Exercise 6.1 (teacher demo)

Let A, B, C, D be matrices with dimensions 5×4, 4×6, 6×2 and 2×5, respectively. Using dynamic programming, determine the multiplication sequence that minimizes the number of scalar multiplications in computing $ABCD$.

Exercise 6.2 (solved in class)

The graph of Fig. 1 depicts a flexible production line. The initial product A has to go through three phases to become the final product J. The cost from each node to another is given next to the corresponding arc.

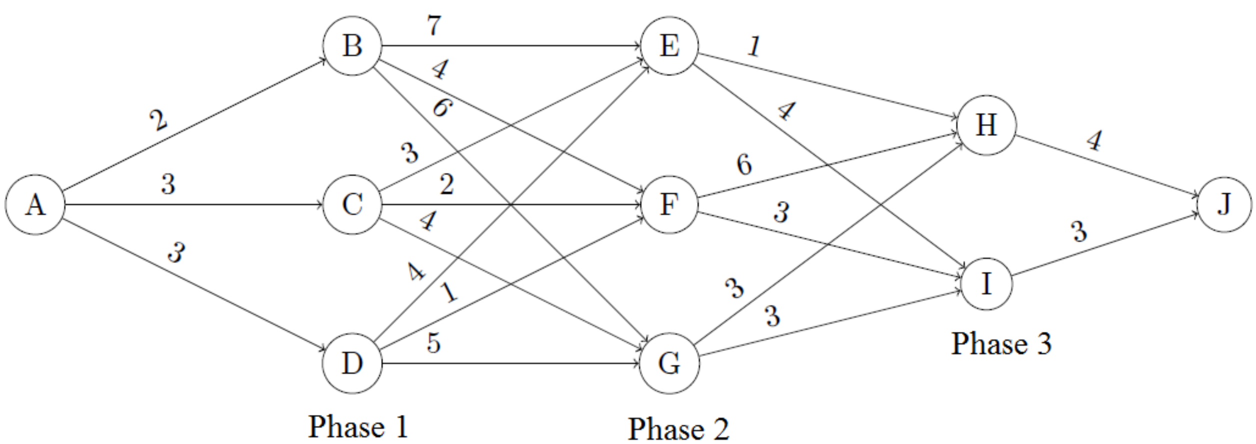

Figure 1: Graph of Exercise 1.2

Calculate the optimal cost-to-go function for each node and determine the steps that have to be taken for to minimize the cost.

Exercise 6.3 (student presents)

You want to plant tomatoes, potatoes and peas on a 180 cm wide allotment. A row of tomatoes or potatoes takes up 40 cm of space, and a row of peas 20 cm. The utility is 10 for a row of tomatoes, 7 for a row of potatoes and 3 for a row of peas. Additionally, according to an EU-directive the maximum number of allowed rows of tomatoes is two. Give a dynamic programming algorithm that could be used to solve this problem.

Exercise 6.4 (solved in class)

A certain string-processing language offers a primitive operation which splits a string into two pieces. Since this operation involves copying the original string, it takes n units of time for a string of length n, regardless of the location of the cut. Suppose, now, that you want to break a string into many pieces. The order in which the breaks are made can affect the total running time. For example, if you want to cut a 20-character string at positions 3 and 10, then making the first cut at position 3 incurs a total cost of $20 + 17 = 37$, while doing position 10 first has a better cost of $20 + 10 = 30$.

Give a dynamic programming algorithm that, given the locations of m cuts in a string of length n, finds the minimum cost of breaking the string into $m + 1$ pieces.

Exercise 6.5 (self-study)

Initially there is A amount of a resource, and it has to be used until the end of N periods. The amount $u(k)$ of the resource used on period k produces the utility $\sqrt{u(k)}$.

a) Formulate the problem, i.e., define the state equation, utility function, control constraints, and boundary conditions for the state variable. Notice that the state and control are continuous!

b) Show that $J_k(x_k) = \sqrt{(N - k)x_k}$ is the optimal cost-to-go function.

