College of Computer & Information Science
Northeastern University
Fall 2017

CS5800
MS Algorithms
18 October 2017

Problem set 4 (Due Wednesday, November 1, 11:59 pm)

- The assignment is due at the time and date specified. Late assignments will not be accepted.
- We encourage you to attempt and work out all of the problems on your own. You are permitted to study with friends and discuss the problems; however, *you must write up your own solutions, in your own words.*
- If you do collaborate with any of the other students on any problem, please do list all your collaborators in your submission for each problem.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class (or the class staff) is strictly prohibited.
- We require that all homework submissions be neat, organized, and *typeset*. You may use plain text or a word processor like Microsoft Word or LaTeX for your submissions. If you need to draw any diagrams, however, you may draw them with your hand.

1. (5 points) Identifying most influential reachable nodes

Let $G = (V, E)$ be a directed graph in which each vertex v has an integer label $\ell(v)$ that denotes the *influence* of v. Assume for simplicity that all the influences are distinct. For each $v \in V$, let $r(v)$ denote the vertex in G that has the largest influence among all vertices reachable from v in G.

Design an algorithm that takes as input G, with the influence values $\ell(v)$ for each v, and computes $r(v)$ for each vertex v; that is, your algorithm should return the following set:

$$\{\langle v, r(v)\rangle : v \in V\}.$$

Prove the correctness of your algorithm. Analyze its worst-case running time. The more efficient your algorithm is in terms of its asymptotic worst-case running time, the more credit you may get.

2. (5 points) Determining a certain cycle

Let $G = (V, E)$ be a directed graph. Let $v \in V$ and $u \in V$ be two different vertices in V. Design an algorithm that determines if there exists a path from v to u that contains a (non-empty) cycle.

Prove the correctness of your algorithm. Analyze its worst-case running time. The more efficient your algorithm is in terms of its asymptotic worst-case running time, the more credit you may get.

3. (5 points) Total weighted finish time

A scheduler needs to determine an order in which a set of n processes will be assigned to a processor. The ith process requires t_i units of timw and has weight w_i. For a given schedule, define the *finish time* of process i to be the time at which process i is completed by the processor. (Assume that the processor starts processing the tasks at time 0.)

For example, consider 4 processes with $t_1 = 5$, $t_2 = 2$, $t_3 = 7$, $t_4 = 4$. And weights $w_1 = 1$, $w_2 = 3$, $w_3 = 2$, and $w_4 = 2$. Consider the schedule $1, 3, 2, 4$. The finish time for process 1 is 5, for process 2 is $t_1 + t_3 + t_2 = 5 + 7 + 2 = 14$, for process 3 is $t_1 + t_3 = 5 + 7 = 12$, and for process

4 is $t_1 + t_3 + t_2 + t_4 = 5 + 7 + 2 + 4 = 18$. The *total weighted finish time* of the schedule is then $5 \cdot 1 + 14 \cdot 3 + 12 \cdot 2 + 18 \cdot 2 = 5 + 42 + 24 + 36 = 107$.

Give a greedy algorithm to determine a schedule that minimizes the total weighted finish time. Prove the correctness of your algorithm, and analyze its worst-case running time.

4. (5 points) "Three-Ary Huffman"

You are given a file with characters drawn from an alphabet of size n, with the ith character of the alphabet having frequency $f[i]$. Your goal is to construct a modified "Three-Ary" Huffman code where you are allowed to use 0, 1, or 2 in your codeword. Design an algorithm that encodes each of the n characters with a variable-length codeword over the values 0, 1, and 2 such that no codeword is a prefix of another codeword and so as to obtain the maximum compression.

Prove the correctness of your algorithm and analyze the worst-case running time of your algorithm.

5. (5 points) Planning a largest party that promotes diverse social interactions

You want to throw a big party and have put together a list of your n friends. You also know who knows whom, so you have a list containing which pairs of people know each other. Being a major socialite and a networking guru, you decide that it would best to invite the maximum number of friends possible, subject to the following constraint: at the party, each person should have at least four people they know and at least four people they do not know.

Design an efficient algorithm that takes as input the list of n people and the list of pairs who know each other and outputs the best choice of party invitees. Analyze the worst-case running time in terms of n. The more efficient your algorithm is in terms of its asymptotic worst-case running time, the more credit you may get.

6. (2 + 3 = 5 points) Spanning trees and cycle representatives

Let $G = (V, E)$ be a connected undirected graph, and let $w : E \to \mathbb{R}_{>0}$ be a (positive) weight function on the edges of G.

(a) We define a *maximal spanning tree* for G to be a spanning tree T such that $\sum_{e \in T} w(e)$ is maximal among all the spanning trees of G. Give an algorithm that finds a maximal spanning tree for G.

(b) We say that a set $F \subseteq E$ of edges *represents* all the cycles of G if every cycle in G has an edge in F; that is, for every cycle C of G, $C \cap F \neq \emptyset$. Give an algorithm that finds a set $F \subseteq E$ of *minimal weight*, such that F represents all the cycles of G.