have stopped.

Proof that the algorithm is correct:

The optimal solution can be written as

$$O_0 \cdot 2^0 + O_1 \cdot 2^1 + O_2 \cdot 2^2 + ... + O_{1000} \cdot 2^{1000}$$

The algorithm's solution can be written as

$$A_0 \cdot 2^0 + A_1 \cdot 2^1 + A_2 \cdot 2^2 + ... + A_{1000} \cdot 2^{1000}$$

Where O_n and A_n represent the number of containers of size 2^n

Observation 1 (a): $0 \leq O_n \leq 1$ for all $0 \leq n < 1000$. Suppose not, then there exists O_k where $k < 1000$ such that $O_k \geq 2$. Observe that two containers of size 2^k can be replaced with one container of size 2^{k+1}, hence reducing the number of containers overall, contradicting the optimality of the choices of O_i.

Observation 1(b): $0 \leq A_n \leq 1$ for all $0 \leq n < 1000$. Suppose not, then there exists $A_k, k < 1000$ such that $A_k \geq 2$. By an observation similar to above, there is a larger container that could have matched 2 containers of size A_k and the greedy algorithm would have chosen that container instead.

Observation 2: There is no way to contain 2^n gallons with only one of each of the containers less than 2^n. This is because $2^n = 2 \cdot 2^{n-1} = 2^{n-1} + 2^{n-1} = 2^{n-1} + 2^{n-2} + 2^{n-2}$ and so on, and hence $2^n - 1 = \sum_{i=0}^{n} 2^i$. There is no way to get 2^n from natural numbers of the kind 2^i where $i < n$, such that each term occurs in the sum at most once.

Proof by contradiction:

Assume for the sake of contradiction that the algorithm does not give the correct output. Then, there must be an optimal solution that uses less containers than the algorithm.

$$\sum_{n=0}^{1000} O_n < \sum_{n=0}^{1000} A_n$$

Which means $O_n \neq A_n$ for some $0 \leq n \leq 1000$.

Let i be the greatest number for which $O_i \neq A_i$. A_i must be greater than O_i because the greedy algorithm always picks the maximum container that will contain all the oil without space left over. Thus at that point, the optimal solution contains 2^i less oil than the output of the greedy algorithm.

By Observation 2 it is not possible for the optimal solution to make up that loss by using at most one container of each smaller size. This contradicts $\sum_{i=0}^{1000} O_i 2^i = \sum_{i=0}^{1000} A_i 2^i$.