Analysis of Algorithms	Problem Set 1
CS 6820, Fall 2020	Due Friday, October 2, 6:00pm

Solve ANY FOUR of the five problems on this problem set. You will be awarded 10 points for the problem you skip, even if you skip problem 4 which is worth a maximum of 20 points. In other words, you get a perfect score for the problem you skip unless it is problem 4, in which case you lose 10 points.

In solving the following problems you are allowed to use, without proof, any facts from the lectures or assigned readings in the Fall 2020 version of CS 6820. Hand in your solutions electronically using CMS. Scanned PDF files are allowed, if you prefer to hand-write your solutions. Each group is responsible for submitting just one solution set.

Almost all of the problems are designed to be challenging, and it is acceptable if you fail to solve four of the five problems; in fact, the expectation is that many groups will not produce four correct solutions.

(1) *(10 points)* Suppose you are given a bipartite graph $G = (L, R, E)$ in which each element of L is colored either red or blue. You are also given two numbers, r and b. Design and analyze a polynomial-time algorithm to decide whether G contains a matching M made up of $r + b$ edges, r of which have red left endpoints and b of which have blue left endpoints. If G contains such a matching, your algorithm should also find one.

(2) *(10 points)* Suppose $G = (L, R, E)$ is a bipartite graph with $|L| = |R| = n/2$ and x is a fractional perfect matching in G. The Birkhoff-von Neumann Theorem asserts that x can be expressed as a convex combination of perfect matchings. In other words, there exists a set of perfect matchings $\{M_1, \ldots, M_k\}$ and a probability distribution p on this set, such that for every edge $e = (u, v)$ of the graph, the probability that e belongs to a matching drawn at random from p is x_{uv}. Design and analyze a polynomial-time algorithm to compute such a set $\{M_1, \ldots, M_k\}$ and distribution p, given the graph G and the fractional perfect matching x.

Remark: It's easy to use Web sources to look up a solution to this problem. You must not do that, as it would constitute a violation of academic integrity.

(3) *(10 points)* If S is a vertex set in a graph $G = (V, E)$, let $\delta(S)$ denote the set of all edges from S to the complementary set $V \setminus S$; for one-element sets we'll denote this set by $\delta(v)$ rather than $\delta(\{v\})$.

Define a *fractional perfect quasi-matching (FPQM)* in an undirected (not necessarily bipartite) graph $G = (V, E)$ to be a function $x : E \to [0, 1]$ that satisfies $\sum_{e \in \delta(v)} x(e) = 1$ for all $v \in V$. A *half-integral* FPQM is one in which $x(e) \in \{0, \frac{1}{2}, 1\}$ for all e. In other words, a FPQM labels each undirected edge of G with a non-negative number, such that the sum of edge labels at every vertex equals 1; furthermore, in a half-integral FPQM all of these labels are either 0, $\frac{1}{2}$, or 1.

Suppose we are given a *complete graph* with an even number of vertices, and a cost $c(e)$ for each undirected edge e. Define the cost of a FPQM to be $c(x) = \sum_{e \in E} c(e)x(e)$. Design and analyze a polynomial-time algorithm to compute a FPQM of minimum cost, and prove that the set of minimum-cost FPQMs always contains a half-integral FPQM.

Remark: this implies that every FPQM is a convex combination of half-integral FPQMs, by a separating hyperplane argument similar to the one presented in class for bipartite fractional perfect matchings. When solving this problem, you don't need to write up the proof of this implication.

(4) Define a *fractional perfect matching (FPM)* in an undirected (not necessarily bipartite) graph $G = (V, E)$ to be a function $x : E \to [0, 1]$ that satisfies the following constraints.

1. **[odd set constraints]** $\sum_{e \in \delta(S)} x(e) \geq 1$ for every set S of odd cardinality.
2. **[total size constraint]** $\sum_{e \in E} x(e) = \frac{1}{2}|V|$

Define the support of x to be the set of all edges e such that $x(e) > 0$.

(a) *(8 points)* Let G be an undirected graph, $x : E \to [0, 1]$ a function satisfying the odd set constraints (but not necessarily the total size constraint), and M a matching contained in the support of x. Suppose that the set F of free vertices with respect to M is non-empty, and that for every vertex v either $\sum_{e \in \delta(v)} x(e) = 1$ or the support of x contains an even-length M-alternating simple path from F to v. Prove that the support of x contains an M-augmenting path.

(b) *(4 points)* Let G be an undirected graph and x a fractional perfect matching in G. Prove that the support of x contains a perfect matching. In solving this part of the problem, you may use the result stated in part (a) even if you didn't solve part (a).

(c) *(8 points)* Prove that every fractional perfect matching in an undirected graph G is a convex combination of perfect matchings. In solving this part of the problem, you may use the results stated in parts (a) and (b), even if you didn't solve those parts.

Hint: in parts (a) and (c), use induction on the number of vertices in G.

(5) [adapted from Kleinberg & Tardos, Problem 7.51] *(10 points)* Some friends of yours have grown tired of the game "Six Degrees of Kevin Bacon" (after all, they ask, isn't it just breadth-first search?) and decide to invent a game with a little more punch, algorithmically speaking. Here's how it works.

You start with a set X of n actors and actresses (henceforth called "actors" regardless of gender), and two players P_0 and P_1. Player P_0 names an actor $x_0 \in X$, player P_1 names an actor x_1 who has appeared in a movie with x_0, player P_0 names an actor x_2 who has appeared in a movie with x_1, and so on. Thus, P_0 and P_1 collectively generate a sequence $x_0, x_1, x_2, \ldots$ such that each actor in the sequence has costarred with the one immediately preceding. A player P_i ($i = 0, 1$) loses when it is P_i's turn to move, and he/she cannot name an actor who hasn't been named before.

Suppose you are given a specific set of actors, X, with complete information on who has appeared in a movie with whom. A *strategy* for P_i, in our setting, is an algorithm that takes a current sequence $x_0, x_1, x_2, \ldots$ and generates a legal next move for P_i (assuming it's P_i's turn to move).

Design and analyze a polynomial-time algorithm that decides which of the two players can force a win, in a particular instance of this game. For partial credit, solve the case in which the graph G that specifies the "movie co-star" relation is bipartite.