Massachusetts Institute of Technology
6.854J/18.415J: Advanced Algorithms
Erik Demaine and David Karger

Handout 15
Monday, October 20, 2003

Problem Set 8

Due: Monday, October 27, 2003.

NONCOLLABORATIVE Problem 1. Another way to formulate the maximum flow problem as a linear program is via flow decomposition. Suppose we consider all (exponentially many) s-t paths P in G, and let f_P be the amount of flow on path P. Then maximum flow says to find

$$\begin{aligned} z &= \max \sum f_P \\ \sum_{P \ni e} f_P &\leq u_e \\ f_P &\geq 0 \end{aligned}$$

(the first constraint says that the total flow on all paths through e must be less than u_e). Take the dual of this linear program and give an English explanation of the objective and constraints.

Problem 2. In class we used a "rescaling" argument to show that if one had an absolute approximation algorithm for maximum independent set, one could transform it into a better absolute approximation algorithm for the problem (and in fact, solve the problem exactly). In this problem, we will show that a similar result holds true for relative approximation: that any constant-factor relative approximation algorithm can be improved to a better constant factor relative approximation.

Suppose one has an α-(relative) approximation algorithm for MIS. Consider the following "graph product" operation for a graph G. Create a distinct copy G_v of G for each vertex v of G. Then connect up the copies as follows: if (u, v) is an edge of G, then connect every vertex in G_u to every vertex in G_v.

(a) Prove that if there is an independent set of size k in G, then there is an independent set of size k^2 in the product graph.

(b) Prove that given an independent set of size s in the product graph, one can find an independent set of size $\sqrt{s}$ in G.

(c) Prove that if there is an α-approximation for MIS for *some* fixed α, then there is a polynomial approximation scheme for MIS.

Since MIS was shown to be MAX-SNP-hard, meaning there is some constant to within which it *cannot* be approximated (unless $P = NP$) this proves that MIS has *no* constant-factor relative approximation.

Problem 3. Consider the problem of scheduling, on one machine, a collection of jobs with given *processing times* p_j, *due dates* d_j, and *lateness penalties* (*weights*) w_j paid for jobs that miss their due dates, so as to minimize the total lateness penalty. (If we let U_j denote the indicator variable for job j completing after its due date, we want to minimize $\sum w_j U_j$.)

(a) Argue that any *feasible* subset of jobs (that can all together be completed by their due dates) might as well be scheduled in order of increasing deadline (so it is sufficient to find a set without worrying about order). **Hint:** if two adjacent jobs in the sequence are out of order, swap them.

(b) Assuming the lateness penalties are polynomially bounded integers, give a polynomial-time dynamic program that finds the fastest-completing maximum-weight feasible subset.

(c) Give a fully polynomial-time approximation scheme for the original problem of minimizing lateness penalty with arbitrary lateness penalties.