CME 305: Discrete Mathematics and Algorithms

Instructor: Reza Zadeh (rezab@stanford.edu)

Problem Session #1 – 02/10/15

1. **(Kleinberg Tardos 11.10)** Suppose you are given an n by n grid graph G with vertex weights $w(v) \geq 0$ that are all distinct and integer. The goal is to choose an independent set S of nodes of the grid, so that the sum of the weights of the nodes in S is as large as possible.

   Consider the "heaviest-first" greedy algorithm. Start with $S = \emptyset$, and while $|V| \neq 0$ pick the node $v_i \in V$ of maximum weight, add v_i to S and delete its neighbors from G. The resulting S will be an independent set by construction.

   (a) Let S be the independent set returned by the algorithm above, and let T be any other independent set in G. Show that, for each node $v \in T$, either $v \in S$, or there is a node $v' \in S$ so that $w(v) \leq w(v')$ and $(v, v') \in E$.

   (b) Show that this algorithm returns an independent set of total weight at least $\frac{1}{4}$ times the maximum total weight of any independent set in the grid graph G.

   Solution: (a) Let T be an independent set in G and let S be the output of the greedy algorithm. Note that the algorithm terminates when all vertices are either deleted or in set S. Thus for $v \in T$, either $v \in S$ or v was deleted which means it has some neighbor v' of maximal weight moved into set S. This vertex must satisfy $w(v') \geq w(v)$ and we have the desired result.

   (b) Let W^* be the weight of the optimal independent set T. Let W be the weight of set S output by the algorithm. For each $v \in T$, define $v' = v$ if $v \in S$ or as its neighbor in S otherwise. Such a neighbor is guaranteed by part (a) and satisfies $w(v') \geq w(v)$.

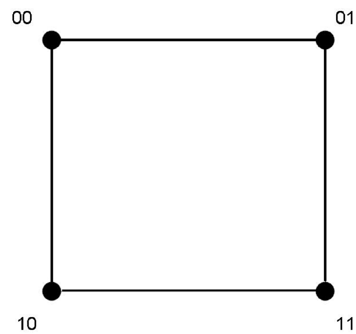
$$\begin{aligned} W^* &= \sum_{v \in T} w(v) \\ &\leq \sum_{v', v \in T} w(v') \\ &\leq 4 \sum_{u \in S} w(u) \\ &= 4W \end{aligned}$$

   where we've used the fact that each $u \in S$ can have at most 4 neighbors in T.

2. An n-dimensional cube can be represented by a graph with 2^n vertices with every vertex corresponding to an n-bit binary number. Two vertices are connected by an edge if their corresponding binary numbers differ by only one bit. For example, the following represents a 2-D cube.

   Prove that every n-dimensional cube has a Hamiltonian cycle.

   Solution: Do this by induction on dimension n. The base case pictured has the Hamiltonian cycle $00, 01, 11, 10$.



We assume there exist one for an n dimensional cube (IH). To show for $n+1$, fix the first bit at 0. Use (IH) on the last n bits to construct a walk using the Hamiltonian cycle without taking the last step. We are now at vertex $0i_1i_2\dots i_n$. Take a step to $1i_1i_2\dots i_n$ switching the first bit. Recreate the walk from before in reverse order. We now find the walk in the same configuration as when we started with the exception of the first bit which is 1. Take the last step switching the first bit to 0. We are back where we started visiting each vertex exactly once and constructing a Hamiltonian cycle in an $n+1$-dimensional cube.

3. **(Lovasz, Pelikan, and Vesztergombi 8.5.6)** A *double star* is a tree that has exactly two nodes that are not leaves. How many unlabeled double stars are there on n nodes?

   Solution: We can represent each unlabeled double star by an unordered pair $\{a, b\}$ where a is the number of leaves adjacent to one of the two non-leaf nodes and b is the number of leaves adjacent to the other. Therefore, to compute the number of double stars we need to find the number of sets $\{a, b\}$ (note that $\{a, b\} = \{b, a\}$) such that $a, b > 0$ and $a + b + 2 = n$. The first condition comes from the fact that the non-leaf nodes must be adjacent to at least one leaf and the second condition is simply the statement that, taken together, the leaves and the two non-leaf nodes make up the entire node set.

