TDT4120 Algorithms and Data Structures

Examination, August 7, 2020, 09:00–13:00

Academic contact Magnus Lie Hetland

Support material code A

Problems

1 You have an array with n integers and are to find the k largest numbers. Some solutions will have lower asymptotic running time while others will be easier to implement, and have lower overhead due to simpler data structures, for example. Discuss possible solutions with different running times, such as $O(kn)$, $O(n \lg n)$, $O(n \lg k)$ and $O(n)$. Briefly describe how they work, and what advantages and disadvantages they have. Which of your solutions will work if you want to avoid extra memory usage, and perform operations in place, by switching the positions of items? Which of these in-place solutions would you have used in an actual implementation?

Explain and elaborate. Link to relevant theory, possibly in different parts of the curriculum.

2 An article from the fifties describes two problems with corresponding algorithms: to connect all the vertices in a graph using edges with a minimum total weight, and to connect two given nodes using edges with a minimum total weight. Discuss the relationship between the problems and how they are solved. What similarities and differences do you see? Are there differences in which graphs they work on? Argue briefly for your answer.

Explain and elaborate. Link to relevant theory, possibly in different parts of the curriculum.

3 When an algorithm is executed, a "step" is often repeated many times, until we have found the solution for a given instance. How does this relate to decomposing the instance into sub-instances (subproblems)? How does this differ between design methods? What role does mathematical induction play in this context? Which relationship does decomposition have to reductions and hardness proofs, and how is your explanation consistent with this?

Explain and elaborate. Link to relevant theory, possibly in different parts of the curriculum.

4 Your friend Lurvik claims to have invented a new algorithm for finding shortest paths in weighted, directed graphs, where the weights are positive integers. His idea is to transform edges (u, v) with weight $w(u, v) = k > 1$ to paths $\langle u, x_1, x_2, \ldots, x_{k-1}, v \rangle$ with length k, and then use BFS to to find the shortest paths. What are the advantages or disadvantages of this method? Discuss relationships with algorithms in the syllabus. Could you have done something similar in order to find maximum flow with integer capacities? What kind of algorithm would you then have needed to take BFS's place?

Explain and elaborate. Link to relevant theory, possibly in different parts of the curriculum.

5 In basic complexity theory, we tend to use NP as a kind "universe" for the problems we look at. Discuss the pros and cons of this. Give some arguments for this being a large class. Also give examples of types of problems that are *not* in NP, and that still are of great practical significance. Given what we tend to use the class NP for, why is this not so problematic?

Explain and elaborate. Link to relevant theory, possibly in different parts of the curriculum.