# COMS 4236 – Test 2 Practice Problems

For these practice problems you may use the following Chernoff bounds.

**Chernoff bounds available for this question.** Let $X_1, \ldots, X_t$ be independent 0–1 random variables, $X = \sum_{j=1}^{t} X_j$, and $\mu = \mathbb{E}[X]$.

*Multiplicative form.* For $0 < \delta \leq 1$,

$$\Pr[X \geq (1+\delta)\mu] \leq \exp\left(-\frac{\delta^2 \mu}{3}\right),$$

$$\Pr[X < (1-\delta)\mu] \leq \exp\left(-\frac{\delta^2 \mu}{2}\right).$$

*Additive form (Chernoff–Hoeffding).* For $a > 0$,

$$\Pr[|X - \mu| \geq a] \leq 2\exp\left(-\frac{2a^2}{t}\right).$$

## 1. What would $\mathsf{L} = \mathsf{NL}$ buy at larger space bounds?

Assume $\mathsf{L} = \mathsf{NL}$. Prove that, for every space-constructible $s(n) \geq \log n$, $\mathsf{NSPACE}(s(n)) = \mathsf{DSPACE}(s(n))$. Use the configuration graph of an $s(n)$-space machine, but do not construct that graph explicitly. Account for the space used to simulate an input-head position in the graph's exponentially long encoding and to answer a request for one symbol of that encoding.

**Solution.** The configuration graph has $2^{O(s(n))}$ vertices and an explicit encoding of length $2^{O(s(n))}$. Its adjacency bits and initial/final labels can be generated in $O(s(n))$ space; multiple accepting configurations can be connected to one designated target. By the assumption, STCON has a deterministic logspace decider. Simulate it on the virtual graph encoding: its work tape and virtual input-head position have length $O(\log 2^{O(s(n))}) = O(s(n))$. Recompute the requested encoding symbol using separate $O(s(n))$ workspace, rather than storing the encoding. The total is $O(s(n))$, not its square. The reverse containment is immediate.

## 2. The strength of the reduction matters

(a) Suppose a language $K$ is $\mathsf{PSPACE}$-complete under *polynomial-time* many-one reductions and $K \in \mathsf{NL}$. Prove that $\mathsf{P} = \mathsf{PSPACE}$. Explain why the same simulation argument does not put every language in $\mathsf{PSPACE}$ in $\mathsf{NL}$.

(b) Show unconditionally that no language in $\mathsf{NL}$ can be $\mathsf{PSPACE}$-hard under *logspace* many-one reductions. You may use Savitch's theorem and the deterministic space hierarchy theorem.

**Solution.**

**(a) Polynomial-time reductions.** For any $A \in \mathsf{PSPACE}$, there is a polynomial-time many-one reduction $f$ to $K$. Since $K \in \mathsf{NL} \subseteq \mathsf{P}$, compute $f(x)$ and run a polynomial-time decider for $K$. The reduction's output has polynomial length, so this is a polynomial-time algorithm for $A$. Thus $\mathsf{PSPACE} \subseteq \mathsf{P}$, and the usual reverse inclusion gives $\mathsf{P} = \mathsf{PSPACE}$.

The same simulation does not give an NL algorithm: $f$ is only known to be polynomial-time, not logspace. Even recomputing one bit of $f(x)$ need not use logarithmic work space. The issue is the reduction's work space, not merely its output length; a logspace transducer is perfectly capable of printing a polynomial-length output.

**(b) Logspace reductions.** If $K \in \mathsf{NL}$ were PSPACE-hard under logspace reductions, closure of NL under such reductions would imply $\mathsf{PSPACE} \subseteq \mathsf{NL}$. This closure uses the usual virtual-input simulation: retain the NL decider's configuration and its input-head position, and recompute requested bits of the deterministic logspace reduction's output on demand. Both pieces of workspace are logarithmic because the reduction's output has polynomial length.

Savitch would then imply

$$\mathsf{PSPACE} \subseteq \mathsf{NL} \subseteq \mathsf{DSPACE}((\log n)^2).$$

But deterministic space hierarchy gives

$$B \in \mathsf{DSPACE}((\log n)^3) \setminus \mathsf{DSPACE}((\log n)^2),$$

and $B \in \mathsf{PSPACE}$. This is a contradiction. Hence no such logspace-hard language in NL exists, unconditionally.

## 3. Error amplification: RP versus BPP

Let $A$ be a language and let $0 < \delta < 1/2$ be a desired error bound.

(a) Suppose a polynomial-time randomized algorithm $A$ never accepts an input $x \notin L$ and for every $x \in L$, $Pr_r[A(x, r) = 1] \geq 1/q(n)$ where $q(n) \geq 2$ is a fixed polynomial. Give a polynomial-time algorithm that never accepts a NO instance and has error at most $2^{-n}$ on every YES instance. Prove that your new randomized algorithm achieves this acceptance criterion, and explain why it still runs in polynomial-time.

