CS 174 Homework Assignment 6 (due Thursday, March 21)

1. Theorem 5.1 of Motwani and Raghavan shows that every undirected graph with m edges has a bipartite subgraph with at least $m/2$ edges. Describe a derandomized algorithm for finding such a subgraph. (Hint: proceed sequentially through the vertices).

2. Given a set of n vectors $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$, where for each i we have $\mathbf{v}_i \in R^n$ and $|\mathbf{v}_i| = 1$, show that there exist values $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$, where each $\epsilon_i \in \{-1, 1\}$, such that

$$|\epsilon_1 \mathbf{v}_1 + \epsilon_2 \mathbf{v}_2 + \cdots \epsilon_n \mathbf{v}_n| \leq \sqrt{n}$$

and also that there exist values $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$, where each $\epsilon_i \in \{-1, 1\}$, such that

$$|\epsilon_1 \mathbf{v}_1 + \epsilon_2 \mathbf{v}_2 + \cdots \epsilon_n \mathbf{v}_n| \geq \sqrt{n}$$

3. **Finding big cliques**

(a) As in Note 7, let X_k denote the number of k-cliques in a random graph G in the $\mathcal{G}_{n,1/2}$ model. Suppose we fix $n = 1000$ (i.e., we are considering random graphs with 1000 vertices). Plot the values of $\mathrm{E}[X_k]$ as a function of k, for k ranging from 0 to 20. [Note: the formula for $\mathrm{E}[X_k]$ is given in Note 7. You will need to apply some intelligence in this computation, to avoid overflow from the huge factorials. You may, if you wish, compute approximate values (accurate to, say, three figures), but if you do this you must think carefully first and justify your approximations.]

(b) From part (a), deduce a value ϵ for which you can say that

$$\Pr[G \text{ contains a 17-clique}] \leq \epsilon.$$

(c) It is possible to show (by a tedious calculation, which you are *not* expected to do) that the variance of X_k for k in the above range is small, and hence, with high probability, G does in fact contain a clique of size 15. Therefore, we see that the largest clique in G is almost certain to be of size 15 or 16. Let's see if we can find a clique of this size. The most obvious approach is via a *greedy* algorithm, as follows:

start with $K = \emptyset$
for $i := 1$ **to** n **do**
    if vertex i is adjacent to all vertices in K
then add i to K
output K

Show that this algorithm will always output a *maximal clique*, i.e., a clique to which it is not possible to add any more vertices. (Note that a "maximal" clique is not the same as a "largest" clique, though of course a largest clique must be maximal.) Show also that the running time of the algorithm is $O(n^2)$.

(d) Implement the above algorithm. Run your implementation on 100 random graphs of 1000 vertices each, and record in a table the sizes of the cliques that it finds. How far are these cliques from the size of the largest clique we computed in part (c)?

(e) Now let's try to analyze the greedy algorithm. Denote by q_k the probability that the algorithm terminates with a clique of size k. Since this clique (K, say) must be maximal (from part (c)), it must be the case that *no* vertex in $V - K$ is adjacent to all of the vertices in K. Deduce from this fact that $q_k \leq \binom{n}{k}(1 - 2^{-k})^{n-k}$. [Note: be *very careful* in your argument to justify any independence assumptions you are making.]

(f) Define $k_0 = (1 - \epsilon) \lg n$, where ϵ is any positive constant. Deduce from part (e) that $q_{k_0} \leq \mathrm{e}^{-Cn^\epsilon}$ for some constant C (independent of ϵ) and all sufficiently large n. Hence show that

$$\Pr[\text{greedy algorithm outputs a clique of size} \leq (1 - \epsilon) \lg n] \to 0 \quad \text{as } n \to \infty.$$

[Note: With a bit more work, one can show that this is essentially the best that the algorithm does: i.e., one can also show that

$$\Pr[\text{greedy algorithm outputs a clique of size} \geq (1 + \epsilon) \lg n] \to 0 \quad \text{as } n \to \infty,$$

for any constant $\epsilon > 0$. So, roughly speaking, we can say that the algorithm is almost certain to find a clique of size about $\lg n$.]

(g) **EXTRA CREDIT ONLY:** Nobody knows of an efficient algorithm that consistently finds cliques larger than those found by the above greedy algorithm. However, we saw in lectures that such a graph almost certainly contains a clique of size about $2 \lg n$, which is about a factor of 2 larger. Try to develop an efficient algorithm that does better than your greedy algorithm on random graphs of 1000 vertices. I will award a small prize for the winning algorithm (subject, of course, to verification on independently generated random graphs). I will also award a prize to any efficient algorithm that, with reasonable probability, finds cliques of size 15. [Note: if you wish, you may also build some randomness into your algorithm.]