Homework 3: Dynamic Programming

Handed out Fri, Oct 13. **Revised due date:** Monday, Oct 23, 11:59pm (electronic submission through ELMS.)

Problem 1. Show the results of the DP algorithm for computing the longest common subsequence (LCS) as presented in class on the following input:

$$X = \langle \text{ABCACB} \rangle \qquad Y = \langle \text{BACAB} \rangle$$

(a) Show the contents of the lcs table (see, e.g., Fig. 4(a) from Lecture 11). Also, show the matrix of helper values to obtain the solution. I would suggest representing them as arrows (see, e.g., Fig. 4(b) from Lecture 11), but you may represent them as you like, as long as you explain your convention to he grader.

(b) Which sequence do you get if you apply the function get-lcs-sequence as given in class. (**Warning**: There are multiple LCS's of the same length, and the function given in class returns one of these. I will give 50% credit for any LCS, and full credit for the same one generated by the algorithm given in class. See the Extra-Credit Problem for more.)

Problem 2. In this problem we consider some variations of the longest common subsequence (LCS) problem. In all instances the input consists of two sequences, $X = \langle x_1, \ldots, x_m \rangle$ and $Y = \langle y_1, \ldots, y_n \rangle$ and the output involves the length or weight of a common subsequence (or a variant thereof).

In each of the following instances, present a recursive DP formulation. (You do *not* need to give pseudocode, just the recursive rule.) Justify the correctness of your solution.

Hint: For all DP formulations, don't forget to include (1) the basis case(s), and (2) how to obtain the final answer given your formulation.

(a) **Weighted CS:** The letters are drawn from an alphabet Σ. For each symbol $z \in \Sigma$, let $w(z)$ denote the *weight* of this symbol. The *weighted common subsequence* (WCS) is the common subsequence $Z = \langle z_1, \ldots, z_k \rangle$ that maximizes the total weight $\sum_{j=1}^{k} w(z_j)$. (For example, in Fig. 1(a), the standard LCS is $\langle$ACCBA$\rangle$, which has a weight of 22 but the WCS is $\langle$BBA$\rangle$, which has a weight of 25.)

(b) **LCS with One-Sided Repeats:** Given X and Y, we define a *common subsequence with repeats* to be a sequence $Z = \langle z_1, \ldots, z_k \rangle$ that is a subsequence of Y, and if some repeated contiguous symbols of Z are collapsed to a single symbol (e.g., "CCCC" $\rightarrow$ "C"), the result is a subsequence of X. The objective is to maximize the sum of the number of symbols of X that are matched in the LCS and the number of symbols of Y that are matched in the LCS. (For example, in Fig. 1 we show a possible input-output combination. Six symbols of X are matched and ten symbols of Y are matched, for a total of 16.)

While you do not have to implement your algorithm, for full credit it should be clear that your solution can be implemented in $O(mn)$ time.

