# Problem Set 4

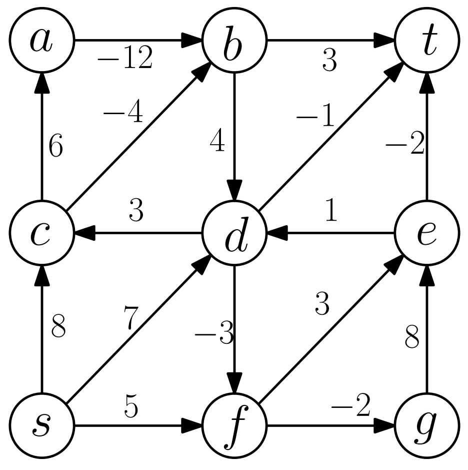
CMSC 27230: Honors Theory of Algorithms

Assigned January 30, Due February 13

## Problem 1: Bellman-Ford (25 points)

(a) 15 points: Use the Bellman-Ford algorithm to find the minimum length of a path from $s$ to $t$ in the following graph. For your answer, for all $v \in V(G) \setminus \{s\}$ and all $k \in N$, give the minimum total length of a walk from $s$ to $v$ which uses at most $k$ edges. Also give the shortest path(s) from $s$ to $t$.

Note: You may stop when you get the same answers for $k$ as you did for $k - 1$. Also, if you wish, you may solve this problem by writing a program for it and having the program output the answers. If you do this, you should submit your code as part of your answer.



(b) 10 points: What would happen if the edge from $d$ to $c$ had length 1 instead of length 3?

Note: For this part, you should describe what happens and why, but you do not need to run the Bellman-Ford algorithm again (though you are welcome to do so) .

## Problem 2: Word Segmentation (Rephrasing of Chapter 6, Problem 5 of Kleinberg-Tardos, 25 points)

In the word segmentation problem, we are given a string of characters $y_1 \ldots y_n$ and we are asked to segment these characters into words. For example, given the string "belowfreezing" we would want to decompose it into "below freezing". To model this problem, assume that we have a quality function $q$ which takes a string $x_1 \ldots x_k$ and outputs a quality score $q(x_1 \ldots x_k)$ based on how close it is to a word and how common the word is. Given this quality function $q$, we want to find a segmentation which maximizes the total quality of the segments. In other words, we want to find a sequence $i_i, \ldots, i_m$ such that $1 = i_1 < i_2 < \cdots < i_m \leq n$ and

$$\sum_{j=1}^{m} q(y_{i_j} \ldots y_{i_{j+1}-1})$$

is maximized where we take $i_{m+1} = n + 1$.

Give a polynomial time algorithm for this problem, prove that your algorithm is correct, and analyze its runtime.

**Remark 0.1.** *As noted in Kleinberg-Tardos, word-segmentation can be improved by also considering the meaning of the words and their context. For example, the string "theyouthevent" can be segmented into words in several different ways including "the you the vent", "the youth event" and "they out he vent" but "the youth event" makes much more sense than the other possibilities.*

## Problem 3: Traveling Performer (25 points)

Let's say that you are a performer and you are planning a tour. There are $n$ venues $v_1, \ldots, v_n$ that you could potentially perform at. Each venue has $v_i$ has an time interval $[a_i, b_i]$ when they would expect you to perform and a payment $p_i$ which they are willing to pay you. While you can choose your starting venue freely, it takes time $t_{ij}$ to get from venue $i$ to venue $j$ and you must arrive on time to each performance on your schedule. How can you plan a schedule which maximizes your total revenue?

We can state this problem mathematically as follows.

**Definition 0.2.** *We say that a schedule $S = (i_1, \ldots, i_k)$ is valid if for all $j \in [k-1]$, $b_{i_j} + t_{i_j i_{j+1}} \leq a_{i_{j+1}}$.*

Problem: Find a valid schedule $S = (i_1, \ldots, i_k)$ which maximizes your total revenue $P = \sum_{j=1}^{k} p_{i_j}$.

Give a polynomial time algorithm to solve this problem, prove that your algorithm is correct, and analyze its runtime.

## Problem 4: Balanced Team Building (25 points)

There are three groups of $3n$ people $A = (a_1, \ldots, a_{3n})$, $B = (b_1, \ldots, b_{3n})$, and $C = (c_1, \ldots, c_{3n})$. You are trying to build the best team you can by taking $n$ people from group $A$, $n$ people from group $B$, and $n$ people from group $C$. However, you cannot freely choose the people from each group. Instead, there are $3n$ rounds where for round $i$, you can choose either person $a_i \in A$, $b_i \in B$, or $c_i \in C$ and these people have skill levels $s(a_i)$, $s(b_i)$, and $s(c_i)$ respectively. Given this, how can you choose your team so that the total skill of all of your players is maximized?

We can state this problem mathematically as follows. Find a partition $(I_A, I_B, I_C)$ of $[3n]$ such that $|I_A| = |I_B| = |I_C| = n$ and the total skill $S = \sum_{i \in I_A} s(a_i) + \sum_{i \in I_B} s(b_i) + \sum_{i \in I_C} s(c_i)$ is maximized.

For example, if $n = 2$ and we have the following skill levels

1. $s(a_1) = 3$, $s(a_2) = 6$, $s(a_3) = 5$, $s(a_4) = 3$, $s(a_5) = 4$, and $s(a_6) = 7$.
2. $s(b_1) = 2$, $s(b_2) = 3$, $s(b_3) = 3$, $s(b_4) = 2$, $s(b_5) = 3$, and $s(b_6) = 6$.
3. $s(c_1) = 4$, $s(c_2) = 8$, $s(c_3) = 3$, $s(c_4) = 6$, $s(c_5) = 7$, and $s(c_6) = 7$.

then it is optimal to take $b_1$ in round $1$, $a_2$ in round $2$, $a_3$ in round $3$, $c_4$ in round $4$, $c_5$ in round $5$, and $b_6$ in round $6$. This corresponds to the partition $I_A = \{2, 3\}$, $I_B = \{1, 6\}$, and $I_C = \{4, 5\}$ and gives a total skill of

$$S = s(b_1) + s(a_2) + s(a_3) + s(c_4) + s(c_5) + s(b_6) = 2 + 6 + 5 + 6 + 7 + 6 = 32$$

Give a polynomial time algorithm to solve this problem, explain why it is correct (you do not need to give a rigorous proof), and analyze its runtime. Hint is available.

## Hints

4. For this problem, your subproblems will have several different parameters.

In general, when designing dynamic programs, a good question to think about is as follows. If you have made a sequence of choices, what information do you need to know about these choices in order to know what your future options are and evaluate the performance at the end?