Approximation algorithms – final exam

Uriel Feige

20 February 2019

1 General instructions

Please hand in, either via e-mail or a hard copy in my mailbox, by Wednesday February 27 at 2pm.

If you anticipate difficulties in meeting this deadline, let me know as early as possible.

If any of the questions are not clear, please ask for clarifications.

The exam is worth half the final grade of the course, whereas the other half is the average grade of the best five homework assignments. The number of points specified for each question is the maximum one can obtain for it. To get this maximum, the answer has to be correct, explained well, and not too complicated. Partial credit will be given to partly correct answers, or poorly explained answers, or complicated answers in cases where there are considerably simpler answers.

You need to solve the questions by yourself – without assistance from other persons. You may consult references (a book, a published paper, etc.), but if you do so, cite the references that you consult in the appropriate place in your answer. You may use without proof any fact proved in class, but state what you use.

2 The questions

Question 1. A K_3 is a set of three vertices, every two of which is connected by an edge. Consider the following minimization problem that we refer to as K_3 removal (K3R). The input is a graph $G(V,E)$ and a weight function $w : E \to R$ that assigns nonnegative weights to the edges. The goal is to select a subset $E' \subset E$ of edges of minimum weight $\sum_{e \in E'} w(e)$ whose removal from the graph gives a new graph $G(V, E \setminus E')$ with no K_3.

1. Design a strongly polynomial time algorithm that provides a factor 3 approximation for K3R. (10 points.)

2. Write an integer program (IP) representing the K3R problem, in which the variables are $x_{i,j}$ for every edge (i,j), with the interpretation that

$x_{i,j} = 1$ if (i, j) is in E' and $x_{i,j} = 0$ otherwise. Show how to round the linear programming relaxation of this IP so as to get a factor 3 approximation for K3R. Also, write the dual of this linear program, and the complementary slackness conditions for the primal-dual pair of linear programs. (10 points.)

3. Show that every graph contains a solution E' of cost $\sum_{e \in E'} w(e) \leq \frac{1}{2} \sum_{e \in E} w(e)$. Give a deterministic algorithm that finds such a solution. (5 points, as I am providing the hint that bipartite graphs contain no K_3.)

4. Give an approximation algorithm with an approximation ratio better than 3 for K3R. (Up to 15 points, depending on what you find.)

5. Give evidence that approximating K3R beyond a certain ratio is hard. The evidence can be of one or more of the following types: integrality gap examples showing that the value of the LP relaxation can be significantly smaller than the true optimal solution; examples in which the algorithms that you presented fail to find a solution of significantly better quality than that guaranteed by the approximation ratio that you prove; reductions from other problems that show that a good approximation for K3R would either imply P=NP, or lead to progress on some well known open problem. (Up to 10 points, depending on the nature of the evidence and the ratio.)

Question 2. In the following question, you may assume in item 3 that you have a solution for item 2, even if you could not solve item 2.

1. A legal 4-coloring of a graph associates one of four colors to every vertex, such that for every edge its endpoints have different colors. Present a semidefinite program (SDP) relaxation for the 4-coloring problem, show that indeed the SDP is feasible if the input graph is 4-colorable, and briefly explain how one can (approximately) solve the SDP in polynomial time. (15 points.)

2. For some fixed $0 < \delta < 1$, design a randomized polynomial time algorithm, that for every Δ and every 4-colorable graph with maximum degree Δ, provides a legal coloring of its vertices with $O(\Delta^{\delta})$ colors. (Up to 20 points, depending also on the value of δ that you find.)

3. For some fixed $0 < \delta' < \delta$ (where δ is as in item 2), design a randomized polynomial time algorithm, that for every n and every 4-colorable graph on n vertices, provides a legal coloring of its vertices with $O(n^{\delta'})$ colors. (Up to 15 points, depending also on the tradeoff between the value of δ' that you find and the complexity of your answer.)

Good luck.