CSE 3500 **Homework 6** Released on November 9, 2021
Due on November 16, 2021

Competitive Analysis, Greedy and Graph Algorithms

Please complete this problem set by November 16, 2021 at 11:59PM. You cannot use late days for this assignment because the solutions will be posted immediately to help prepare for the test.

Problem 0 – Birthday Party! (40%)

In the typical algorithmic setup, we have complete information as input and a desired decision or optimization as output. For example, given a weight graph, compute its maximum spanning tree. Online algorithms are distinct from this typical setup and are used to model situations of incomplete or streaming data. With incomplete information we cannot guarantee optimality, but, we can reason about an algorithm's effectiveness. This, as we learned in class, is referred to as *competitive analysis.*

The CSE3500 class is invited to Professor Derek's birthday party in a week's time at 1:00pm. You can't think of a good enough excuse not to go, so you RSVP 'yes'. During lunch time a week later, you realize that you forgot to get Derek a gift. Panicking, you walk over to a fruit bin in the dining hall where you can grab kiwis or pineapples. Pineapples are ten times as heavy as kiwis, so your bag can hold either 40 kiwis, 4 pineapples, or any combination of the two that respect the bag constraints.

You do not know if Derek enjoys pineapples or kiwis. But you do know that for each kiwi you bring, Derek will gain $K \geq 0$ joy; likewise, for each pineapple that you bring, Derek will gain $P \geq 0$ joy.

Task 1: Design an algorithm, inspired by competitive analysis, to choose the number of kiwis and pineapples to bring.

Recall that in the notes we defined the competitive ratio as $\max_{input,oracle} \frac{c_{ALG}(input)}{c_{oracle}(input)}$ where c_A is cost associated with algorithm A. In this problem, we do not have a cost that we are trying to minimize, we have joy for which we want to maximize. So, we can invert this equation and define the competitive ratio as $\max_{input,oracle} \frac{j_{oracle}(input)}{j_{ALG}(input)}$ where j_A is the joy associated with algorithm A. In other words, the competitive ratio is the ratio between the worst-case cost incurred by the optimal cost and the online strategy.

Task 2: What is the competitive ratio of your algorithm with respect to an oracle that brings the optimal presents? Your algorithm *does not* have to yield the optimal competitive ratio.

Task 3: The optimal offline strategy implemented by the oracle is a function of P and K: $Opt(P, K)$. What is $Opt(P, K)$? Show that it is optimal. *Hint: express joy as a function of*

n_K, K, n_P, and P with a constraint indicating how much you can carry in your bag.

Given your algorithm and an understanding of *Opt*, you can move on to the competitive ratio. Let n_K and n_P be the number of kiwis and pineapples chosen by the algorithm.

Claim: The algorithm $ALG = (n_K = 20, n_P = 2)$ **has a competitive ratio of** 2**.**

The competitive ratio of ALG is the maximum over a set of ratios and we wish to show that the maximum over that set is equal to a specific number. But how can you show that the competitive ratio is equal to a single number (2 here) when you cannot use equalities?

Formally, you will show that the competitive ratio is bounded above and below by 2:

$$2 \leq \frac{j_{oracle}(input)}{j_{ALG}(input)} \leq 2 \tag{1}$$

First, you should show that the competitive ratio is at least 2. The competitive ratio is defined as the maximum over a set of inputs. You can show that the maximum of a set is lower bounded by l by showing that there exists an entry within the set that is at least l. In other words, you need to find an input (n_K, n_P) such that $\frac{j_{oracle}(input)}{j_{ALG}(input)} \geq 2$.

Task 4: Find such a case and show that it gives a ratio of 2.

Finally, you must show that the competitive ratio is at most 2. The "at least" direction was easier; you only have to exhibit a single case. For the "at most" direction, you have to show that for every possible instance of $(n_K \geq 0, n_P \geq 0)$, the competitive ratio is upper bounded by 2. Consider an arbitrary instance (n_K, n_P) such that $n_K \geq 0$ and $n_P \geq 0$.

Task 5a: Show that the special case of $n_K = 0$ and $n_P = 0$ yields a joy upper bounded by 2.

Task 5b: Having established that $n_K > 0$ or $n_P > 0$, use the answer to **Task 3** and the **Claim** to show that

$$\frac{j_{oracle}(input)}{j_{ALG}(input)} \tag{2}$$

is upper bounded by 2. *Hint: Your answer to Question 3 should be expressible as two cases. Scrutinize those cases here.*

Task 5c: Write a statement concluding your results.

Problem 1 – Matryoshka dolls (30%)

Russian nesting dolls are wooden dolls of decreasing size in which the smaller dolls can be placed inside the larger dolls (Figure 1). UConn happens to have the world's largest shadow economy dedicated to the buying and selling of nesting dolls. Professor Derek sees an economic opportunity and decides to plan an elaborate Russian nesting doll heist[1].

Figure 1: Russian nesting dolls.

