Advanced Analysis of Algorithms - Homework III

K. Subramani
LCSEE,
West Virginia University,
Morgantown, WV
{ksmani@csee.wvu.edu}

1 Instructions

1. The homework is due on November 5, in class.
2. Each question is worth 4 points.
3. Attempt as many problems as you can. You will be given partial credit, as per the policy discussed in class.
4. The work must be entirely your own. You are expressly **prohibited** from consulting with colleagues or the internet (with the exception of the material on the course website, your class notes and [NN09]).

2 Problems

1. Let $\mathbf{A}[1 \cdots n]$ denote an unsorted array of n distinct numbers. An inversion pair of $\mathbf{A}$, is a pair of indices (i, j), such that $i < j$ and $A[i] > A[j]$. Devise a divide-and-conquer algorithm to compute the number of inversion pairs in $\mathbf{A}$.
2. In class, we repeatedly exploited the fact that all edge costs were non-negative in arguing the correctness of the Floyd-Warshall algorithm for the shortest paths problem. Now assume that the edge costs can be negative.
   (a) Identify the conditions under which the optimal substructure property **does not** hold.
   (b) Devise a dynamic programming based algorithm for the All-Pairs shortest path problem in the presence of negative weights. Your algorithm should either produce the All-Pairs shortest paths or indicate that the input instance does not have the optimal substructure property.
3. (a) Compute the product of the two matrices below, using Strassen's matrix multiplication algorithm.
   $$\mathbf{X} = \begin{pmatrix} 9 & 3 \\ 2 & -1 \end{pmatrix}, \quad \mathbf{Y} = \begin{pmatrix} 1 & 2 \\ 2 & -1 \end{pmatrix}$$
   (b) Compute the optimal parenthesization of the following matrix chain: $\langle A_{10\times 15} \cdot B_{15\times 9} \cdot C_{9\times 7} \cdot D_{7\times 10}\rangle$.
   In both problems, you are required to show all the intermediate steps (and tables, if necessary).
4. Suppose that we are given a directed acyclic graph $\mathbf{G} = \langle V, E\rangle$ with real-valued edge weights and two distinguished vertices s and t. Describe a dynamic programming approach for finding a longest weighted simple path from s to t. Establish the correctness of your algorithm and give an asymptotic bound on its running time.
5. (a) In class, we discussed the notion of binary search trees. Argue that the number of distinct binary search trees on n nodes is $\frac{1}{n+1}\binom{2n}{n}$.

(b) In the Optimal Binary Search Tree (OPST) problem, we are required to organize n keys, $key_1 \leq key_2 \leq \ldots key_n$, in a binary search tree so that the expected search time is minimized. Consider the dynamic program algorithm developed in class for the OPST problem.

Let $A[i,j]$ denote the expected search time of the optimal binary search tree on keys key_i through key_j. As discussed in class,

$$\begin{aligned} A[i,j] &= \min_{i \leq k \leq j} A[i,k] + A[k+1,j] + \sum_{m=i}^{j} p_m, \text{ if } i < j \\ A[i,i] &= p_i \\ A[i,i-1] &= 0 \\ A[j,j+1] &= 0 \end{aligned} \tag{1}$$

We can record the actual value of k that created the optimal split in System (1), by using an auxiliary array. Define $root[i,j]$ to be value of k that minimizes $A[i,j]$. Argue that there are always roots of optimal subtrees such that

$$root[i,j-1] \leq root[i,j] \leq root[i+1,j], \text{ for all } 1 \leq i \leq j \leq n$$

Use the above proof to design an algorithm for the OPST problem that runs in $O(n^2)$ time. Note that the algorithm discussed in class runs in $\Theta(n^4)$ time.

References

[NN09] Richard Neapolitan and Kumarss Naimipour. *Foundations of Algorithms Using C++ Pseudocode*. Jones and Bartlett, 2009.