TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

TU Delft

Exam IN4301 Advanced Algorithms – Part I of III

October 11, 2018, 9:00-10:30

Mathijs de Weerdt

- This is a closed book examination with 4 questions worth of 20 points in total.
- Your mark for this exam part will be the number of points divided by 2.
- Use of book, readers, notes, and slides is not allowed.
- Use of (graphical) calculators is not permitted.
- Specify your name, student number and degree program, and indicate the total number of submitted pages on the first page.
- Write clearly, use correct English, and avoid verbose explanations. Giving irrelevant information may lead to a reduction in your score.
  Notice that almost all questions can be answered in a few lines!
- This exam covers Chapters 10 of Kleinberg, J. and Tardos, E. (2005), *Algorithm Design*, all information on the slides of the course of the first 4 lectures, and the set of papers as described in the study guide.
- If your average for all three exam parts is at least a 5.0, and so is the average of the programming exercises, as well as the average of the homework exercises, then your final mark for this course is the average of these three marks, rounded to the nearest half of a whole number. That is, 9.7 is rounded to 9.5, and 5.8 is rounded to 6.
- The total number of pages of this exam is 1 (excluding this front page).

1. (4 points) The *2-weights vertex cover problem* has as input an undirected graph $G = (V, E)$ where each vertex $v \in V$ has an associated weight $w(v)$ of 1 or 2, i.e., $w(v) \in \{1, 2\}$, and as output whether a vertex cover exists with weight at most k exists.

   Describe a search tree algorithm for this problem with an upper bound on the runtime of $O^*(2^k)$. (Mind that this is not a standard unweighted vertex cover problem. The algorithm should thus use $w(v)$ for some v somewhere.)

2. We continue with the 2-weights vertex cover problem from the previous question. We next discuss an algorithm to translate the input (G, k) of the problem to an instance (G', k').

   (a) (1 point) Give the formal definition of when such an algorithm shows that the problem is *kernelizable*.

   A first component of the algorithm is the following rule: isolated vertices will never be part of the cover; simply remove them from the graph. (Rule 1)

   (b) (1 point) Explain how the graph can be reduced if there is a vertex u with only one neighbor (v) (Rule 2). (Hint: Make sure that all cases for $w(u) \in \{1, 2\}$ and $w(v) \in \{1, 2\}$ are covered.)

   (c) (1 point) Explain how the graph can be reduced if there is a vertex v with degree $k' + 1$ or more (Rule 3).

   (d) (3 points) Prove that if a vertex cover of weight at most k exists, the instance (G', k') reduced with the above rules has at most k'^2 vertices.

3. (4 points) Let $(Tr = (T, F), \{V_t : t \in T\})$ be a tree decomposition of $G = (V, E)$. Remove an edge $(t_1, t_2) \in F$ from T, and thus remove the respective vertices $V_{t_1} \cap V_{t_2}$ and their incident edges from G. The result consists of two independent trees T_1, T_2. Let G_{T_i} be the resulting subgraphs associated with the trees T_i. Prove that these subgraphs do not share any vertices.

4. (a) (4 points) Consider the following algorithm to compute the length of the optimal solution for the traveling salesman problem on a set of cities $\{1, 2, \ldots, n\}$ with distances $d(i, j)$ for $i, j \in \{1, 2, \ldots, n\}$.

   foreach $i \in \{1, 2, \ldots, n\}$ **do**
   $M[\{i\}, i] \leftarrow d(1, i)$
   for *size* $s \in 2, 3, \ldots, n$ **do**
   foreach *subset* $S \subseteq \{2, \ldots, n\}$ *of size* s **do**
   foreach $i \in S$ **do**
   ...
   return $\min_{i \neq 1} \{M[\{2, \ldots, n\}, i] + d(i, 1)\}$

   Provide the missing line in the algorithm.

   (b) (2 points) Give a tight upper bound on the runtime of this algorithm, also including the polynomial part.

End of test