University of Freiburg
Dept. of Computer Science
Prof. Dr. F. Kuhn
P. Bamberger, P. Schneider

Algorithm Theory
Exercise Sheet 3

Due: Tuesday, 9th of November, 2021, 4 pm

Exercise 1: Scheduling *(8 Points)*

Given n jobs of lengths $t_1 \dots, t_n$ with *one deadline* $d \geq 0$, we want to schedule these jobs such that the *average lateness* is minimized. That is, for each job i we want to find a start time and finishing time $0 \leq s(i) \leq f(i)$ with $f(i) - s(i) = t_i$ such that the intervals $[s(i), f(i)]$ are pairwise non-overlapping (overlapping start- and endpoints are allowed) and the average over all $L(i) = \max\{0, f(i) - d\}$ is minimal.

(a) Describe a greedy algorithm for this problem. *(3 Points)*

(b) Prove that it computes an optimal solution. *(5 Points)*

Exercise 2: Prefix Codes *(12 Points)*

Imagine you have n characters $c_1, \dots, c_n$ and each has a frequency $f_1, \dots, f_n$ (w.l.o.g. sorted ascending) with which it occurs in a text. The goal is to compute a code over $\{0, 1\}$ for each character (i.e., assign a unique bit sequence to each character) which is prefix-free, i.e., no codeword is a prefix of another.

Such a *prefix code* can be obtained by constructing a full binary tree[1]: Use the characters $c_1, \dots, c_n$ as leaves, assign 0 and 1 to all edges, such that internal nodes have a child with a 0-edge *and* a child with a 1-edge. The code of c_i is then given by the bits on the path from the root to the leaf c_i.

The goal is to minimize the total length of a message with the given frequency of symbols, i.e. $\sum_{i=1}^{n} f_i \cdot \ell_i$, where ℓ_i is the length of the codeword of c_i. Analogously, we want to find a full binary tree that minimizes $\sum_{i=1}^{n} f_i \cdot d_i$, where d_i is the (unweighted) length of the path from root to c_i (depth).

Such a tree can be constructed with a greedy method: Start with $c_1, \dots, c_n$ as leaves (w.l.o.g. sorted by frequency). Add an internal node and make the two least frequent characters c_1, c_2 its children (break ties arbitrarily). The internal node becomes a new character c_{n+1} with frequency $f_{n+1} = f_1 + f_2$. Then "remove" the leaves c_1, c_2 and recurse on the characters $c_3, \dots, c_{n+1}$ (i.e., treat c_{n+1} as new leaf). We call the resulting tree the *greedy tree* and the resulting prefix-code for $c_1, \dots, c_n$ the *greedy code*.

(a) Construct the greedy tree and greedy code for $n = 6$ characters with frequency $f_i = i$. *(3 Points)*

Remark: for more consistent solutions, assign 0 the left-child edges and 1 to right-child edges.

(b) Show that there is an *optimal* full binary tree T with leaves $c_1, \dots, c_n$ (i.e., that minimizes $\sum_{i=1}^{n} f_i \cdot d_i$), in which the two least frequent elements c_1, c_2 are siblings. *(5 Points)*

Hint: Show that for two siblings c_j, c_k which are at largest depth in some full binary tree it does not make $\sum_{i=1}^{n} f_i \cdot d_i$ larger if we swap c_j with c_1 and c_k with c_2.

(c) Give an inductive argument that the greedy code is optimal. *(4 Points)*

[1] In a full binary tree each node has 0 or 2 children.