

CS 473: Algorithms, Fall 2008

HW 4 (due Tuesday, September 30, 11am)

This homework contains four 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.

1. (25 pts) Problem 4.7 from the text book.
2. (25 pts) Problem 4.15 from the text book.
3. (20 pts) Problem 4.17 from the text book. State and justify the running time of your algorithm.
4. (30pts) In this problem you will analyze an algorithm for computing a minimum spanning tree (MST) of a connected undirected graph $G = (V, E)$ with n vertices and m edges. Each edge $e \in E$ has a cost/weight w_e . Assume for simplicity that the edge weights are distinct. Here is the algorithm.
 - $E' \leftarrow \emptyset$.
 - repeat
 - Compute the connected components of the graph $G' = (V, E')$.
 - Let k be the number of components and S_i be the vertices of the i 'th component
 - for $i = 1$ to k do
 - * find the minimum weight edge $e \in E$ leaving S_i (that is, e has one end point in S_i and the other in $V - S_i$)
 - * $E' \leftarrow E' \cup \{e\}$
 - endfor
 - Until ($k = 1$)
 - return E' as the edges of the MST

Two comments on the algorithm. First, the connected components are computed before the for loop and hence the edges added to E' during the for loop do not change them. Second, an edge e may be added to E' twice in the for loop from two different components but note that E' is a set and hence the second addition is irrelevant.

Analyze the above algorithm using the following steps.

- Prove that the above algorithm correctly computes an MST.
- Prove that the algorithm terminates in $O(\log n)$ iterations of the outer repeat-until loop.

- Show how each iteration of the repeat-until loop can be implemented in $O(m + n)$ time.
- Conclude that the algorithm correctly computes an MST in $O((m + n) \log n)$ time.