Technical University of Denmark

Written examination, May 16, 2023.

Course name: Algorithms and Data Structures 1

Course number: 02105

Aids: all aids, no internet access.

Duration: 4 hours.

Weight: written part (exercise 1) - 50%, multiple choice part 40%, mandatory exercise - 10%. The weight is approximate. The grade is based on an overall assessment of the multiple-choice part, the written part of the exam, and the mandatory exercise.

Instructions for the exercise

Write your answers to the exercises and submit them using the digital exam system. Your answer should be in the form of a `pdf`-file containing exactly 1 page in the a4-format. The top of the page should clearly list your your study id. The rest of page should contain your answers to the each of the exercises. Number each of your answers with the number of the corresponding exercise. Use a clearly readable font of size 10pt or more and margins of 2cm or more.

Asymptotic bounds should be as tight as possible. Unless otherwise specified, the base of all logarithms is 2 and $\log^k n$ means $(\log n)^k$.

"Give an algorithm" means that you should describe your solution in a short, precise, and unambiguous manner and analyze the complexity of your solution. Unless specified otherwise, the description should be in natural language and not pseudocode. The analysis should explain how you derived the complexity bound.

"Argue correctness of your algorithm" means that you should provide a short argument for correctness of your algorithm.

"Analyze the running time of your algorithm in the relevant parameters (parameters x, y, ..) of the problem" means that you should analyze the running time using the explicitly stated parameters of the problem (parameters x, y, ..).

1 Party!

We are interested in organizing a party for large a group of people. The party consists of x *persons* $p_0, \ldots, p_{x-1}$ and y *friendships* $f_0, \ldots, f_{y-1}$. Each friendship $f = (p, p')$ is a pair of persons p and p' where $p \neq p'$. If (p, p') is a friendship we say that p and p' are *friends*. Also, if p is friends with p', then p' is also friends with p.

1.1 Briefly describe how to model a party as a graph.

1.2 We are now interested in determining if it is possible to partition the persons in the party into two teams T_1 and T_2 such that no person in T_1 is friends with another person in T_1 and no person in T_2 is friends with another person in T_2. Note that since it is a partition of the persons every person should be on exactly one team.

Give an algorithm that given a description of a party determines whether or not we can make such a partition. Argue correctness of your algorithm. Analyze the running time of your algorithm in terms of parameters x and y.

1.3 Unfortunately, you do not have up-to-date contact information for the persons in the party and must rely on friendship to send out invitations. We say that a person p' is *reachable* from person p if there is a sequence of persons starting in p and ending in p' that are pairwise connected by friendships (i.e., each person in the sequence is friends with the next person in the sequence). If p' is reachable from p we also say that p can *reach* p'.

Give an algorithm that given a description of a party and a threshold parameter k, where $x/4 \leq k \leq 3x/4$, determines if it is possible to reach at least k persons in the party from person p_0. Argue correctness of your algorithm. Analyze the running time of your algorithm in terms of parameters x and y.

1.4 You realize that the previous exercise is slightly too simple since the strengths of the friendships are also important in communicating invitations. For each friendship $f = (p, p')$ in the party we associate a *strength*, denoted $\mathsf{strength}(f)$. Each strength is a positive integer in the range $[1, 10 \cdot x^2]$.

Give an algorithm that given a description of a party determines the smallest strength s such that we can reach all persons in the party from person p_0 using only friendships of strength at most s. We assume here that all persons in the party are reachable from p_0 if we use friendships of any strength. Argue correctness of your algorithm. Analyze the running time of your algorithm in terms of parameters x and y.