Algorithms Core 15-750 PROBLEM SET # 1 Spring 1998

DUE: 10:30AM Monday 2 February 1998

1 Crossover point for Strassen's Algorithm

In class we presented Strassen's algorithm for multiplying n by n matrices in $O(n^{\log 7})$ time. Assume that all the work is in the arithmetic operations.

a) Compute the smallest k such that Strassen does fewer arithmetic operations than the naive matrix multiplications on matrices of size 2^k by 2^k.

b) Compute the smallest even integer n such that one level of Strassen's algorithm followed by naive matrix multiplication does fewer arithmetic operations than the naive matrix multiplications on matrices of size n by n. (The algorithm is: take two n by n matrices, split each into four $n/2$ by $n/2$ matrices, organize the multiplications of the blocks according to Strassen. For each of the smaller products do the computation in the naive, direct way, without calling Strassen recursively.)

c) Compute the smallest integer n divisible by 4 such that two levels of Strassen followed by naive matrix multiplication does fewer arithmetic operations than one level of Strassen followed by naive matrix multiplication on matrices of size n by n.

2 Space for Strassen's Algorithm

In class we presented Strassen's algorithm for multiplying n by n matrices in $O(n^{\log 7})$ time. Show how to organize his algorithm so that it will use at most $2n^2$ space for the input, n^2 space for the output, and at most n^2 extra work space.

In general this problem can be messy because it needs space for the recursion stack as well as the space for the pointers to maintain the matrices as quad-trees. You can assume that the only space needed is to store scalars. Further assume that you may not overwrite the input, but you may use the output space for scratch space.

3 LP Problem

Show how the problem of finding a bridge in 2D can be reduced to solving a linear programming problem with two degrees of freedom.

4 Backwards Analysis of a Random Binary Search Tree

In this problem you will analyze the expected cost of constructing a random binary search tree. Assume that the keys K to be inserted into a binary search tree are the integers from 1 to n.

In the forward direction we extract a random key from K and insert it into the binary search tree, see CLR Chap 13.3. In CRL or AHU they show that the expected cost is $O(n \log n)$.

The insertion cost can be computed directly from the tree.

Let NodeWeight(v) be the number of nodes in the subtree rooted at v. Then the total insertion cost for the binary search tree T is $InsertionCost(T) = \sum_{v \in T} NodeWeight(v)$. We will use backwards analysis to estimate the insertion cost for a random binary search tree.

In this exercise we shall analyze the cost by running the algorithm "backwards". That is we pick the elements from K last one first. We start by picking the last element first by picking a random element from K and choosing it to be the last element insert and thus a leaf of the tree. Of the remaining elements in K we next pick the second to last element. If we stop this process at any given time we will have keys which have not been picked and those keys which are now already in one of the subtrees formed. You will get a lot out of a small example

Hints:

- Show how to determine the weight of a node if it is the next element picked.
- What is the average or expected weight of the remaining key.
- Use this estimate to finish the analysis.

5 Extra Credit

You may use Winograd's version of Strassen for problems 1 and 2. When using this version find a solution that uses $4/3n^2$ extra space.

Winograd's version of Strassen's algorithm:

$$
\begin{array}{lll}
s_1 = a_{21} + a_{22} & m_1 = s_2 s_6 & t_1 = m_1 + m_2 \\
s_2 = s_1 - a_{11} & m_2 = a_{11} b_{11} & t_2 = t_1 + m_4 \\
s_3 = a_{11} - a_{21} & m_3 = a_{12} b_{21} & \\
s_4 = a_{12} - s_2 & m_4 = s_3 s_7 & \\
s_5 = b_{12} - b_{11} & m_5 = s_1 s_5 & \\
s_6 = b_{22} - s_5 & m_6 = s_4 b_{22} & \\
s_7 = b_{22} - b_{12} & m_7 = a_{22} s_8 & \\
s_8 = s_6 - b_{21} & &
\end{array}
$$

The elements of the product matrix are:

$$
\begin{aligned}
c_{11} &= m_2 + m_3 \\
c_{12} &= t_1 + m_5 + m_6 \\
c_{21} &= t_1 - m_7 \\
c_{22} &= t_2 + m_5
\end{aligned}
$$