HW 7 (due Monday, 6pm, October 26, 2015)

NEW CS 473: Theory II, Fall 2015 Version: **1.2**

Collaboration Policy: This homework can be worked in groups of up to three students.
Submission is online on moodle.

---

1. (100 PTS.) Much matching about nothing.

(A) (20 PTS.) Read about Hall's theorem and its proof (for example, on wikipedia). A graph is ***regular*** if every vertex has the same number of edges incident to it (and this number is non-zero). Prove that every regular bipartite graph has a perfect matching using Hall's theorem. A matching is ***perfect*** if all vertices are incident on a matching edge.

(B) (20 PTS.) Prove that the edges of a k-regular bipartite graph (i.e., a graph where every vertex has degree k) can be colored using k colors, such that no two edges of the same color share a vertex. Describe an efficient algorithm for computing this coloring.

(C) (20 PTS.) Given a bipartite graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$, describe an algorithm, as efficient as possible, for computing a k-regular bipartite graph that is contained in G, and uses all the vertices of G. Naturally, the algorithm has to return the largest k for which such a graph exists, together with this regular subgraph. Prove the correctness of your algorithm.
(Hint: Modify the simple algorithm seen in class for computing maximum matching in bipartite graphs. Other natural algorithms do not seem to work for this problem.)

(D) (20 PTS.) You are given an algorithm **alg** that in $T(n, m)$ time, can return the largest cardinality matching in a graph with n vertices and m edges. You are given a complete graph G on n vertices, with distinct weights on the edges. Describe an algorithm, as fast as possible, that uses (and does relatively little else) **alg**, and computes the minimum weight w, such that if we remove from G all the edges heavier than w, then the remaining graph $\mathsf{G}_{\leq w}$ contains a perfect matching.

(E) (20 PTS.) You are given a set of n clients, and m shops. A specific shop i can serve at most c_i clients. Every client, has the set of shops they are willing to shop in. Describe an algorithm, as efficient as possible, that decides for a client which shop to use, such that no shop exceeds its capacity [if such a solution exists, naturally]. (You are not allowed to use hashing or network flow in solving this problem.