   First, assume n is even. Then a can take on values in $A = \{1, 2, \dots, (n-2)/2\}$. A smaller value contradicts $a > 0$ and a larger one implies that $b \le n/2 - 2$ so $b \in A$ and we have counted $\{a, b\}$ twice. Now, assume n is odd. Then a can take on values $A = \{1, 2, \dots, (n-3)/2\}$. A smaller value again contradicts $a > 0$ and a larger one implies $b \le (n-3)/2$ so again $b \in A$ resulting in double counting.

   Thus the number of unlabeled double stars is given by $(n-2)/2$ whenever n is even and $(n-3)/2$ whenever n is odd.

4. **(Lovasz, Pelikan, and Vesztergombi 8.5.10)** If C is a cycle, and e is an edge connecting two nonadjacent nodes of C, then we call e a *chord* of C. Prove that if every node of a graph G has degree at least 3, then G contains a cycle with a chord.

   Solution: Suppose $u_0u_1\dots u_k$ is the longest path in $G(V, E)$. Since $d(u_0) \ge 3$, there exist two neighbors w, v of u_0 not equal to u_1. Since $u_0 u_1 \ldots u_k$ is the longest path, we must have that $w, v \in \{u_2, u_3, \ldots, u_k\}$. Suppose $w = u_i, v = u_j$ with $i \leq j$. Then $u_0 u_1 \ldots u_j u_0$ is a circle with chord $u_0 u_i$.

5. Show that every graph $G = (V, E)$ has a subgraph on at least $|E|/2$ edges which is bipartite.

   Solution: The claim is easily shown by probabilistic method. Consider flipping a fair coin for each node; if heads, put the node in a set S and if tails, do not. So we have a partition of the nodes into two sets, S and $V \setminus S$. Now consider the subgraph obtained by just taking the edges crossing this cut, then we have created a bipartite subgraph of G, and the expected number of edges in the subgraph (in the cut) is $|E|/2$, so by probabilistic method, there must exist some cut (subgraph) with at least this many edges.

6. Suppose $G = (V, E)$ has degree sequence d_1,...,d_n, where $n = |V|$. Show that G has an independent set of size at least:

$$\sum_{j=1}^{n} \frac{1}{d_j + 1}$$

   Hint: consider a random permutation $\pi(*)$ of the vertices and consider the set: $A = \{x \in V | \pi(x) < \pi(v), \forall y \in N(x)\}$.

   Solution: We claim that A is independent set because for every edge, only the lesser node can be in A, so at most one endpoint of the edge is in A. Now we have $\mathrm{E}[A] = \sum_{v \in V} \mathrm{E}[(1_{v \in A})] = \sum_{v \in V} \Pr[v \in A] = \sum_{v \in V} \frac{1}{1+d_v}$, and we are done by probabilistic method.

7. Let G be an undirected simple graph on n vertices and with m edges. Given a pair of vertices (s, t) design a random walk based algorithm to determine whether the two are connected. Your algorithm should must store only the current position of the random walk and have polynomial running time.

   Solution: Here is the algorithm. Fix k such that $1/2^k$ is "very" small. Run a random walk starting at s for $4m(n-1) - 1$ steps or until we hit t. Repeat this process for k trials and report that s and t are disconnected if and only if we do not hit t during any of these k trials. The running time of our algorithm is $O(nm)$ which is polynomial in the input.

   Next, note that $\mathbb{E}(H_{st}) \leq C(G)$ (the hitting time of t starting at s is shorter than the time to hit all vertices). Then applying Markov's inequality and using the result of part (a) we obtain

$$\mathbb{P}(H_{st} \geq 4m(n-1)) \leq \frac{\mathbb{E}(H_{st})}{4m(n-1)} \leq \frac{C(G)}{4m(n-1)} \leq \frac{1}{2}.$$

   This implies that starting at s and running the random walk for $4m(n-1)-1$ steps, the probability that we do not hit t given that s and t are connected (i.e. the probability of of failure) is at most 1/2. If we run this procedure k times and never hit t, the probability that s and t are connected is $\mathbb{P}(H_{st} \geq 4m(n-1))^k \leq 1/2^k$. So if we never hit t we know with high probabililty that s and t are disconnected.

8. Show that every instance of 3-SAT has assignment of variables that satisfy at least a 7/8 fraction of the clauses.

   Solution: We know a clause is not satisfied if all literals are false. Consider tossing a fair coin for each variable to set (heads to true, tails to false). Then all three literals of a clause will be false with probability 1/8. By linearity of expectation, expected number of satisfied clauses = 7/8 times number of clauses. And we are done by probabilistic method.