CS 466/666 Algorithm Design and Analysis **Last Updated:** June 12, 2026
Homework 2 **Due Date: June 26, 2026**

Please list your collaborators, including AI assistance, for *each* question. This will not affect your marks.

There are totally 65 marks, and the full mark is 50. This homework is counted 5% of the course. Please read the course outline for the late submission policy.

---

1. **Approximate MST in Dynamic Streams**

   (10 marks) Let $G = (V, E, w)$ be a connected undirected graph on n vertices, given by a dynamic stream of edge insertions and deletions. Assume all edge weights are integers in $[1, W]$. You may use as a black box that a spanning forest of an unweighted dynamic graph can be maintained using $\widetilde{O}(n)$ space. Design a one-pass dynamic-streaming algorithm that outputs a $(1+\epsilon)$-approximate minimum spanning tree of G, using $\widetilde{O}\left(\frac{n \log W}{\epsilon}\right)$ space.

2. **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.

3. **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 $v \in V - s$ (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.

4. **Balanced Coloring**

   (10 marks) Suppose we are given a ground set of n elements $U = \{u_1, u_2, \ldots, u_n\}$ and m subsets $S_1, \ldots, S_m \subseteq U$, where each S_i has exactly t elements and each element belongs to at most t subsets. Our goal is to find a 2-coloring of the ground set, say red and blue, so that each subset has roughly the same number of red and blue elements. Show that there exists a 2-coloring in which every subset S_i has at least $t/2 - c\sqrt{t \log t}$ red elements and at most $t/2 + c\sqrt{t \log t}$ red elements for some absolute constant c.

5. **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.)

6. **Fractional Flow**

   (10 marks) Let $G = (V, E)$ be a directed graph and $s, t \in V$ and k is a positive integer. A fractional s-t flow solution with value k is an assignment of each edge e to a fractional value $x(e) \in [0, 1]$, satisfying that

$$\sum_{e \in \delta^{\text{out}}(s)} x(e) - \sum_{e \in \delta^{\text{in}}(s)} x(e) = k$$

   and

$$\sum_{e \in \delta^{\text{in}}(v)} x(e) = \sum_{e \in \delta^{\text{out}}(v)} x(e) \quad \forall v \in V - \{s, t\},$$

   where $\delta^{\text{in}}(v)$ denotes the set of directed edges with v as the head (incoming edges of v), and $\delta^{\text{out}}(v)$ denotes the set of directed edges with v as the tail (outgoing edges of v). Given a fractional s-t flow solution with value k, design a randomized algorithm to return an integral s-t flow with value k in $\tilde{O}(|E|)$ time, i.e., x satisfies the above constraints and moreover $x(e) \in \{0, 1\}$ for all $e \in E$.