CSCI5450 Randomness and Computation **Last Updated:** Fri, Mar 29, 2013
Homework 2 **Due Date: Tue, Apr 23, 2013, in class**

This homework is counted 20% of the whole course. There are total 80 marks.

You are allowed to discuss with others and use any references, but if you do so please list your collaborators and cite your references for each question. This will not affect your marks. In any case, you must write your own solutions (meaning that after you understand the solutions either by your own or from discussions with others or from references, you must write the solutions in your own words without any help from others and without any help from the references). Not writing your own solutions or not listing your collaborators or not citing your references may be considered plagiarism.

If you have any questions or comments please let me (chi@cse.cuhk.edu.hk) and/or your TA Chiu (tckwok@cse.cuhk.edu.hk) know.

---

1. **Graph Coloring**

   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 $S(u)$.

   (a) (10 marks) 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.

   (b) (10 marks) Give a polynomial time randomized algorithm to find such a coloring.

2. **k-SAT**

   (10 marks) Generalize the randomized algorithm for 3-SAT to k-SAT. What is the expected time of the algorithm as a function of k?

   (Remarks: (1) you can assume that the number of clauses is bounded by a polynomial in n where n is the number of variables, (2) you will get most of the marks if you can beat 2^n for constant k.)

3. **Cover Time**

   (10 marks) Prove that the cover time of a simple connected undirected regular graph (i.e. every vertex has the same degree) is $O(n^2 \log n)$.

4. **Card Shuffling**

   We study the following Markov chain on shuffling n cards. In each step, we pick two random cards and exchange their positions.

   (a) (5 marks) Prove that this Markov chain will converge to the uniform distribution (of all permutations) from any initial permutation.

   (b) (10 marks) Prove that $\tau(\epsilon) \leq O(n^2)$ (see L07 for the definition of $\tau(\epsilon)$).

5. **Network Coding**

   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 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 an 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 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 F and arithmetic over 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.

   (15 marks) Prove that this algorithm outputs the correct answers for all vertices $v \in V - s$ with high probability when $F = \Theta(\text{poly}(|V|))$. Give a fast implementation and an upper bound on the total running time to compute the edge connectivity from s to all vertices $v \in V - s$.

6. **Maximum Load**

   (10 marks) We have shown that the maximum load when n items are hashed into n bins using a hash function chosen from a 2-universal family of hash functions is at most $\sqrt{2n}$ with probability at least 1/2. Generalize this argument to k-universal hash functions. That is, find a value such that the probability that the maximum load is larger than that value is at most 1/2. Find the smallest value of k such that the maximum load is at most $3 \ln n / \ln \ln n$ with probability at least 1/2 when choosing a hash function from a k-universal family.