**MIT 6.859 – Randomness and Computation, with Ronitt Rubinfeld**
Collaborators: Benjamin Rossman and Benjamin Snyder

# Problem Set 1 – Solutions

## Problem 1

Our approximation scheme $\mathcal{B}$ will run $\mathcal{A}$ (with precision $\epsilon$) as a black box $2k+1$ times and will output the median of $\mathcal{A}$'s answers. If at most $k$ of $\mathcal{A}$'s answers are imprecise, $\mathcal{B}$ will output a correct value, because in that event the median is forced to lie in $[f(x)/(1+\epsilon), f(x)(1+\epsilon)]$.

Let $X_i$ be the indicator that the $i$-th call to $\mathcal{A}$ produces an imprecise answer, and set $X = \sum_{i=1}^{2k+1} X_i$. Then the probability that $\mathcal{B}$ outputs an imprecise value is $\delta = \mathbf{Pr}[X > k]$.

We have that $\mu = \mathbf{E}[X] \leq k/2 + 1/4$. Set $\eta = 3/4$, and apply the Chernoff bound $\mathbf{Pr}[X > (1+\eta)\mu] \leq \exp(-\eta^2\mu/3)$ to get that:

$$\delta \leq \exp(-3k/32)$$

Solving for $k$ yields that $k \geq 11\ln(1/\delta)$. Therefore $\mathcal{B}$ requires $2k+1 \geqq 22\ln(1/\delta)+1$ invocations of $\mathcal{A}$, which gives us runtime $O\big((1/\epsilon)|x|\ln(1/\delta)\big)$.

We cannot produce an $\epsilon$-approximation scheme that runs in $O(\log 1/\epsilon)$ time, because the runtime dependence on $\epsilon$ is blackbox and for any $\mathcal{B}$-scheme $\mathcal{A}$ can be adversarially chosen to subvert $\mathcal{B}$'s proper function.

## Problem 2

Let $a_i$ denote the $i$-th row of $A$, let $B_j$ denote the $j$-th column of $B$, and let $c_{ij}$ denote the $(i, j)$-th element of $C$. Our goal is to verify that $a_i B_j = c_{ij}$ for all $i, j \in [n]$.

Generate a random $(1 \times n)$-dimensional 0/1-matrix $R$ and test whether $RAB = RC$. Repeat 2 times, and output "pass" if both times succeed, "fail" otherwise.

Compute $RAB$ as $((RA)B)$ in $O(n^2)$; $RC$ also takes $O(n^2)$ time.

If $C = AB$ the algorithm will output "pass". Otherwise, let $j$ be a column in $C$ within which there are one or more entries $c_{ij} \neq a_i B_j$. Since we are working in $\mathbb{Z}_2$, if $R$ has non-zero entries at an odd number of indices corresponding to errors in $C$, then $RAB \neq RC$ and the errors will be noticed.

Let $i$ be the index of some particular error in the $j$-th column $C_j$ of $C$. Depending on the choice of value of $R$'s entries in non-$j$ locations, either a choice of 0 or 1 at the $j$-th location will ensure that the above condition is met. Either way this happens with probability 1/2.

Therefore one iteration of this routine captures an error with probability 1/2. Hence, two iterations will not omit an error with probability 3/4.

## Problem 3

Let $X_c$ be the indicator that clause $c$ is not satisfied under a random assignemnt of all involved variables. We have that $\mathbf{E}[X_c] = \mathbf{Pr}[X_c = 1] = (1/2)^3 = 1/8$, since there is a unique assignment of the involved variables that doesn't satisfy $c$.

Let $X = \sum_c X_c$, then $E[X] = m/8$ (using linearity of expectation), where $m$ is the number of clauses. Applying Markov, we get $\mathbf{Pr}[X > m/8] < 1$, therefore there exist some assignment which satisfies at least $7m/8$ of the clauses.

## Problem 4

We are trying to calculate:

$$\begin{aligned} R(t) &= \min_n \ (\forall \text{ coloring } \exists \text{ mono subset}) \\ &= \min_n \ \neg\neg(\forall \text{ coloring } \exists \text{ mono subset}) \\ &= \min_n \ \neg(\exists \text{ coloring } \forall \text{ subset non-mono}) \\ &= \max_n \ (\exists \text{ coloring } \forall \text{ subset non-mono}) \end{aligned}$$

Let $n$ and $t$ be parameters. Pick a random coloring. Let $X_s$ be the indicator that a subset $s \subseteq V(K_n)$ of size $t$ is monochromatic. We have that $\mathbf{E}[X_s] = \mathbf{Pr}[X_s = 1] = 2^{1-\binom{t}{2}}$. Let $X = \sum_s X_s$, and by linearity of expectation $\mathbf{E}[X] = \binom{n}{t}2^{1-\binom{t}{2}}$.

