CS 466/666 Algorithm Design and Analysis **Last Updated:** June 12, 2020
Homework 3 **Due Date: June 29, 2020**

You are allowed to discuss with others but not allowed to use any references except the course notes and the books "Probability and Computing" and "Randomized Algorithms". Please list your collaborators for each question. This will not affect your marks. In any case, you must write your own solutions.

There are totally 60 marks, and the full mark is 50. This homework is counted 10% of the course. The extra marks will not be carried to other parts of the course. Please read the course outline for the late submission policy.

---

1. **Algebraic matching**

   (10 marks) Given a bipartite graph where each edge is red or blue and a parameter k, design an algorithm that determines if there is a perfect matching with exactly k red edges in $O(n^\omega)$ field operations with high probability, when the field size is $\Theta(\text{poly}(n))$.

   In this problem, you can assume that the determinant of a matrix where each entry is an element in $\mathbb{F}[x]$ of degree at most d can be computed in $O(dn^\omega)$ field operations, where $\mathbb{F}[x]$ is the set of single variate polynomials with coefficients in a finite field $\mathbb{F}$ and $\omega \approx 2.37$ is the matrix multiplication exponent. Note that the determinant of such a matrix would be a single variate polynomial, and you can assume that the list of coefficients of this polynomial will be returned to you in $O(dn^\omega)$ field operations.

2. **Network coding**

   (15 marks) Suppose $G = (V, E)$ is a directed acyclic graph and $s \in V$ is the only vertex with indegree zero. In this problem, we would like to design a fast (and distributed) algorithm to compute the edge connectivity from s to v for every $v \in V - s$ (i.e. the number of edge-disjoint directed paths from s to v).

   Consider the following "network coding" algorithm. Let $e_1, e_2, \ldots, e_d$ be the d out-going edges of s. Choose a finite field $\mathbb{F}$. Initially, we assign a d-dimensional unit vector $\vec{e_i}$ to each edge e_i, where $\vec{e_i}$ is the standard unit vector with one in the i-th position and zero otherwise. Then, we follow the topological ordering to process the vertices. When we process a vertex x, there is already a d-dimensional vector computed (where each entry is an element in $\mathbb{F}$) for each of its incoming edge. Now, for each outgoing edge of x, we compute a d-dimensional vector for it by taking a random linear combination of the incoming vectors in x (i.e. random coefficients from $\mathbb{F}$ and arithmetic over $\mathbb{F}$). We repeat this process until every edge in the graph has a d-dimensional vector. Finally, for each vertex v, we compute the rank of its incoming vectors, and return this value as the edge connectivity from s to v.

   Prove that this algorithm outputs the correct answers for all vertices $v \in V - s$ with high probability when $|\mathbb{F}| = \Theta(\text{poly}(|V|))$. Give a fast implementation and an upper bound on the total running time to compute the edge connectivity from s to all vertices $v \in V - s$. You can assume that the rank of an $d \times k$ matrix for $k \leq d$ can be computed in $O(dk^{\omega-1})$ field operations where $\omega \approx 2.37$ is the matrix multiplication exponent.

3. **Graph Drawing**

(10 marks) A graph is *planar* if it can be drawn on the plane such that the edges do not intersect with each other (except that they meet at the vertices). It is a well-known result that a simple planar graph with n vertices can have at most $3n - 6$ edges. We say a graph G has intersecting number k if k is the maximum number such that any drawing of G on the plane has at least k pairs of edges intersecting. By the above result, a simple bound is that the intersecting number is at least t if the graph has at least $3n - 6 + t$ edges. Prove that the intersecting number is at least $m^3/(64n^2)$ for any simple graph with at least $m \geq 4n$ edges. (Hint: Do random sampling and apply the simple bound.)

4. **Isolated Vertices**

(10 marks) Consider a random graph $G_{n,p}$ with $p = c \ln n/n$ for a constant c. Prove that if $c < 1$ then, for any constant $\epsilon > 0$ and for n sufficiently large, the graph has isolated vertices with probability at least $1 - \epsilon$.

(Remark: This implies that the uniform sampling algorithm for graph sparsification in L03 requires $\Omega(n \ln n)$ edges to form a good sparsifier if the input graph is a complete graph.)

5. **Graph Coloring**

(15 marks) Let $G = (V, E)$ be an undirected graph and suppose each $v \in V$ is associated with a set $S(v)$ of $32r$ colors, where $r \geq 1$. Suppose, in addition, that for each $v \in V$ and $c \in S(v)$ there are at most r neighbors u of v such that c lies in $S(u)$.

Use local lemma to prove that there exists a proper coloring of G assigning to each vertex v a color from its class $S(v)$ such that, for any edge $(u, v) \in E$, the colors assigned to u and v are different.

Furthermore, give a polynomial time randomized algorithm to find such a coloring.