CS161 Homework 5 **Due:** 20 May 2016
Submit on Gradescope **Handed out:** 13 May 2016

Instructions: Please answer the following questions to the best of your ability. If you are asked to design an algorithm, please describe it clearly and concisely, prove its correctness, and analyze its running time. If you are asked to show your work, please include relevant calculations for deriving your answer. If you are asked to explain your answer, give a short ($\sim$ 1 sentence) intuitive description of your answer. If you are asked to prove a result, please write a complete proof at the level of detail and rigor expected in prior CS Theory classes (i.e. 103). When writing proofs, please strive for clarity and brevity (in that order). Cite any sources you reference.

1 (14 points) Poe Selling Fish

Poe the Penguin is setting up a business to sell fish to his fellow penguin friends. He predicts the quantity of sales to expect over the next n days. Let d_i denote the number of sales he expects on day i. Poe procures fish from Barr the Bear in batches, and Barr requires a fixed fee of K each time Poe requests a batch (regardless of the number of fish Poe requests; also Poe doesn't need to pay for the fish, just the fee). We will assume that Poe gets a batch of fish (if any) in the morning, and all sales happen in the afternoon. Any fish that are not sold are stored until the next day. Poe can store at most S fish, and it costs C to store a single fish for a day. Poe starts out with no fish on day 1.

In this problem we will design a dynamic programming algorithm that decides how to place requests so that Poe satisfies all the demands d_i, of his penguin friends and minimizes his costs. (Assume that S, C, K, d_i's are all positive integers.)

(a) (2 points) Let the subproblem represent the optimal way to satisfy demands for days 1 to i with a total of s fish left over after day i. Let $M[i, s]$ denote the cost of an optimum solution to this subproblem (but not including the storage fee of the unsold fish on day i, if any). What values of i and s should we plug in for the problem we want to solve? How many subproblems do we have in total?

(b) (2 points) Let us try to solve a subproblem $M[i, s]$ using the values of the previous subproblems $M[i-1, z]$ where $z \in [0, S]$. $M[i-1, z]$ is the total cost incurred by the end of day $i-1$, and Poe will have to pay zC for storage (if $z = 0$, this will be 0). On day i, recall that Poe needs to sell d_i fish and have s fish remaining after the sale. Let us consider two cases as follows.

(i) If $s + d_i > S$, then Poe must request a batch on day i (regardless of what z is because $z \leq S$). In this case, we can solve $M[i, s]$ as follows:

$$M[i, s] = \min_{z \in [0,S]} (M[i-1, z] + zC + K)$$

(ii) If $s + d_i \leq S$, then depending on what z is, Poe may not need to order fish (to avoid the fee):

$$M[i, s] = \min \left(\min_{z \in [0, s+d_i-1]} (M[i-1, z] + zC + K), (M[i-1, s+d_i] + (s+d_i)C) \right)$$

What should the base casse be? If we use this to solve the problem, what would be the runtime of this algorithm?

(c) (7 points) It turns out that we can optimize the algorithm presented above, and get a faster algorithm. In both cases shown above, we can show that $z = 0$ achieves the minimum (where min is computed over a range of z). Prove this.

(d) (3 points) Let us define $M[i,s] = \infty$ if ($i > 0$ and $s > S$) or ($i = 0$ and $s > 0$) and define $M[0,0] = 0$. Fill in the blanks below using Part (b) and (c). If we use this to solve the problem, What is the runtime of the new algorithm?

$$M[i,s] = \min\left(\boxed{}, \boxed{}\right).$$

2 (14 points) Fun with Walks

Consider a weighted, directed graph G with n vertices and m edges that have integer weights. A graph walk is a sequence of not-necessarily-distinct vertices $v_1, v_2, \ldots, v_k$ such that each pair of consecutive vertices v_i, v_{i+1} are connected by an edge. This is similar to a path, except a walk can have repeated vertices and edges. The length of a walk in a weighted graph is the sum of the weights of the edges in the walk. Let s, t be given vertices in the graph, and L be a positive integer. We are interested counting the number of walks from s to t of length exactly L.

a) (5 points) Assume all the edge weights are positive.

