CS 170 HW 9 (Optional)

Due 2021-11-01, at 10:00 pm

1 Study Group

List the names and SIDs of the members in your study group. If you have no collaborators, you must explicitly write none.

In addition, we would like to share correct student solutions that are well-written with the class after each homework. Are you okay with your correct solutions being used for this purpose? Answer “Yes”, “Yes but anonymously”, or “No”

You may submit your solutions if you wish them to be graded, but they will be worth no points

The two problems below are dynamic programming problems and should be viewed as having 3 parts each. You should find a function \( f \) which can be computed recursively so that evaluation of \( f \) on a certain input (or combining its evaluations on a few inputs) gives the answer to the stated problem.

- Part (a) is to define \( f \) in words (without mention of how to compute it recursively). You should clearly state how many parameters \( f \) has, what those parameters represent, what \( f \) evaluated on those parameters represents, and how you should use \( f \) to get the answer to the stated problem.

- Part (b) is to give a recurrence relation showing how to compute \( f \) recursively, including a description of the base cases.

- In part (c) you should give the running time and space for solving the original problem using computation of \( f \) via memoization or bottom-up dynamic programming. If you need to use certain data structures to make computation of \( f \) faster, you should say so.

Note: if there are multiple solutions to solve the stated dynamic programming problem, you should describe the most time-efficient one you know. If there are multiple solutions with the same asymptotic time complexity, you should describe the implementation that gives the best asymptotic space complexity.

2 Equivalent Strings

We are given two strings \( A, B \) of length \( n, m \) respectively. These two strings can contain English characters a to z, as well the special character ?. We say \( A \) and \( B \) are equivalent if it is possible to replace every instance of ? with a (possibly empty) string of English characters, such that the resulting strings (containing only English letters) are the exact same.

For example, “ab?” is equivalent to “a?cd”, since with the above replacements we can transform both strings into “abcd”. Similarly, “a?bc” is equivalent to “abc”, since we are allowed to replace ? with the empty string.

Give an efficient dynamic programming algorithm to determine if two strings are equivalent. Give a three-part DP solution as defined above.
3 Geometric Knapsack

Suppose we have a piece of cloth with side lengths $X, Y$, where $X, Y$ are positive integers, and a set of $n$ products we can make out of the cloth. Each product is a rectangle of dimensions $a_i \times b_i$ and of value $c_i$, where all these numbers are positive integers.

We want to cut our large piece of cloth into multiple smaller rectangles to sell as products. Any rectangle not matching the dimensions of one of these products gets us no value. To cut the cloth, we are using a machine that takes one piece of cloth and cuts it into two, where the dividing line between the two pieces must be a vertical or horizontal line going all the way through the cloth. For example, the following is a valid series of cuts:

\[ \begin{array}{c}
\text{original} \\
\text{cut} \rightarrow \\
\text{cut} \rightarrow \\
\text{final} \\
\end{array} \]

Give an efficient algorithm that determines the maximum value of the products that can be made out of the single $X \times Y$ piece of cloth. You may produce a product multiple times, or none if you wish. Give a three-part DP solution as defined above.

4 Vertex Cover Dual

In this problem, we consider the unweighted vertex cover problem. In this problem, we are given a graph and want to find the smallest set of vertices $S$ such that every edge has at least one endpoint in $S$.

Recall the LP for this problem:

\[
\min \sum_v x_v \text{ s.t. } \forall (u,v) \in E : x_u + x_v \geq 1, \forall v \in V : x_v \geq 0
\]

In an integral solution, $x_v = 1$ if we include $v$ in our vertex cover, and $x_v = 0$ if we don’t include $v$.

(i) Write the dual of the vertex cover LP. Your dual LP should have a variable $y_e$ for every edge.

(ii) Consider the integer version of the dual you wrote, i.e. we enforce $y_e \in \{0, 1\}$. Similarly to vertex cover, we can interpret $y_e = 1$ as indicating that we include $e$ in our solution and $y_e = 0$ if we don’t include $e$.

