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

WS 1999/2000
Problem Set 11
January 25, 2000

Efficient Algorithms and Data Structures

Problem 1 (5+5 credits)

Consider strings $S = \texttt{acbcdb}$ and $T = \texttt{cadbd}$ over $\Sigma = \{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}\}$ and the alignment function defined in class, i.e.,

$$\begin{aligned} v(S[i], T[j]) &= \begin{cases} +1 & \text{if } S[i] = T[j] \\ -1 & \text{if } S[i] \neq T[j] \end{cases} \\ \text{and } v(S[i], -) &= v(-, T[j]) = -2 \ . \end{aligned}$$

a) Using the dynamic programming algorithm discussed in class, find the value of an optimal alignment of S and T and give all optimal alignments.

b) Use Hirschberg's OPTA procedure to find an optimal alignment of S and T. Show your work by giving a snapshot of the state of the problem after each recursive call of OPTA.

Problem 2 (6+4 credits)

Consider the problem of computing a longest increasing subsequence (LIS) of a sequence of n numbers. The elements of the subsequence need not be adjacent in the sequence. For example, $(1, 3, 4)$ and $(1, 2, 4)$ are longest increasing subsequences of the sequence $(1, 5, 3, 2, 4)$.

a) Give an $O(n^2)$ algorithm to find an LIS using dynamic programming.

b) Give an algorithm that computes an LIS in time $O(n \log n)$.

Problem 3 (10 credits)

Consider the problem of making change for n cents using the least number of coins (cf. Problem 6 of Problem Set 5). Give a dynamic programming algorithm that yields an optimal solution for an *arbitrary* set $\{d_1, d_2, \ldots, d_k\}$ of coin values. Analyze the running-time of your algorithm!

Problem 4 (10 credits)

Instead of using powers of two as in the binary representation of integers, we represent integers as the sum of Fibonacci numbers. In other words, instead of an array of bits, we keep an array of "fits," where the i^{th} least significant fit indicates whether the sum includes the i^{th} Fibonacci number F_i. For example, the fit string 101110 represents the number $F_6 + F_4 + F_3 + F_2 = 8 + 3 + 2 + 1 = 14$. Describe algorithms to increment and decrement a fit string in constant amortized time.
Hint: Most numbers can be represented by more than one fit string.