Describe an algorithm that computes the number of graph walks from s to t of length exactly L in $O((n+m)L)$ time. Prove the correctness and analyze the running time.

b) (9 points) Now assume all the edge weights are non-negative (but they can be 0), but there are no cycles consisting entirely of zero-weight edges. That is, for any cycle in the graph, at least one edge has a positive weight.

Describe an algorithm that computes the number of graph walks from s to t of length exactly L in $O((n+m)L)$ time. Prove correctness and analyze running time.

3 (12 points) Shortest and Longest Paths

Consider a (directed) grid network $G = (V, E)$ in which each node is labeled as $v_{i,j}$ where $1 \le i \le r$ and $1 \le j \le c$, and there is an edge from $v_{i,j}$ to $v_{i,j+1}$ if and only if $j < c$ and an edge from $v_{i,j}$ to $v_{i+1,j}$ if and only if $i < r$. (That is, the edges go from left to right and top-down to adjacent grid points.) For each edge e in the grid network let w_e denote its weight. We will assume that w_e's are integers which can be positive or negative.

For example, Figure 1 shows a grid network when $r = 4$ and $c = 5$ (weights are not shown).

We will use this definition of grid networks in the sub-problems of this question.

(a) (6 points) We want to compute the distance from $v_{1,1}$ to $v_{r,c}$.

Explain how we can use Dijkstra's algorithm to solve this problem. (Recall that edge weights can also be negative.) Prove its correctness, and analyze its running time. Assume that Dijkstra's algorithm is implemented with Fibonacci Heaps.

(b) (6 points) We want to compute the longest path among all paths in this grid network.

Design an algorithm that runs in $O(rc)$ time and finds the longest path. Prove its correctness and analyze its running time. (Recall that edge weights can be negative.)

4 (12 points) Poe Travels with Clones of Barr

Poe the Penguin is a mad scientist, and recently invented a machine that produces clones of Barr the Bear from South Pole dark matter. Poe has produced n clones of Barr the Bear (let us label them as $1, 2, \ldots, n$). Poe wishes to travel the universe with *exactly k clones* as his spaceship has $k+1$ seats (one for himself and k for the clones – Poe does not want to have empty seats!). The machine has a minor flaw in that some clones, if they are separated, would explode, and put an end to the universe. Luckily Poe knows which pairs of clones need to be kept together; let $E = \{(i_1, j_1), (i_2, j_2), \ldots, (i_m, j_m)\}$ be a set of pairs of clones where

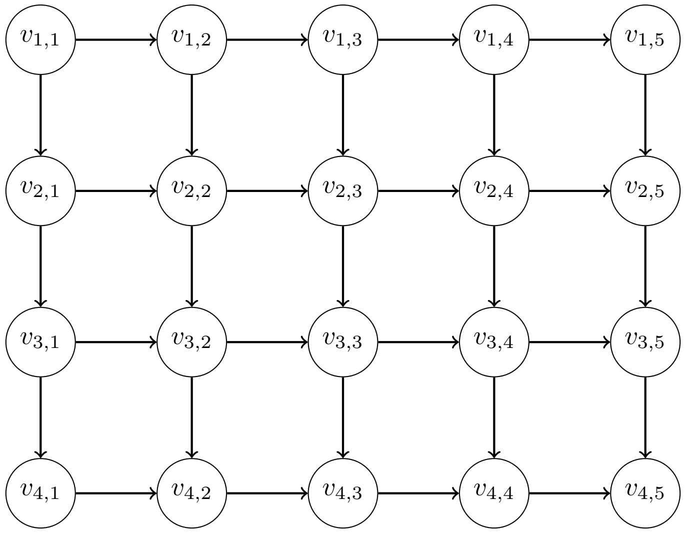

Figure 1: Example: Directed Grid Network of Size 4×5.

each pair (i, j) indicates that clone i and clone j should not be separated; that is, Poe must either take both i and j with him on his spaceship or take neither of them. Can you help Poe find out whether it is possible to choose exactly k clones and take them with him, or Poe should just stay home while regretting his latest invention?

Design an algorithm that takes (n, k, E) as an input, and determines in $O(m + nk)$ runtime which k clones Poe should take with him (or output "S.T.A.Y." if Poe should not travel). Prove correctness of your algorithm and analyze its running time. (You may assume that $k \leq n$ because Poe only has n clones and thus cannot travel with exactly k clones if $k > n$.)