When $\mathbf{E}[X] < 1$ we must have that $\mathbf{Pr}[X \neq 0] > 0$, since by Markov's inequality $\mathbf{Pr}[X \geq 1] < 1$ and $X$ is integral. Therefore we conclude that when $\binom{n}{t}2^{1-\binom{t}{2}} < 1$ there exists a coloring for which all $t$-subsets of $K_n$ are non-monochromatic. Eventually:

$$R(t) \geq \max \left\{ n \middle| \binom{n}{t} 2^{1-\binom{t}{2}} < 1 \right\}$$

## Problem 5

**Part I.** Define $x \in S$ to be "bad" if the collection of subsets that contains $x$ and the collection of subsets that don't contain $x$ have the same minimum weight subset.

First, we show that there are no bad elements if and only if there is a unique minimum weight subset. Suppose, there is a unique minimum weight subset. Then there couldn't be a bad element $x$ because that would imply that there are at least two equally weighted minimum weight subsets. Alternatively, suppose there is no unique minimum weight subset. Let $S_1$ and $S_2$ be two different min-weight subsets, and w.l.o.g. let $x$ be such that $x \in S_1$ and $x \notin S_2$, then $x$ is bad.

Let $X_x$ be the indicator that $x$ is bad. Let $w(T)$ for $T \subseteq S$ be the weight of $T$. Let $U_i$ be the subsets containging $x$, and let $W_j$ be the subsets not containing $x$. Then:

$$\begin{aligned}
\mathbf{Pr}[x \text{ bad}] &= \mathbf{Pr}\big[w(x) + \min\ \{w(U_i)\} = \min\ \{w(W_j)\}\big] \\
&= \mathbf{Pr}\big[w(x) = \min\ \{w(W_j)\} - \min\ \{w(U_i)\}\big] \\
&\leq 1/(2m)
\end{aligned}$$

Note that $A = w(x)$ and $B = \min\ \{w(W_j)\} - \min\ \{w(U_i)\}$ are independent random variables, and hence $\mathbf{Pr}[A = B] \leq 1/(2m)$ since at most one value of $A$ hits $B$.

Consequently, $\mathbf{E}[X_x] = 1/(2m)$. Set $X = \sum_x X_x$ and observe that $E[X] = 1/2$. Since $X$ is integral, Markov implies that with probability at least $1/2$ we have $X = 0$.

**Part II.** We proceed with the following setup. As mentioned in class, the Tutte matrix $A$ of the input graph has non-zero variable entries $x_{ij}$ for each edge $(i, j)$. When there is a bipartite matching $\det(A)$ is a non-zero polynomial.

Let us randomly assign $x_{ij} = 2^{w_{ij}}$, where $w_{ij}$ are drawn uniformly from $\{0, \ldots, 2n^2 - 1\}$. And in accordance with the above lemma, let $S$ be the set of all $(i, j)$ pairs that represent an edge, and let $\{S_\alpha\}$ be a collection of subsets of edges that correspond to perfect matchings.

In the event that there is a minimal weight subset in $\{S_\alpha\}$ (where the weight of $(i,j) \in S$ is $w_{ij}$) the determinant's term corresponding to that subset's perfect matching has a value that is $2^{-w}$ times smaller, for $w \geq 1$, than any other term in the determinant. Therefore, it cannot be canceled out by any other term and therefore the determinant is non-zero. Furthermore (for the same reason), the quantity $t_{ij} = \det(B_{ij})2^{w_{ij}}/2^w$ is non-zero for any edge involved in the minimum weight perfect matching.

Alternatively, it is obvious that $t_{ij} = 0$ for all edges that are not involved in perfect matchings, because they are not involved in non-zero terms of the determinant of $A$.

To distinguish the edges of the minimum weight matching from the edges of other matchings, we need to show that $t_{ij}$ is odd if and only if $(i,j)$ belongs to the minimum weight matching. By construction, the determinant's term corresponding to the minimum weight matching will have absolute value exactly equal to $2^w$, where $w$ is the largest power of 2 that divides $\det(A)$.

Therefore, for every edge $(i,j)$ which belongs to a perfect matching, the quantity $t_{ij}$ can be written as:

$$
\begin{aligned}
t_{ij} &= \left(\sum_{\beta} \prod_{e \in \beta} 2^{w_e}\right) \frac{1}{2^w} \\
&= \sum_{\beta} \prod_{e \in \beta} 2^{w_e - w}
\end{aligned}
$$

where $\beta$ iterates over all matchings that involve $(i,j)$. Note that the above quantity is odd if and only if $(i,j)$ is involved in the minimum weight matching. In that event precisely one of the summation terms equals 1 and the remaining ones are powers of 2.

The above line of reasoning holds true when $\{S_\alpha\}$ has a minimum weight subset, which happens at least half of the time as proven in the above lemma. To prevent incorrect output, at the end we check that the produced edges form a perfect matching indeed.