Homework set 1

Note: the homework sets are not for submission. They are designed to help you prepare for the quizzes.

1. Consider the following variation of the Interval Scheduling Problem. You have a processor that can operate 24 hours a day, every day. People submit requests to run daily jobs on the processor. Each job $j$ comes with a start time $s_j$ and an end time $t_j$. If the job is accepted, then it must run during the time interval $[s_j, t_j)$ every day. Notice however that it is possible that $s_j$ occurs before midnight, and $t_j$ after midnight. Given the input set $J$ of jobs, the goal is to accept as many jobs as possible, subject to the constraint that a processor can run at most one job at a time. Give an efficient algorithm to solve this problem and analyze its running time.

2. We are given a set $J$ of $n$ jobs. The execution of each job consists of two phases. First, it must be pre-processed on a supercomputer, and then it must be finished on a regular computer. Each job $j$ is associated with parameters $p_1(j)$ - the processing time it requires on the supercomputer, and $p_2(j)$ - the processing time it requires on a regular computer. We only have one supercomputer, which can only execute one job at a time. However, we have an unbounded number of regular computers, that can execute any number of jobs simultaneously. Given a schedule $S$, the finish time $C(S)$ of the schedule is the earliest time by which all the jobs have been completed. Our goal is to find a schedule $S$ of all jobs, with minimum finish time $C(S)$. Design an efficient algorithm, prove its correctness, and analyze the running time.

3. We are given a set of $n$ points $X = \{x_1, \ldots, x_n\}$ on the real line. Design an algorithm that finds a minimum-cardinality set of unit-length intervals, that cover all points in $X$. Prove the algorithm’s correctness and analyze its running time.

4. In this problem we consider an extension of the activity selection problem to multiple machines. Suppose we are given a set $J$ of $n$ jobs, where each job $j \in J$ is associated with an interval $[s_j, f_j)$ of time during which it needs to be executed. We are also given $k$ machines $M_1, \ldots, M_k$, each of which can execute at most one job at any given time. That is, if two jobs $j, j'$ are assigned to the same machine $M_i$, then their intervals $[s_j, f_j), [s_{j'}, f_{j'})$ cannot overlap.

   (a) Given any collection $I$ of intervals, and any point $t$, we say that the depth of $t$ is the total number of intervals in $I$ containing $t$, $d(t) = |\{I \in I \mid t \in I\}|$. The depth of the set $I$ of intervals is defined to be the maximum depth of any point $t$. Prove that the set $J$ of jobs can be scheduled on $k$ machines iff the depth of the set of intervals associated with the jobs in $J$ is at most $k$. Give an efficient algorithm for computing the schedule, prove its correctness and analyze its running time.

   (b) Assume now that no two jobs in $J$ have the same finish time $f_j$, and assume that we are given two machines, that is, $k = 2$ (but we do not have any guarantee on the depth of the set of intervals associated with the jobs in $J$). Design an efficient algorithm that schedules the maximum possible number of jobs in $J$ on two machines. Prove its correctness and analyze its running time.

5. Suppose we have an alphabet with $2^k$ characters, and a string in which all characters are almost equally common. That is, for all $x, y \in \Sigma$, $f(x) \leq f(y) < 2f(x)$. How will the Huffman tree look like? What is its cost? Prove your answer.