CS270 Combinatorial Algorithms and Data Structures **Last Updated:** Feb 19, 2015
Homework 1 (complete) **Due Date: March 19, 2015**

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 110 marks, and the full mark is 100. This homework is counted 20% of the course.

Some hints will be provided by March 12 in a separate file.

---

1. **Minimum k-cut**

   (10 marks) Generalizing on the notion of a cut-set, we define a k-way cut-set in an undirected graph as a set of edges whose removal breaks the graph into k or more connected components. Show that the randomized min-cut algorithm can be modified to find a minimum k-way cut-set in $n^{O(k)}$ time.

2. **Quicksort**

   (10 marks) The expected runtime of the randomized quicksort algorithm is about $2n \ln n$ steps where n is the number of elements to be sorted (see Section 2.5 of "Probability and Computing" for a proof). Prove that the probability that the actual runtime is more than say $100n \ln n$ is at most inverse polynomial in n.

3. **k-wise Independence**

   (10 marks) Suppose that we are given m vectors $v_1, \ldots, v_m \in \{0,1\}^l$ such that any k of the vectors are linearly independent modulo 2. Let $v_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,l})$. Let u be chosen uniformly at random from $\{0,1\}^l$. Let $X_i = \sum_{j=1}^{l} v_{i,j} u_j \mod 2$. Show that the X_i are uniform, k-wise independent random bits.

4. **Almost k-wise Independence**

   (10 marks) Let $X_1, \ldots, X_n \in \{0,1\}$ be n (not necessarily independent) random bits. We say that they are ϵ-almost k-wise independent if for any subset S of size k, we have $|\Pr(\cap_{i \in S}(X_i = b_i)) - 1/2^k| \leq \epsilon$ where $b_i \in \{0,1\}$ for $1 \leq i \leq n$.

   Show that in principle there is a sample space of size $O((2^k k \log n)/\epsilon^2)$ for ϵ-almost k-wise independent bits. That is, show that there exists a set of m n-bit strings $y^{(1)}, y^{(2)}, \ldots, y^{(m)} \in \{0,1\}^n$ where $m = O((2^k k \log n)/\epsilon^2)$, such that if we pick a uniform random n-bit string $y^{(j)}$ and set $X_i = y_i^{(j)}$ then the X_i are ϵ-almost k-wise independent bits.

5. **Online Hiring**

   (10 marks) You need to hire a new staff. There are n applicants for this job. Assume that you will know how good they are (as a score) when you interview them, and the score for each applicant is different. So there is a unique candidate with the highest score, but you don't know that the applicant is the best when you interview him/her until you have interviewed all the applicants. The difficulty is

that after you interview one applicant, you need to make an online decision to either give him/her an offer or forever lose the chance to hire that applicant. Suppose the applicants come in a random order (i.e. a uniformly random permutation), and you would like to come up with a strategy to hire the best applicant.

Consider the following strategy. First, interview m applicants but reject them all. Then, after the m-th applicant, hire the first applicant you interview who is better than all of the previous applicants that you have interviewed.

Let E be the event that you hire the best applicant. Let E_i be the event that the i-th applicant is the best and you hire him/her. Compute $\Pr(E_i)$ and show that

$$\Pr(E) = \frac{m}{n} \sum_{j=m+1}^{n} \frac{1}{j-1}.$$

Prove that $\Pr(E) \geq 1/e$ for an appropriate choice of m.

6. **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. 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.

7. **Algebraic Matching**

(10 marks) In the following two sub-problems, 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 ω is the matrix multiplication exponent. Note that the determinant of such a matrix would be a single variate polynomial.

(a) Given a bipartite graph where each edge is red or blue and a parameter k, determine if there is a perfect matching with exactly k red edges in $O(n^\omega)$ field operations with high probability.

(b) Given a bipartite graph with a non-negative weight on each edge, determine the weight of a maximum weighted perfect matching in $O(Wn^\omega)$ field operations with high probability where W is the maximum weight of an edge.

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

9. **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.

10. **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

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

    and

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

    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 $e \in E$.