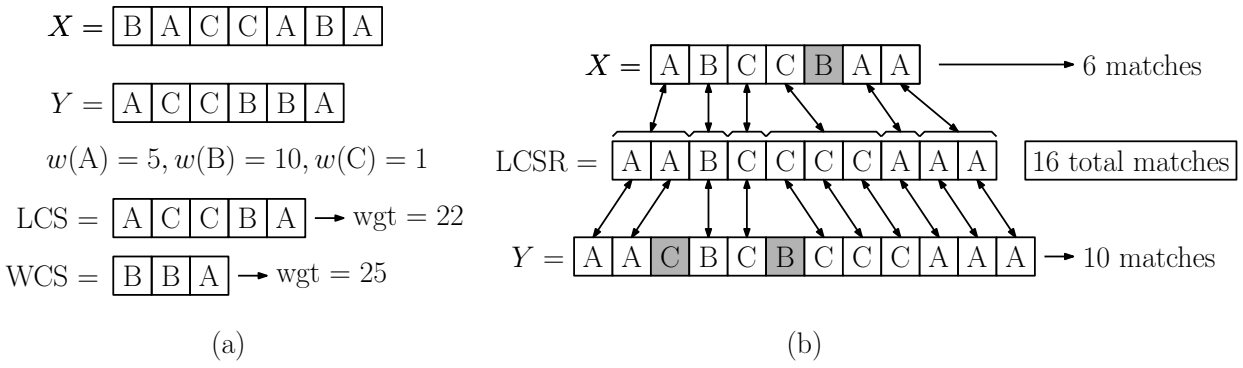

Figure 1: Problem 2(b).

Problem 3. A shipping company has a cargo ship docked in the harbor, onto which it wants to load a number of shipping containers. There are n containers to choose from. The ith container has weight of $w_i > 0$ tons and if it is shipped, it will yield a profit of $p_i > 0$ dollars. (You may assume that both w_i and p_i are integers.) The ship can hold a total weight of at most W tons.

(a) Present a recursive DP formulation for the following problem: Given $p[1..n]$ and $w[1..n]$ compute the maximum profit that can be achieved by selecting any subset of the containers, subject to the weight restriction. (You need only give the rule for computing the profit, no pseudo-code implementation and no need to compute the actual containers in the optimum solution.)

Hint: For $0 \leq i \leq n$ and $0 \leq v \leq W$, define profit(i, v) to be the maximum profit that can be achieved by selecting a subset from among containers $\{1, \ldots, i\}$ for a ship that is capable of holding v total tons. Show how to compute this array, and given the contents of this array, explain how to obtain the final answer.

(b) The shipping company adds an additional constraint that each ship has a maximum capacity of C containers. Modify your solution to part (a) to handle this constraint.

Hint: This may increase the time and space requirements from part (a) by a factor that depends on C.

(c) The shipping company has discovered that it has a second ship, identical to the first. Modify your solution to part (b) to determine the maximum profit achievable by selecting containers and assigning each to one of these two ships.

Hint: This may increase the time and space requirements of your solution to (b) by a factor depending on n, W, and C.

In all instances, justify the correctness of your solution.

Problem 4. A company has an machine for cutting sheet metal. Given a rectangular sheet of metal that is H feet high and W feet wide, this machine can cut through the entire sheet either horizontally or vertically. Each day, the company downloads a list of orders for rectangular pieces of sheet metal. This list consists of n entries, where for $1 \leq i \leq n$, $h_i \times w_i$ denotes the height and width of the piece, and p_i denotes the profit for selling a piece of this size. (The

metal has a directional surface finish, so you cannot use a 5×3 piece to fill an order for a 3×5 piece.) Orders may be filled multiple times, so if a piece of a certain size is worth \$5, the company would receive \$15 for generating three copies of this piece. Any leftover pieces that are not on the order list are wasted and cost \$1 each, irrespective of their size.

The company wants you to write a problem that determines the sequence of cuts to produce the maximum total profit. For example, in Fig. 2(a) we show an input consisting of a sheet of size 12×20. There are $n = 3$ orders as shown in the figure (the blue, green, and red rectangles). In Fig. 2(b) we show a possible solution, of total profit $(2 \cdot \$15 + 2 \cdot \$8 + 2 \cdot \$3) - 3 \cdot \$1 = \$49$.

