Technische Universität München
Department of Computer Science
Chair for Efficient Algorithms
Prof. Dr. Sami Khuri

WS 1999/2000
Problem Set 12
February 1, 2000

Efficient Algorithms and Data Structures

Problem 1 (12 credits)

Suppose we build a "random" binary search tree on nodes $1, \ldots, n$, in the following manner. Generate a random permutation π of $\{1, \ldots, n\}$, and build the desired binary search tree by means of successively inserting $\pi(1), \ldots, \pi(n)$. (To perform an insertion, first search for the specified value, and then insert the new node as the "correct" son (left or right) of the leaf node where the search terminated unsuccessfully.) Call this tree $T(\pi)$.

a) For $n = 4$, write out all possible trees $T(\pi)$ (note that they are not all distinct).

b) Given a binary search tree T on n nodes, we define $C(T)$ to be the average number of nodes examined in searching for a node in the tree T. Show that

$$C(T) = \frac{1}{n} \cdot \sum_{v \in T} \text{depth}(v),$$

where the sum is taken over all nodes v in the tree T.

c) For all 24 trees in a), calculate $C(T)$. What is the average value of $C(T)$?

d) Define $D(T) = n \cdot C(T)$. Then, define $C(n)$ (resp. $D(n)$) to be the average value of $C(T)$ (resp. $D(T)$) over all $n!$ trees $T(\pi)$, where π is a permutation of $\{1, \ldots, n\}$. Show that $D(n)$ satisfies the recurrence:

$$\begin{aligned} D(n) &= \frac{1}{n} \cdot \sum_{i=1}^{n} (D(i-1) + D(n-i) + n), \text{ if } n \geq 2 \\ D(1) &= 1. \end{aligned}$$

e) Solve the above recurrence, obtaining the solution for $D(n)$ in terms of the Harmonic number H_n.

f) What is the growth rate of $C(n)$?

Problem 2 (7 credits)

Given keys a, b, c, d, e, f (in this order) having frequencies $10, 3, 4, 7, 15, 4$ (respectively), use dynamic programming to find the optimal binary search tree. Draw the resulting tree. What is the average access time for this tree?

Problem 3 (7 credits)

Solve the following instance of the $0 - 1$ knapsack problem using dynamic programming. Also, find the contents of the optimal knapsack from the table. The weights are 1, 2, 3, 5, 6, and 8, the profits are 3, 6, 7, 9, 11, and 18 (resp.), and the knapsack capacity is 15.

Problem 4 (7 credits)

Use Floyd's algorithm to solve the all-pairs shortest path problem for the instance given by the following matrix of edge costs:

$$\begin{bmatrix} 0 & 3 & 5 & 7 & 4 \\ 3 & 0 & 1 & 6 & 8 \\ 5 & 1 & 0 & 6 & 3 \\ 7 & 6 & 6 & 0 & 2 \\ 4 & 8 & 3 & 2 & 0 \end{bmatrix}$$

Also, write out all 10 shortest paths.

Problem 5 (7 credits)

Solve the following instance of TSP using backtracking with branch-and-bound:

$$\begin{bmatrix} \infty & 4 & 6 & 8 & 7 & 3 \\ 4 & \infty & 5 & 3 & 6 & 5 \\ 6 & 5 & \infty & 1 & 5 & 4 \\ 8 & 3 & 1 & \infty & 2 & 7 \\ 7 & 6 & 5 & 2 & \infty & 3 \\ 3 & 5 & 4 & 7 & 3 & \infty \end{bmatrix}$$