ECE 374 B - Fall 2024

Homework 5

- **Submit your solutions electronically on the course Gradescope site as PDF files.** If you plan to typeset your solutions, please use the LaTeX solution template on the course web site. If you must submit scanned handwritten solutions, please use a black pen on blank white paper and a high-quality scanner app (or an actual scanner, not just a phone camera). We will mark difficult to read solutions as incorrect and move on.
- **Every homework problem must be done *individually*.** Each problem needs to be submitted to Gradescope before 6AM of the due data which can be found on the course website: https://ecealgo.com/fa24/homeworks.html.
- For nearly every problem, **we have covered all the requisite knowledge required to complete a homework assignment prior to the "assigned" date.** This means that there is no reason not to begin a homework assignment as soon as it is assigned. Starting a problem the night before it is due a recipe for failure.

Policies to keep in mind

- **You may use any source at your disposal**—paper, electronic, or human—but you ***must*** cite ***every*** source that you use, and you ***must*** write everything yourself in your own words. See the academic integrity policies on the course web site for more details.
- **Being able to clearly and concisely explain your solution is a part of the grade you will receive.** Before submitting a solution ask yourself, if you were reading the solution without having seen it before, would you be able to understand it within two minutes? If not, you need to edit. Images and flow-charts are very useful for concisely explain difficult concepts.

See the course web site (https://ecealgo.com/fa24) for more information.

If you have any questions about these policies,
please don't hesitate to ask in class, in office hours, or on Piazza.

Extra Instructions Solutions to a dynamic programming problem have (at minimum) three things:

- A recurrence relation
- A *brief* description of what your recurrence function represents and what each case represents.
- A *brief* description of the memory element/storage and how it's filled in.

1. Matrix multiplication is associative, and so all parenthesizations yield the same product. A product of matrices is fully parenthesized if it is either a single matrix or the product of two fully parenthesized matrix products, surrounded by parentheses. For example, if the chain of matrices is $\langle A_1, A_2, A_3, A_4 \rangle$ then we can fully parenthesize the product in five distinct ways:

   - $\big(A_1, \big(A_2, \big(A_3, A_4\big)\big)\big)$
   - $\big(A_1, \big((A_2, A_3)A_4\big)\big)$
   - $\big((A_1A_2)\big(A_3A_4\big)\big)$
   - $\big((A_1\,(A_2A_3))A_4\big)$
   - $\big(((A_1, A_2)A_3)A_4\big)$

   How we parenthesize a chain of matrices can have a dramatic impact on the cost of evaluating the product. For example, if we have three matrices with dimensions: $A_1[10 \times 100]$, $A_2[100 \times 5]$, $A_3[5 \times 50]$, then $((A_1A_2)A_3)$ takes 7500 scalar multiplications while $(A_1\,(A_2A_3))$ takes 75000 scalar multiplications.

   Given a chain of n matrices, $A_1A_2 \ldots A_n$, where matrix A_i has dimension $r_i \times c_i$, provide a dynamic programming algorithm that finds the mininum number of scalar multiplications required to evaluate the product of the matrices.

2. Consider the problem of neatly printing a paragraph with a monospaced font (all characters having the same width) on a printer. The input text is a sequence of n words of lengths $l_1; l_2; \ldots; l_n$, measured in characters. We want to print this paragraph neatly on a number of lines that hold a maximum of M characters each. Our criterion of "neatness" is as follows. If a given line contains words i through j, where $i \leq j$, and we leave exactly one space between words, the number of extra space characters at the end of the line is $M - j + i - \Sigma_{k=i}^{j} l_k$, which must be nonnegative so that the words fit on the line. We wish to minimize the sum, over all lines except the last, of the cubes of the numbers of extra space characters at the ends of lines. Give a dynamic-programming algorithm to print a paragraph of n words neatly on a printer. Analyze the running time and space requirements of your algorithm.

3. Binomial coefficients are a family of positive integers that have a number of useful properties and they can be defined in several ways. One way to define them is as an indexed recursive function, C(n, k), where the "C" stands for "choice" or "combinations." In this case, the definition is as follows:

$$C(n,0) = 1$$
$$C(n,n) = 1$$
$$C(n,k) = C(n-1,k-1) + C(n-1,k) \qquad 0 < k < n$$

Describe a scheme for computing $C(n,k)$ using memoization.

4. Suppose we are given a collection $A = \{a_1, a_2, ..., a_n\}$ of n positive integers that add up to N. Design an O(nN)-time algorithm for determining whether there is a subset $B \in A$ such that $\Sigma_{a_i \in B} a_i = \Sigma_{a_i \in A-B} a_i$.