10 | 10 | 8 | 10

6.856 Dan Ports
Randomized Algorithms drkp@mit.edu
2005/03/16 Collaborators: {sarahl, gpickard}

Problem Set 6

Problem 1:

a) We claim that the set of vertices output over all phases is a maximal independent set. It is an independent set because during each phase, one vertex corresponding to every edge is deleted, so no adjacent vertices are added during one phase, and the neighbors of every vertex are deleted, so no adjacent vertices are added during the next phase. We can see that it is maximal because a vertex is deleted only when it is added to the set or when one of its neighbors is added. Since there are no vertices remaining at the end, one of these is true in every vertex. So the set is maximal because no more vertices can be added.

b) Suppose a vertex v has degree d. The vertex remains marked as long as it is the lower-weight endpoint of each of its edges, i.e. if it has lower weight than any of its d neighbors. Since the weights of the vertices are determined uniformly at random, we can consider their relative rankings to be uniformly distributed. Thus, in the set consisting of v and its d neighbors, v has the lowest weight with probability $\frac{1}{d+1}$.

This analysis assumes that no two vertices are assigned the same weight. We will ensure that unique weights are assigned to the vertices by checking whether any two vertices have the same weight, and if so generating new random weights, and repeating if necessary. This can be performed in parallel using one processor per pair of vertices. The probability that two vertices are assigned the same weight from a set of n^4 weights is $\frac{n}{n^4} = \frac{1}{n^3}$, which is small enough that with high probability $\log n$ attempts will succeed.

c) We use the same definition of a good vertex: a vertex v is good if it has at least $\frac{d(v)}{3}$ neighbors of degree no more than $d(v)$. We claim that a good vertex is removed from the graph in a given phase with constant probability. A vertex is removed from the graph if it or one of its neighbors remains marked. We consider the probability of some neighbor of a good vertex v being selected for the IS. Each neighbor u will be added to the IS with probability $\frac{1}{d(u)+1}$ from above. Since v is good, it has at least $\frac{d(v)}{3}$ neighbors of degree no more than $d(v)$. Each of these neighbors will be selected for the IS with probability at least $\frac{1}{d(v)+1}$, so the probability that v is removed in a given phase is at least

$$\frac{\frac{d(v)}{3}}{d(v)+1} = \frac{1}{3+\frac{3}{d(v)}} \geq \frac{1}{6}$$

since $d(v)$ is at least 1.

Now we have proven that a good vertex is removed from the graph in a given phase with constant probability. In class we showed that if any edge with a good neighbor has constant probability α to be removed — which is precisely what we have just proven — $O(\log m)$ iterations suffice to reduce the number of edges by a polynomial factor, and so we have a RNC algorithm.

Problem 2:

We want to determine the successor matrix representation of the all-pairs shortest path in a graph where edge lengths are integers from 1 to B. Recall that we can perform a witnessed Boolean

matrix multiplication in $O\left(MM(n)\log^2 n\right)$ time. We modify Seidel's algorithm slightly to handle the non-unit edge lengths.

The distance between two neighboring vertices i and k and some other vertex k no longer differs by 1 but rather by B, so we have the inequalities

$$\forall k \;\; D_{ij} - B \leq D_{kj} \leq D_{ij} + B \tag{1}$$
$$\exists k \;\; D_{kj} = D_{ki} + D_{ij} \tag{2}$$

Rather than consider distances mod 3 as in Seidel's algorithm, we consider distances mod $2B+1$. If k is a neighbor on the shortest path from i to j, then $D_{kj} = D_{ki} + D_{ij}$; by Equation 1 this is true if and only if $D_{kj} \equiv D_{ki} + D_{ij}$, and by Equation 2 a neighbor k satisfying this condition must exist. We define the Boolean matrix $D^{(s)}$ to have $D^{(s)}_{kj} = 1$ iff $D_{kj} \equiv s \pmod{2B+1}$. Then k is on the shortest path from i to j if $D_{ij} = D_{ik} + D_{kj}$, which is the case when there exists some integer x $(1 \leq x \leq 2B+1)$ such that $D^{(x)}_{ik} = 1$ and $D^{(D_{kj} - x \bmod 2B+1)}_{kj} = 1$. This is exactly when $\left(D^{(x)} D^{(D_{kj} - x \bmod 2B+1)}\right)_{ij} = 1$. Furthermore, we can use a witnessed Boolean matrix multiplication of these two matrices to identify the successor k.

Define $W^{(x,d)}$ to be the witnessed product $\left(D^{(x)} D^{d-x \bmod 2B+1)}\right)$. Since computations are performed modulo $2B+1$, there are only $O(B)$ possible values for and x and d, and so we can compute all the $W^{(x,d)}$ matrices with $O\left(B^2\right)$ witnessed multiplications. Then it is straightforward to compute our successor matrix from these:

$$S_{ij} = W^{(x,D_{ij})} \text{ for any } x \text{ such that } W^{(x,D_{ij})} \text{ is non-zero.}$$

This algorithm requires $O\left(B^2 MM(n)\log^2 n\right)$ time.

Problem 3:

We consider the problem of finding a minimum-weight perfect matching in a bipartite graph G. Our algorithm is essentially the same as in the unweighted case, except that the edges now have initial weight. As before, we generate a graph G' that is likely to have a unique perfect matching. We do so by scaling up the initial edge weights by a factor of $4mn$, then adding a perturbation selected uniformly at random from 1 to $2m$. The remainder of the algorithm is exactly as in the unweighted case.