Using this interpretation, what does the objective say? What do the constraints say? What problem is this? (You don’t have to be formal.)

(iii) True or False: If we have an integer primal solution with cost $C$ and a fractional dual solution with cost at least $C/2$, the size of the vertex cover corresponding to the primal solution is at most twice the size of the smallest vertex cover. Briefly justify your answer.
5 Permutation Games

A permutation game is a special form of zero-sum game. In a permutation game, the payoff matrix is \( n \times n \), and has the following property: Every row and column contains exactly the entries \( p_1, p_2, \ldots, p_n \) in some order. For example, the payoff matrix might look like:

\[
P = \begin{bmatrix}
p_1 & p_2 & p_3 \\
p_2 & p_3 & p_1 \\
p_3 & p_1 & p_2 \\
\end{bmatrix}
\]

Given an arbitrary permutation game, describe the row and column players’ optimal strategies, justify why these are the optimal strategies, and state the row player’s expected payoff (that is, the expected value of the entry chosen by the row and column player).

6 Bottleneck Edges

Consider the following network (the numbers are edge capacities):

(a) Find the following:
- A maximum flow \( f \), specified as a list of \( s - t \) paths and the amount of flow being pushed through each.
- A minimum cut (the set of edges with the smallest total capacity, whose removal disconnects \( S \) and \( T \)), specified as a list of edges that are part of the cut.

(b) Draw the residual graph \( G_f \) (along with its edge capacities). In this residual network, mark the vertices reachable from \( S \) and the vertices from which \( T \) is reachable.

(c) An edge of a network is called a bottleneck edge if increasing its capacity results in an increase in the maximum flow. List all bottleneck edges in the above network.

(d) Give a very simple example (containing at most four nodes) of a network which has no bottleneck edges.

(e) Give an efficient algorithm to identify all bottleneck edges in a network. (Hint: Start by running the usual network flow algorithm, and then examine the residual graph.)
7 Multiplicative Weights Intro

Multiplicative Weights
This is an online algorithm, in which you take into account the advice of \( n \) experts. Every day you get more information on how good every expert is until the last day \( T \).

Let’s first define some terminology:

- \( x_i^{(t)} \) = proportion that you ‘trust’ expert \( i \) on day \( t \)
- \( l_i^{(t)} \) = loss you would incur on day \( t \) if you invested everything into expert \( i \)
- total regret: \( R_T = \sum_{t=1}^{T} \sum_{i=1}^{n} x_i^{(t)} l_i^{(t)} - \min_{i=1,\ldots,n} \sum_{t=1}^{T} l_i^{(t)} \)

\( \forall i \in [1, n] \) and \( \forall t \in [1, T] \), the multiplicative update is as follows:

\[
 w_i^{(0)} = 1 \\
 w_i^{(t)} = w_i^{(t-1)} (1 - \epsilon)^{l_i^{(t-1)}} \\
 x_i^{(t)} = \frac{w_i^{(t)}}{\sum_{i=1}^{n} w_i^{(t)}}
\]

If \( \epsilon \in (0, 1/2] \), and \( l_i^{(t)} \in [0, 1] \), we get the following bound on total regret:

\[
 R_T \leq \epsilon T + \frac{\ln(n)}{\epsilon}
\]

Let’s play around with some of these questions. For this problem, we will be running the randomized multiplicative weights algorithm with two experts. Consider every subpart of this problem distinct from the others.

(a) Let’s say we believe the best expert will have cost 20, we run the algorithm for 100 days, and epsilon is \( \frac{1}{2} \). What is the maximum value that the total loss incurred by the algorithm can be?

(b) What value of \( \epsilon \) should we choose to minimize the total regret, given that we run the algorithm for 25 days?

(c) We run the randomized multiplicative weights algorithm with two experts. In all of the first 140 days, Expert 1 has cost 0 and Expert 2 has cost 1. If we chose \( \epsilon = 0.01 \), on the 141st day with what probability will we play Expert 1? (Hint: You can assume that \( 0.99^{70} = \frac{1}{2} \)