

CS 473: Algorithms, Fall 2008

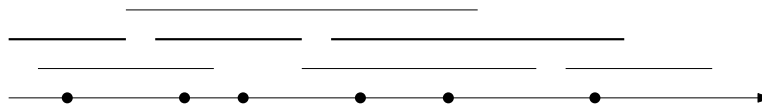
HW 6 (due Tuesday, October 21, 11am)

This homework contains three problems. **Read the instruction for submitting homework on the course webpage.** In particular, *make sure* that you write the solutions for the problems on separate sheets of paper and then staple them together. Write your name and netid on each sheet.

Collaboration Policy: For this homework you are allowed to work in groups of up to 3 students each. Starting this week, a third of the on-campus students will be presenting their homework orally. Please see the newgroup for instructions on which groups will be presenting orally and the instructions for signing up for a slot. The other groups will submit a written homework.

For each of the problems, state the *space* and running time of your algorithm.

- (30 pts) Problem 6.5 from the text book. Read Section 6.3 to help with this problem.
- (35 pts) You are given n points p_1, p_2, \dots, p_n on the real line. The location of p_i is given by its coordinate x_i . You are also given m intervals I_1, I_2, \dots, I_m where $I_j = [a_j, b_j]$ (a_j is the left end point and b_j is the right end point). Each interval j has a non-negative weight w_j . An interval I_j is said to *cover* p_i if $x_i \in [a_j, b_j]$. A subset $S \subseteq \{I_1, I_2, \dots, I_m\}$ of intervals is a *cover* for the given points if for each p_i , $1 \leq i \leq n$, there is some interval in S that covers p_i . In the figure below, the intervals shown in bold form a cover of the points.



The goal is to find a minimum weight cover of the points. Note that a minimum weight cover may differ from a cover with minimum number of intervals.

- Give an example to demonstrate that the greedy algorithm that iteratively picks the interval with minimum ratio of weight to number of remaining uncovered points covered (this changes as algorithm picks intervals) does not yield an optimum solution.
 - Describe an algorithm that computes an optimum solution.
- (35 pts) A sequence is *palindromic* if it is the same whether read left to right or right to left. An example is $m, a, l, a, y, a, l, a, m$ (*Malyalam* is a Southern Indian language). Given a sequence a_1, a_2, \dots, a_n give an $O(n^2)$ algorithm to compute a *longest palindromic subsequence* of the given sequence. For example, the sequence below

$A, C, G, T, G, T, C, A, A, A, A, T, C, G$

has many palindromic subsequences, include A, C, G, C, A and A, A, A, A .