

Homework Assignment #1

Due: September 28, 2010, by 10:30 am

(in the drop box for your CSCC73 tutorial section, near room SW-626A)

*Appended to this document is a cover page for your assignment. Fill it out, staple your answers to it, and deposit the resulting document into the course drop box (without putting it in an envelope).*

**Question 1.** (10 marks) Consider a variation of the problem of minimising lateness, discussed in KT §4.2: Instead of minimising the *maximum lateness over all jobs*, we wish to minimise the *sum of the latenesses of all jobs*. (Recall that the lateness of job  $j$  in a schedule  $s$  is defined as  $\max(0, f_s(j) - d(j))$ , where  $f_s(j)$  is the finish time of  $j$  in  $s$ , and  $d(j)$  is the deadline of  $j$ .)

Does the algorithm described in KT §4.2 solve this variation of the problem? If so, prove that this is the case; if not, provide a counterexample.

**Question 2.** (10 marks) We are given an interval  $I = [S, F]$  and a set of intervals  $\{I_1, I_2, \dots, I_n\}$  that covers  $I$ ; i.e.,  $I \subseteq \cup_{i=1}^n I_i$ . We wish to find a subset of  $\{I_1, I_2, \dots, I_n\}$  that covers  $I$  and has as few intervals as possible.

(Think of  $I$  as the period during which a building must be guarded, and  $I_1, I_2, \dots, I_n$  as the intervals in which each of  $n$  guards can be on duty. We want to find the smallest number of guards so that at all times during  $I$  at least one guard is on duty.)

**a.** (3 marks) Consider the following greedy strategy. Assume that  $I_1, I_2, \dots, I_n$  are sorted in decreasing order of the length of their intersection with  $I$ . (Rename them, if needed.) Start with the empty subset, and consider each interval in turn, keeping the interval under consideration in the subset if and only if it covers a part of  $I$  that isn't already covered by intervals kept so far. Give a counterexample showing that this strategy does not necessarily yield a desired subset of  $I_1, I_2, \dots, I_n$ .

**b.** (7 marks) Describe an efficient greedy algorithm that finds a desired subset of  $I_1, I_2, \dots, I_n$ . Prove that your algorithm is correct, and analyse its running time.

**Question 3.** (10 marks) The optimal length of a paddle for a canoeist is his/her shoulder height. We have  $n$  canoeists with shoulder heights  $h_1, h_2, \dots, h_n$ , and  $n$  paddles of length  $\ell_1, \ell_2, \dots, \ell_n$ . We must assign exactly one paddle to each canoeist (and, of course, different paddles to different canoeists). Such an assignment can be conveniently represented by a permutation  $\pi$  of  $1, 2, \dots, n$ : canoeist  $i$  is assigned paddle  $\pi(i)$ . The penalty of such an assignment for canoeist  $i$  is  $|h_i - \ell_{\pi(i)}|$  — i.e., how far off the optimal is the length of the paddle assigned to the canoeist. The average penalty of such an assignment is  $\frac{1}{n} \sum_1^n |h_i - \ell_{\pi(i)}|$ . We wish to find an assignment that minimises the average penalty.

For each of the following two greedy strategies, either show that it determines a desired assignment or provide a counterexample:

**a.** Find a pair  $(i, j)$  that minimizes  $|h_i - \ell_j|$  (i.e., a canoeist-paddle pair of minimum penalty). Assign paddle  $j$  to canoeist  $i$  (i.e.,  $\pi(i) = j$ ) and apply the same strategy to the remaining canoeists and paddles, until all canoeists are assigned paddles.

**b.** Sort the canoeists in increasing shoulder height and the paddles in increasing length, and assign to each canoeist the corresponding paddle. In other words, the canoeist of minimum shoulder height gets the shortest paddle, the canoeist of second minimum shoulder height gets the second shortest paddle, and so on.

**Question 4.** (10 marks) Let  $G$  be a connected, undirected graph, where each edge has a cost  $c(e)$ . Prove that if no two edges have the same cost,  $G$  has a unique MST.

**Hint.** Let  $e_1, e_2, \dots, e_{n-1}$  and  $f_1, f_2, \dots, f_{n-1}$  be MSTs of  $G$ , where  $c(e_1) < c(e_2) < \dots < c(e_{n-1})$  and  $c(f_1) < c(f_2) < \dots < c(f_{n-1})$  — i.e., the two lists of edges are sorted in increasing cost. Prove that, for each  $i$ ,  $e_i = f_i$ .

**Question 5.** (10 marks) Consider Huffman's algorithm.

**a.** (2 marks) Give an example of a (small) set of symbols and their associated frequencies so that the maximum frequency of any symbol is equal to  $2/5$ , and Huffman's algorithm *could* produce a tree in which no codeword has length 1. Show such a tree that could be produced by Huffman's algorithm in your example.

**b.** (8 marks) Prove that for any set of symbols, if some symbol has frequency (strictly) greater than  $2/5$ , Huffman's algorithm will necessarily produce a codeword of length 1.

# Cover page for CSCC73 Assignment #1

Submitted by

(1) Family Name: \_\_\_\_\_

(2) Family Name: \_\_\_\_\_

Given Name: \_\_\_\_\_

Given Name: \_\_\_\_\_

Student Number: \_\_\_\_\_

Student Number: \_\_\_\_\_

*By virtue of submitting this homework I/we acknowledge that I am/we are aware of the policy on homework collaboration for this course.*