To show correctness, we first show that a minimum-weight matching α in G' will indeed always be a minimum weight matching in the original graph G. The matching is minimum weight in G', so no other matching has lower weight in G'. The only case in which this would not be a minimum-weight matching in G would be if there were some other matching β that had lower weight in G but higher weight in G'. We show that this is impossible. Let w'_α be α's weight in G', and w_α be α's weight in G. By definition, $w'_\alpha = (4mn)w_\alpha + x$ for some perturbation x. Similarly, $w'_\beta = (4mn)w_\beta + y$ for some other perturbation y, which is at least $(4mn)(w_\alpha + 1) + y$, since edge weights are integers. So

$$w'_\beta - w'_\alpha \geq (4mn)(w_\alpha + 1) + y - ((4mn)w_\alpha + x) = 4mn + y - x$$

But y and x are the total perturbations on a matching; since each matching has n edges and each edge receives perturbation at most $2m$, x and y are both bounded by $2mn$. So $w'_\beta - w'_\alpha$ will never be negative and so β with lower weight in G but higher weight in G' can exist.

Now we need only show that it is likely that G' will have a unique minimum-weight perfect matching. The proof is based on the Isolating Lemma. First, note that the result above proves there is no danger of a perfect matching with minimum weight in G becoming a minimum-weight matching in G'. So we can safely ignore all perfect matchings that do not have minimum weight in G. Let $\mathcal{F}$ be the set of perfect matchings with minimum weight in G'. Each of these matchings has the

same weight; call it w. So we can subtract away each edge's initial weight, reducing the set system to the unweighted case. This gives a set system $(X, \mathcal{F})$ where X is the set of (now unweighted) edges, and each edge receives a random weight chosen uniformly and independently from 1 to $2m$. The isolating lemma tells us that there is a unique minimum weight set in $\mathcal{F}$ with probability at least $1/2$. This means that with probability $1/2$, there is a perfect matching in $\mathcal{F}$ that has a unique minimum perturbation added to it. Since every perfect matching in $\mathcal{F}$ had minimum weight in G', a matching in $\mathcal{F}$ with unique minimum perturbation is a unique minimum-weight perfect matching, and so our algorithm for finding unique minimum-weight perfect matchings can find it. This proves the claim.

Note that this algorithm is clearly in RNC because the scaling and random perturbation of edge weights can be performed in parallel, then the RNC algorithm for finding the matching can be applied.

Answer the question at the end.

Problem 4:

a) Define Y to be the random variable giving the number of hits that occur with n samples, i.e. $X = \sum_{i=1}^{n} I_i$. Certainly $\mathbf{E}[X] = np$ and so our estimator is $\hat{p} = \frac{1}{n}Y$. We wish to obtain a $\hat{p}$ such that $\mathbf{Pr}[|\hat{p} - p| > \epsilon] \leq \delta$. Equivalently, $\mathbf{Pr}[|Y - np| > n\epsilon] \leq \delta$.

Recall the additive form of the Chernoff bound:

Lemma 1 (Chernoff). *If X is a sum of n independent random variables with mean μ, then*

$$\mathbf{Pr}[X - \mu > n\epsilon] \leq e^{-2n\epsilon^2}$$
$$\mathbf{Pr}[X - \mu < -n\epsilon] \leq e^{-2n\epsilon^2}$$

This gives the immediate corollary that

$$\mathbf{Pr}[|X - \mu| > n\epsilon] \leq 2e^{-2n\epsilon^2}$$

We can apply this bound to Y, since it is a sum of independent indicator random variables. Since $\mu_Y = np$, we obtain

$$\mathbf{Pr}[|Y - np| > n\epsilon] \leq 2e^{-2n\epsilon^2}$$

We want this to occur with probability at most δ, so

$$\begin{aligned} 2e^{-2n\epsilon^2} &\leq \delta \\ -2n\epsilon^2 &\leq \ln \delta/2 \\ 2n\epsilon^2 &\geq \ln 2/\delta \\ n &\geq \frac{1}{2\epsilon^2} \ln 2/\delta \end{aligned}$$

b) Suppose we have an estimation algorithm that estimates p to within a multiplicative factor of $(1 \pm \epsilon)$ with error probability $1/4$ and wish to reduce this error probability to some δ by performing k estimations and taking the median. We give a bound on k as a function of δ.

Let X be the median of k estimations. We first consider $\mathbf{Pr}[X > (1+\epsilon)p]$. Note that this only occurs if at least $k/2$ of the estimations are greater than $(1+\epsilon)p$. Define I_i to be the indicator random variable that takes value 1 if the ith trial exceeds p by a factor of $1+\epsilon$. By definition,

$\mathbf{Pr}[I_i = 1] \leq 1/4$. Then define $Y = \sum_{i=1}^{k} I_i$, $\mathbf{Pr}[X > (1+\epsilon)p] = \mathbf{Pr}[Y > k/2]$. So

$$\begin{aligned}\mathbf{Pr}[X > (1+\epsilon)p] = \mathbf{Pr}\left[Y > \frac{k}{2}\right] &\leq \binom{k}{\frac{k}{2}} \left(\frac{1}{4}\right)^{k/2} \\ &\leq \left(\frac{ke}{\frac{k}{2}}\right)^{k/2} \left(\frac{1}{4}\right)^{k/2} \\ &\leq \left(\frac{e}{2}\right)^{k/2}\end{aligned}$$

Now we want to ensure that this is less than δ. So

$$\begin{aligned}\left(\frac{e}{2}\right)^{k/2} &\leq \delta \\ \frac{k}{2} \ln \frac{e}{2} &\leq \ln \delta \\ k &= O(\log \delta)\end{aligned}$$