COMP 36111

Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Advanced Algorithms 1

Date: Tuesday 16th January 2018

Time: 09:45 - 11:45

Please answer all THREE Questions.

This is a CLOSED book examination

The use of electronic calculators is permitted provided they are not programmable and do not store text

[PTO]

1. Maximal matchings and stable matchings.

   a) Define the term *matching* in the context of a bipartite graph. (2 marks)

   b) Define the notion of a *flow network*. Given a flow network N, explain what a *flow* is for N. (2 marks)

   c) Give an algorithm for computing the maximum flow through a flow network. The algorithm should run in time bounded by a polynomial function of the number of nodes in the flow network and the maximum value of any of the edges in the flow network. You need not prove the correctness of your algorithm. (6 marks)

   d) Explain how maximal matchings in bipartite graphs can be computed in polynomial time by computing maximal flows in flow networks. (2 marks)

   e) Let U and V be finite, non-empty sets of the same cardinality. Assume that each element of U is associated with a total preference ordering for the elements of V, and each element of V is associated with a total preference ordering for the elements of U. What is meant by a stable matching between U and V with respect to the given preferences. (2 marks)

   f) Describe the Gale-Shapley algorithm for generating stable matchings, and explain clearly why it terminates. You need not prove that the matchings it produces are stable. (6 marks)

2. Topological sorting, 2-SAT and NLOGSPACE

a) Let $G = (V, E)$ be a directed, acyclic graph. What is a *topological sort* of G? (2 marks)

b) Using pseudocode, write a linear-time algorithm which, when given a directed graph, computes a topological sort of G if G is acyclic, and returns the word "cyclic" otherwise. (4 marks)

c) Using pseudocode, write a non-deterministic logarithmic-space algorithm which, given a directed graph, $G = (V, E)$ and vertices $u, v \in V$, has a successful run if and only if there is a path in G from u to v. (2 marks)

d) Define the problem *k-SAT*, where $k \geq 2$. (2 marks)

e) Let Γ be an instance of 2-SAT. Let P be the set of proposition letters occurring in Γ, and V the set of literals involving the proposition letters in P. Show how to effectively construct a graph G with vertices V, with the property that Γ is unsatisfiable if and only if there exists a $p \in P$ such that the vertices p and $\neg p$ are reachable from each other in G. You must prove that G has this property. (6 marks)

f) Using your answer to the previous part, show that the complement of the problem 2-SAT is in NLOGSPACE. (2 marks)

g) State the Immerman-Szelepcsényi theorem and deduce that the problem 2-SAT is in NLOGSPACE. You need not prove the Immerman-Szelepcsényi theorem. (2 marks)

3. Linear programming, integer linear programming and NPTIME.

   a) Compute the maximum value of the quantity $2x + y$ subject to the constraints

$$\begin{aligned} 2x + 9y &\leq 90 \\ 8x + y &\leq 80 \\ -3x + 5y &\leq 35, \end{aligned}$$

   where the variables x and y range over the non-negative reals. You may use either graphical reasoning or any other method of your choice, such as the simplex algorithm; however, you must in all cases show your working clearly.

   (8 marks)

   b) Define the problems *linear programming feasibility* (LP) and *integer linear programming feasibility* (ILP).

   (4 marks)

   c) In the context of complexity theory, what is meant by a *many-one logarithmic-space reduction*?

   (2 marks)

   d) Give a many-one logarithmic-space reduction of 3-SAT (defined in Question 2e) to integer linear programming feasibility, and show that it really is a reduction.

   (4 marks)

   e) Explain the term NPTIME-*complete* (with respect to many-one logarithmic-space reductions), and show that that integer linear programming feasibility is NPTIME-complete. You should clearly state any standard results you use.

   (2 marks)

END OF EXAMINATION