Randomized Algorithms (600.464/664)
Assignment 5: Solutions

P.C. Shyamshankar

Due: October 18, 2012

1. Derandomize an algorithm for the version of the set discrepancy problem where each $|A_{ij}| \leq 1$, using the conditional probability method.

We start with the standard randomized algorithm, namely that we choose random variables $X_1, X_2, \ldots, X_n$ each uniformly and independently distributed in $\{-1, +1\}$. We can show that there exists an assignment of values to each $X_i$ such that $\|AX\|_\infty \leq \sqrt{2n \ln 2}$. The weaker bound $2\sqrt{n \ln n}$ still holds with high probability, but we only need to show existence in this case. Whichever bound we use, let us denote that by $c$.

Let us define the events $Q_{2i-1}$ and $Q_{2i}$, for $1 \leq i \leq n$ such that $Q_{2i-1}$ is the event that $A_i X < -c$, and $Q_{2i}$ is the event that $A_i X > +c$. We then choose a value $a_k$ for each $X_k$, given that we have chosen such values for every $X_i$ for $i < k$, as follows:

$$a_k = \arg \min \sum_{i=1}^{2n} P\left[ Q_i | X_k = a_k, \bigwedge_{j=1}^{k-1} X_j = a_j \right]$$

(1)

If the sum starts off as less than one, and we choose the value of $a_k$ that minimizes the sum, then it will always remain less than one, and the final assignment will satisfy $\|AX\|_\infty \leq c$.

As there are polynomially many probabilities to compute over the course of the assignment, the entire algorithm will run in polynomial time if each probability can be computed in polynomial time.

We can compute $P[Q_i | X_1 = a_1, X_2 = a_2, \ldots, X_k = a_k]$ as:
If we were to compute these values as part of a dynamic programming table, we would require the indices to be real-valued, which would negate the advantage of dynamic programming in performing a polynomial time computation.

Therefore, these probabilities cannot be computed in polynomial time, and the derandomized algorithm itself will not run in polynomial time.

2. Derandomize the probabilistic proof of the existence of a 3-partite subgraph of weight at least \( \frac{2W}{3} \) in an edge-weighted graph of total weight \( W \).

We start with a similar algorithm to the bipartite graph probabilistic proof. We define a random variable \( X_i \) for each vertex \( i \) such that each \( X_i \) is independently and uniformly distributed over \( \{0, 1, 2\} \).

For each edge \( e = (i, j) \), let \( Y_e \) be 0 if \( X_i = X_j \), 1 otherwise. In this case,

\[
P[Y_e = 0] = \frac{1}{3} \quad \quad \quad (4)
\]

\[
P[Y_e = 1] = \frac{2}{3} \quad \quad \quad (5)
\]

\[
E[Y_e] = \frac{2w_e}{3} \quad \quad \quad (6)
\]

If \( Y = \sum_e Y_e \) and \( W = \sum_e W_e \), then
\[ E[Y] = \sum_{\in} E[Y_{\in}] \]
\[ = \sum_{\in} \left( w_{\in} \cdot \frac{2}{3} \right) \]
\[ = \frac{2W}{3} \]
\[ \overset{\text{def}}{=} c \]

We can derandomize this algorithm using the conditional expectation method by assigning values to each \( X_i \) such that \( E[Y_{\in}] \) is maximized at each stage. Since we are guaranteed by the probabilistic proof that such an assignment exists, following this assignment sequence will result in one such solution.

For any given \( k \), assuming we have made assignments \( a_i \) for \( X_i \) for \( i \leq k \), we can assign \( a_{k+1} \) for \( X_{k+1} \) as:

\[ a_{k+1} = \arg \max_{a \in \{0, 1, \ldots, p-1\}} E[Y|X_1 = a_1, \ldots, X_k = a_k, X_{k+1} = a] \]

There are three expectations, corresponding to \( a \in \{0, 1, 2\} \) on the right hand side, which must be computed for each of the \( n \) assignments. Since there are \( O(n^2) \) edges, each expectation takes \( O(n^2) \) time to compute, and the entire derandomization process itself takes polynomial time.

At the end of the derandomization process, we will obtain an assignment of values \( a_i \) for each \( X_i \) such that \( Y \geq c \), which is our desired solution.

3. Let \( p \) be any prime, and let \( X_1, X_2, \ldots, X_n \) be independent and uniformly distributed over \( \{0, 1, \ldots, p-1\} \). Let \( a_1, a_2, \ldots, a_n \in \{0, 1, \ldots, p-1\} \), with \( a_1 \neq 0 \). Let \( Y_1, Y_2, \ldots, Y_n \) be defined as:

\[ Y_i = \sum_{j=1}^{i} a_j X_{i-j+1} \]

Prove that \( Y_1, Y_2, \ldots, Y_n \) are independent and uniformly distributed.

The joint distribution of \( Y_1, Y_2, \ldots, Y_n \) is given by:

\[ P \left( \bigwedge_{i=1}^{n} Y_i = c_i \right) = P \left[ a_1 X_1 = c_1, a_1 X_2 + a_2 X_1 = c_2, \ldots, \sum_{j=1}^{n} a_j X_{i-j+1} = c_n \right] \]
The above system of equations has a unique solution, let this solution be $X_i = b_i$. Then,

$$P \left[ \bigwedge_{i=1}^{n} Y_i = c_i \right] = P \left[ \bigwedge_{i=1}^{n} X_i = b_i \right]$$

(14)

$$= \prod_{i=1}^{n} P [X_i] \quad (\because X_i \text{ are independent})$$

(15)

$$= \frac{1}{p^n} \quad (\because X_i \text{ are uniformly distributed})$$

(16)

We can marginalize all but one of the $Y_i$s, to get:

$$P \left[ Y_i = c_i \right] = \sum_{1 \leq j \leq n, j \neq i} \frac{1}{p^n}$$

(17)

$$= \frac{1}{p}$$

(18)

$$\Rightarrow Y_i \text{ are uniformly distributed.}$$

(19)

Also,

$$\prod_{i=1}^{n} P \left[ Y_i = c_i \right] = \prod_{i=1}^{n} \frac{1}{p^n} = \frac{1}{p^n} = P \left[ \bigwedge_{i=1}^{n} Y_i = c_i \right]$$

(20)

$$\Rightarrow Y_i \text{ are independent.}$$

(21)