University of Washington
Department of Computer Science and Engineering
CSE 417, Autumn 2020

October 21, 2020

Homework 4, Due Wednesday, October 28, 1:30 pm, 2020

Turn in instructions: Electronics submission on GradeScope. Submit as a PDF, with each problem on a separate page.

Problem 1 (10 points):

Let S be a set of intervals, where $S = \{I_1, \ldots, I_n\}$ with $I_j = (s_j, f_j)$ and $s_j < f_j$. A set of points $P = \{p_1, \ldots, p_k\}$ is said to be a *cover* for S if every interval of S includes at least one point of P, or more formally: for every I_i in S, there is a p_j in P with $s_i \leq p_j \leq f_i$.

Describe an algorithm that finds a cover for S that is as small as possible. Argue that your algorithm finds a minimum size cover. You algorithm should be efficient. In this case $O(n \log n)$ is achievable but it is okay if your algorithm is $O(n^2)$. You may assume that the intervals are sorted in order of finishing time.

Problem 2 (10 points):

The paragraphing problem is: Given a set of words $w_1, \ldots, w_n$ with word lengths $l_1, \ldots, l_n$, break the words into consecutive groups, such that the sum of the lengths of the words in each group is less than a fixed value K. (We will ignore the issue of putting spaces between words or hyphenation; these are minor details.) The words remain in the original order, so the task is just to insert line breaks to ensure that each line is less than length K.

The greedy algorithm for line breaking is to pack in as many words as possible into each line, e.g., to put words into a line one at a time until the length bound K is reached, and break the line before the word w_r that caused the the bound to be exceeded.

Prove that the Greedy Algorithm is optimal in the sense that it produces a paragraph with the smallest number of lines. For a formal proof, induction is recommended. The key to this problem is coming up with the right induction hypothesis.

Hint: The key to proving the algorithm optimal, is to show that the output of the greedy algorithm is no longer than any legal paragraphing. You can represent breaking the words into lines by giving the index of the last word on each line. For example, the previous paragraph can be represented as $[17, 33, 41]$. Start the proof by assuming that the indices of greedy paragraphing are $[i_1, i_2, \ldots, i_k$ and the indices of some other paragraphing are $[j_1, j_2, \ldots, i_m]$, and then argue that $k \leq m$.

Problem 3 (10 points):

Let $G = (V, E)$ be a directed graph with integral edge costs in $\{1, 2\}$. Give an $O(n + m)$ time algorithm that given vertices $s, t \in V$ finds a shortest path from s to t.

Programming Problem 4 (10 points):

The degree of a vertex is the number of neighbors that the vertex has. Write a program to compute the distribution of vertex degrees for an undirected graph. For an input graph G determine the number of vertices of degree d for each d between 0 and n.

Test your program on random graphs using the generator from HW 3. Use values of n of 1000 (or larger). You should report results for values of p in the range 0.002 and 0.02.

Programming Problem 5 (10 points):

Implement the greedy algorithm for graph coloring discussed in class (Lecture 9, Slide 12, Coloring Algorithm version 2). Run the algorithm on random graphs. Use values of n of 1000 (or larger). You should report results for values of p in the range 0.002 and 0.02. How many colors are needed on the average.

Programming Problem 6 Extra Credit (10 points):

Implement a graph coloring algorithm that performs better than the simple greedy coloring algorithm. Compare your results with the maximum degree of the graph, along with the simple greedy algorithm. You should report results for values of p in the range 0.002 and 0.02. How many colors are needed on the average.