Foundations of Algorithms
Homework 5

Arthur Nunes-Harwitt

1. (a) Consider the following chain of six matrices: A_1, A_2, A_3, A_4, A_5, and A_6, where A_1 is 5×10, A_2 is 10×3, A_3 is 3×12, A_4 is 12×5, A_5 is 5×50, and A_6 is 50×6. Find an optimal parenthesization of this matrix-chain. Show both the table containing the optimal number of scalar operations for all slices and the choice table.

   (b) Prove using the strong form of induction that for any $n \in \mathbb{N}$, if $n \geq 1$ then a full parenthesization of an n-element expression has $n - 1$ pairs of parentheses.

2. (a) CLRS 15.3-1

   (b) Draw the recursion tree for the merge-sort algorithm on an input sequence of length 16. Explain why memoization fails to speed up a good divide-and-conquer algorithm like merge-sort.

   (c) Consider a variant of the matrix-chain multiplication problem in which the goal is to parenthesize the sequence of matrices so as to *maximize* the number of scalar multiplications. Does this problem exhibit optimal substructure?

3. Consider the problem of neatly printing a paragraph on the screen (or on a printer). The input text is a sequence S of n words (represented as strings) of lengths $\ell_1, \ldots, \ell_n$ (measured in characters). The input bound M is the maximum number of characters a line can hold. The key to neatly printing a paragraph is to identify in the text sequence the lines of the paragraph so that new-lines can be placed at the end of each line. We can formalize the notion of the "badness" of a line as the number of extra space characters at the end of the line or ∞ if the bound M is exceeded. We can formalize the notion of the "badness" of a paragraph as the badness of the worst (i.e., maximum) line of the paragraph not including the last line. Thus to identify the lines for a neat paragraph, we seek to minimize the badness of the paragraph.

   (a) If a given line contains words i through j, and we leave exactly one space between words, the number of extra space characters at the end of the line is $M - j + i - \sum_{k=i}^{j} \ell_k$. Write a mathematical function $s(S, M, i, j)$ that computes the number of extra space characters at the end of a line.

   (b) (**project**) Write a procedure `extraSpace` that implements s.

   (c) Use the function s to write a mathematical function $bl(S, M, i, j)$ that computes line badness.

   (d) (**project**) Write a procedure `badnessLine` that implements bl.

   (e) Write a recursive mathematical function $mb(S, M)$ that computes the minimum paragraph badness (using slicing).

   (f) Write a recursive mathematical function $mb'(S, M, i)$ where $mb'(S, M, i) = mb(S[i:], M)$.

   (g) (**project**) Write a recursive procedure `minBad` that implements the function mb'. It should take three parameters: S, M, and i a slicing index.

(h) (**project**) Write a procedure `minBadDynamic` that implements the function mb' using dynamic programming. It should take only two parameters: S, and M.

(i) (**project**) Write a procedure `minBadDynamicChoice` that implements the function mb' using dynamic programming. In addition to returning $mb(S, M)$, it should also return the choices made. Then write a procedure `printParagraph` which takes two parameters: S and M that displays the words in S on the screen using the choices of `minBadDynamicChoice`. What is the asymptotic running time of your algorithm?

4. CLRS 24.1-1