Let there be n nesting dolls. The i^{th} nesting doll has a total weight of w_i and total value of v_i. However, these particular dolls are unique, i.e., they are infinitely decomposable. In other words, if x_i is the normalized amount of the doll that we can take, we can take any fraction of the nesting doll $x_i \in [0, 1]$. For example, $x_i = 1$ would mean that we take the whole doll and all infinitesimally small dolls incurring a weight of w_i and adding value v_i. Whereas, if $x_i = [0, 1)$ we would take the fractional amount x_i of w_i, incur a weight of $x_i w_i$ and adding value $x_i v_i$.

Derek has a backpack with a capacity of Z, meaning that the sum of the items he takes must weigh no more than Z

$$\sum_i x_i w_i \leq Z \tag{3}$$

and he wants to select the weights for each item such that his total profits

$$\sum_i x_i v_i \tag{4}$$

are maximized.

Develop a greedy algorithm that takes in nesting dolls $x_1, \ldots, x_n$, weights $w_1, \ldots, w_n$, and values $v_1, \ldots, v_n$, and sets the x_i values to maximize $\sum_i x_i v_i$. Prove your algorithm is correct.

[1]Stealing is, in general, bad.

Problem 2 – Min-cuts (15%)

Let A and B be a partition of the nodes of $G(V, E)$ such that $s \in$A and $t \in B$. The capacity of a cut denoted $c(A, B)$, is the sum of the capacities out of A. Enumerate *all* the minimum $s - t$ cuts in the flow networks in Figures 2 and 3; give their value.

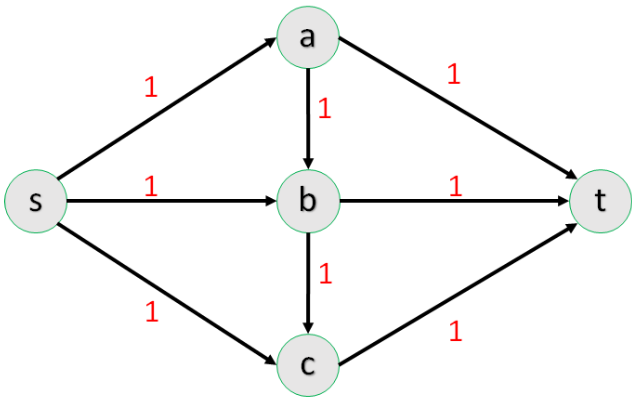

Figure 2: Flow graph 1. The capacities are labeled on the edges.

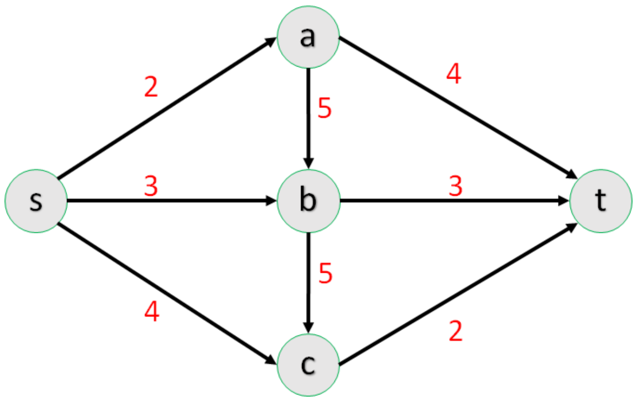

Figure 3: Flow graph 2. The capacities are labeled on the edges.

Problem 3 – Reducing problems to flow (15%)

1. Let figure 3 represent a company's transportation network from the factory (source) to the warehouse (sink), and the capacity of the roads in between intermediate cities (a, b, and c). Recently, the company has decided to expand and build more factories and warehouses. Their new flow network is shown in Figure 4. Modify Figure 4 to allow the new network to be solved with the single source, single sink Ford-Fulkerson algorithm.

2. Consider a different formulation of the hospital-resident matching problem that is arguably closer to reality. We have n hospitals and m residents. Each resident $r \in R$ selects 3 hospitals $h \in H$ that are acceptable; we assume that a hospital will accept any resident. A matching M in G is a subset of edges such that each node appears in at most one edge in M.

   Give a polynomial-time algorithm to find an M with maximum cardinality. For this problem, you need to translate the text into a mathematical formulation, so define the data structures you use to model the problem. If you make assumptions, make sure you state them. You do not need to prove the runtime but justify why it is correct.

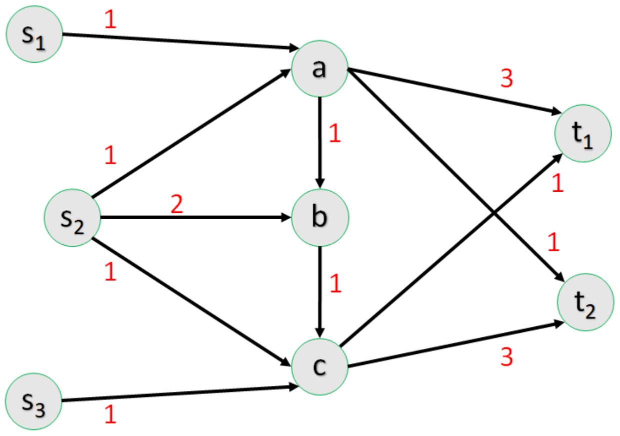

Figure 4: A company expanding its business. The capacities are labeled on the edges.