Tik–106.4100 Design and Analysis of Algorithms, autumn 2007

Exam, December 14th, 2007

Write the following clearly on top of each paper you submit: "T-106.4100 Design and Analysis of Algorithms, December 14th, 2007", your full name, student ID and study programme, and the total number of papers you submit.

1. a) (2p) Which of the following conjectures are correct and which are incorrect? Give a mathematical justification for each answer!
   i. $n^2 + 7n \in O(n^3)$
   ii. $14n^2 + 5n \log n + 100 \in \Theta(n^2)$

   b) (2p) Explain (using a few sentences) how the average case time complexity of a given algorithm can be calculated. Give the general principle. Do not just explain how to calculate the average case time complexity of some special algorithm (like quicksort). (However, you may clarify your explanation by using an example as long as you also explain the general principle).

   c) (2p) Explain what is the difference between amortized complexity and average case complexity.

2. a) (3p) Solve the following recurrence, when n is a power of two. An exact answer is required (an answer in Θ or O notation is not enough).

$$T(n) = \begin{cases} 1, & \text{when } n = 1 \\ 4T(n/2) + 3n^2 & \text{when } n > 1 \end{cases}$$

   b) (3p) Make a good guess to solve the following recurrence and check your result by using induction.

$$T(n) \leq \begin{cases} 1, & \text{when } 1 \leq n \leq 3 \\ T(n-3) + 2n & \text{when } n > 3 \end{cases}$$

3. a) (1p) For which purpose can a heap (for example a binary heap, a binomial heap or a Fibonacci heap) be used when implementing Prim's algorithm?

   b) (5p) Explain the main differences between binomial heaps and Fibonacci heaps. How do these differences affect the time complexity of different operations on these heaps?

4. (6p) Let $G = (V, E)$ be a directed graph with nodes $v_1, v_2, \ldots, v_n$. We say that G is an *ordered graph* if it has the following properties.

   - Each edge goes from a vertex with a lower index to a vertex with a higher index. That is, every directed edge has the form (v_i, v_j) with $i < j$. However, there may exist pairs of vertices v_i and v_j so that there is no edge (v_i, v_j) although $i < j$.
   - Each vertex except v_n has at least one edge leaving it. That is, for every vertex $v_i, i = 1, 2, \ldots, n-1$, there is at least one edge of form (v_i, v_j).

   The length of a path is the number of edges in it. Your task is to construct an algorithm which solves the following problem: Given an ordered graph G, find the longest path that begins at v_1 and ends at v_n. Your algorithm must use dynamic programming. (If it does not, you cannot obtain full points from your solution.)

   Do not write the code of the algorithm, but explain the principle of your algorithm, present the expressions used by dynamic programming and the order in which the the values of the expressions are calculated. Explain also how the vertices belonging to the longest path can be found out in addition to finding out the the length of the longest path.

5. (6 p) Write pseudocode for an algorithm that classifies the edges of a directed graph into the following classes: tree edge, forward edge, back edge, cross edge. For each edge (u, v), your algorithm has to output the vertices u and v and the class of the edge (u, v). What is the time complexity of your algorithm? Give a short justification for the time complexity.

Please, fill the course feedback form in the web page of the course!