Exercise 6.6 (homework, worth 4 points)

You are the manager of a production line in a marble factory. The factory buys large blocks of marble and cuts them into smaller pieces, and sells them further to customers. The factory produces marble pieces in three different sizes (from smallest to largest): $y_1 \times x_1$, $y_2 \times x_2$ and $y_3 \times x_3$. A piece i is smaller than j if $y_i \cdot x_i < y_j \cdot x_j$. There is a utility assigned to the different pieces: u_1, u_2, u_3. The utility reflects that some of the pieces are more profitable than others.

The marble is quite fragile, so there is a certain way to cut it. Let say you have bought a block of marble of size $m \times n$. Now, if you want to cut a $y_1 \times x_1$ piece out of a $m \times n$ block, you either first have to cut a $m \times x_1$ piece and then further cut the $y_1 \times x_1$ piece out of the $m \times x_1$ block. Alternatively, you can start by cutting a $y_1 \times n$ block and then further cut the $y_1 \times x_1$ piece out of the $y_1 \times n$ block. The allowed ways to cut are depicted in the figure below. For simplicity, you do not have to consider the possibility to rotate the blocks.

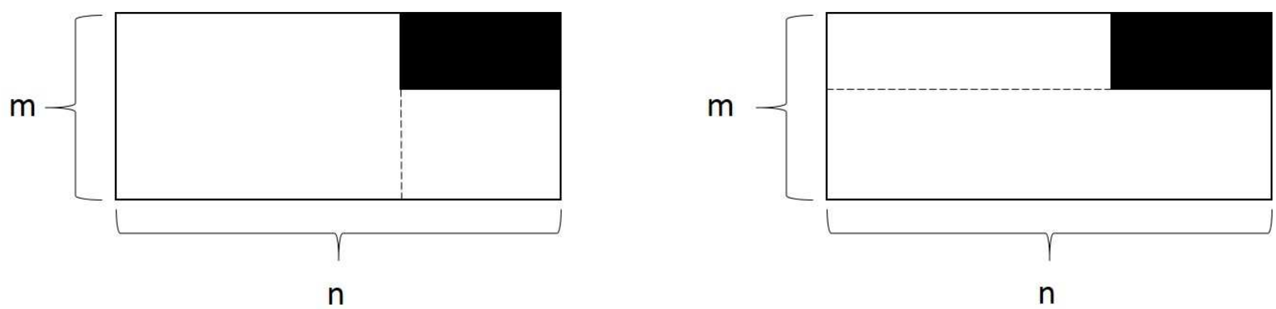

Figure 2: Permitted ways to cut out the piece that is represented by the area colored black.

Given any $m \times n$ block of marble, and any three different sized pieces $i = 1, 2, 3$, your task is to cut the smaller pieces of the block, so that the sum of their utility is maximized. You can assume that the dimensions of the pieces being cut are integer valued.

Use dynamic programming to solve the problem. In your answer, return a MATLAB code that takes as an input the size of the block, the dimensions of the pieces that can be cut, and their associated utilities, and returns the optimal utility. If your algorithm returns opt_util = 12500 for the parameters $m = n = 100$, $y = [1, 4, 3]$, $x = [2, 2, 3]$, and $u = [2, 10, 11]$, you can be fairly confident that it works correctly. This is the test case we use on your algorithm. We also test it for bugs on a couple of pathological cases, like a block, where only a certain piece can be cut.

Name the code file you submit as `student_name_dp.m`.

The MATLAB code we use to test your function is the following:

```
% Clear all variables from memory.
clear all;
clc;

% Dimensions of the large marble block.
m = 100;
n = 100;

% Dimensions of the pieces.
y = [1,4,3];
x = [2,2,3];

% Utilities of the pieces.
u = [2,10,11];

% Calling your DP-algorithm, to solve the problem of cutting the  m x n block
% into smaller pieces. Assume that y, x, and u are ordered correctly.
% As a fun competition, we will compare your solutions based on performance
% (hence the tic and toc statements).
tic;
opt_util = student_name_dp(m,n,y,x,u)
toc;
```