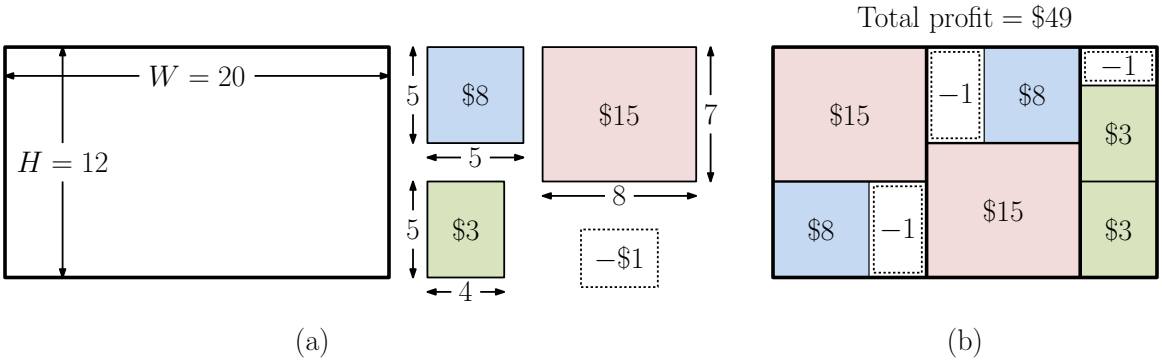

Figure 2: Problem 4.

In this problem we will develop a dynamic programming algorithm that takes as input H, W, and the arrays $h[1..n]$, $w[1..n]$, and $p[1..n]$ and determines the sequence of cuts to maximize profit. You may assume that all the input quantities are positive integers. Note that there always exists a trivial solution in which no cuts are made and the entire sheet is wasted, for a total profit of −\$1.

Present a recursive DP formulation for this problem and justify the correctness of your answer.

Hint: For $1 \leq a \leq H$ and $1 \leq b \leq W$, let profit(a, b) denote the maximum profit achievable by cutting a sheet of height a and width b.

Challenge Problem 1. Returning to Problem 1, observe that in this example there are many LCS's of equal length, such as $\langle BCAB\rangle$,$\langle ACAB\rangle$, and $\langle BACB\rangle$. How would you modify the LCS algorithm (both filling out the LCS table (lcs$[i, j]$) and the helper table ($h[i, j]$) so that it would be possible to obtain *all* the LCS strings?

How would you modify the algorithms to output the *number* of distinct LCS sequences?

Challenge Problem 2. Recall Problem 4 on cutting sheet metal.

(a) Based on your DP formulation for Problem 4, present an algorithm (in pseudo-code) that computes the maximum profit. Your answer may be either in the form of a memoized solution or a bottom-up solution. Derive your algorithm's running time. (For full credit, try to come up with as efficient an algorithm as you can.) Your answer should also include the addition of "helper" information to allow you to reconstruct the final sequence of cuts.

Hint: I would expect a running time that is a polynomial in n, H, and W.

(b) Given your answer to (a), present an algorithm that uses your helper values to output the set of cuts to obtain the optimum profit.

You can provide your answer in any reasonable way. Given the recursive nature of the problem, I think that a natural approach would be to nest the instructions, by first specifying the direction and location of the cut, then listing the cuts made to the left/lower piece, followed by the cuts made to the right/upper piece. It may be easier to list the locations of the cuts relative to the lower left corner of the current piece, as opposed to the lower left corner of the original sheet.

For example, here is a possible description of the solution of Fig. 2(b).

```
Vertical cut at x = 8 (
  Horizontal cut at y = 5 (
    Vertical cut at x = 5 (
      5 x 5 piece for $8
    ) (
      5 x 2 waste for -$1
    )
  ) (
    7 x 8 piece for $15
  )
) (
  Vertical cut at x = 8 (          [Aside: We cut off 8 above, so this is at x = 16]
    Horizontal cut at y = 7 (
        7 x 8 piece for $15
      ) (
        Vertical cut at x = 2 (    [Aside: We cut off 8 above, so this is at x = 10]
          5 x 2 waste for -$1
        ) (
          5 x 5 piece for $8
      )
    )
  ) (
    Horizontal cut at y = 5 (
      5 x 4 piece for $3
    ) (
      Horizontal cut at y = 5 (    [Aside: We cut off 5 above, so this is at y = 10]
        5 x 4 piece for $3
      ) (
        1 x 4 waste for -$1
      )
    )
  )
)
```