(b) Suppose instead that a polynomial-time randomized algorithm gives the correct membership answer with probability at least $2/3$ on every input. Give an amplification procedure with error at most $\delta$. Identify the independent indicator variables to which you apply a concentration bound, and give an explicit sufficient number of repetitions.

For both parts, specialize your bound to $\delta = 2^{-n}$ and state whether the amplified algorithm is still polynomial-time.

**Solution.**

**(a) One-sided error, with success at least $1/q(n)$.** Run the given algorithm $k$ times using fresh independent random bits, and accept if at least one run accepts. A NO instance is never accepted. For a fixed YES instance, each run fails with probability at most $1 - 1/q(n)$. Independence gives

$$\Pr[\text{all } k \text{ runs fail}] \leq \left(1 - \frac{1}{q(n)}\right)^k \leq e^{-k/q(n)},$$

where the last inequality uses $1 - u \leq e^{-u}$. To make this at most $\eta$, it suffices to choose

$$\boxed{k = \left\lceil q(n) \ln \frac{1}{\eta} \right\rceil}.$$

In particular, for $\eta = 2^{-n}$, take $k = \lceil q(n)n \ln 2 \rceil$. This is $O(q(n)n)$ repetitions and hence polynomially many.

**(b) Two-sided error, using the stated multiplicative lower tail.** Run the algorithm independently an odd number $k$ of times and output the majority answer. Fix an input. Let $X_i = 1$ if run $i$ gives the *correct* answer and $X_i = 0$ otherwise. Set

$$X = \sum_{i=1}^{k} X_i, \qquad \mu = \mathbb{E}[X] \geq \frac{2k}{3}.$$

A wrong majority means $X < k/2$. Choose $\delta = 1/4$; then

$$(1-\delta)\mu = \frac{3}{4}\mu \geq \frac{3}{4} \cdot \frac{2k}{3} = \frac{k}{2}.$$

Consequently, the event $X < k/2$ is contained in the lower-tail event $X < (1-\delta)\mu$. Apply exactly the supplied inequality:

$$\begin{aligned} \Pr[\text{wrong majority}] &= \Pr[X < k/2] \\ &\leq \Pr[X < (1 - \tfrac{1}{4})\mu] \\ &\leq \exp\left(-\frac{(1/4)^2 \mu}{2}\right) = e^{-\mu/32} \\ &\leq e^{-(2k/3)/32} = e^{-k/48}. \end{aligned}$$

Thus the smallest odd integer

$$\boxed{k \geq 48 \ln \frac{1}{\eta}}$$

suffices. For $\eta = 2^{-n}$, this is $k = O(n)$. If one original run takes time $t(n)$ and uses $b(n)$ random bits, the amplified algorithm takes $O(kt(n) + k)$ time and at most $kb(n)$ random bits. These costs are polynomial in $n$ for both parts when $\eta = 2^{-n}$.

## 4. BPP closure properties

Let $L, L' \in \mathsf{BPP}$. Is $L \cap L' \in BPP$? Justify your answer.

**Solution.** Yes: if $L, L' \in \mathsf{BPP}$, then $L \cap L' \in \mathsf{BPP}$.

First amplify each of the two deciders so that its error on every input is at most $1/6$. Part 3(b) achieves this with a fixed odd number $k \geq 48 \ln 6$ of independent repetitions, so the amplified deciders are still polynomial-time.

On input $x$, run both amplified deciders and accept if and only if *both* accept. There are two cases.

**If $x \in L \cap L'$:** both correct answers are YES. The combined algorithm rejects only if at least one decider makes an error. By the union bound,

$$\Pr[\text{reject}] \leq \frac{1}{6} + \frac{1}{6} = \frac{1}{3}, \qquad \Pr[\text{accept}] \geq \frac{2}{3}.$$

**If $x \notin L \cap L'$:** at least one correct answer is NO. The combined algorithm can accept only if a decider for a language not containing $x$ falsely accepts. Hence

$$\Pr[\text{accept}] \leq \frac{1}{6} \leq \frac{1}{3}.$$

These are the required BPP guarantees, and the total time is polynomial. Independence between the two deciders is not needed for the union bound; each amplification uses fresh independent runs.

## 5. A short-seed PRG would imply $\mathsf{P} = \mathsf{BPP}$

Suppose there is a fixed constant $c$ and a family of deterministic generators

$$G_s : \{0,1\}^{\ell(s)} \longrightarrow \{0,1\}^s, \qquad \ell(s) \le c \log s.$$

The seed length $\ell(s)$ is computable in time polynomial in $s$, and a single algorithm computes $G_s(z)$ from $(1^s, z)$ in time polynomial in $s$. For every Boolean circuit $C$ with $s$ input bits and at most $s$ gates, assume

$$\left| \Pr_{z \in \{0,1\}^{\ell(s)}}[C(G_s(z)) = 1] - \Pr_{r \in \{0,1\}^s}[C(r) = 1] \right| \le \frac{1}{10}.$$

All distributions in this display are uniform. The circuit bound is measured in the *output length* $s$, not in the much shorter seed length.

Prove that $\mathsf{P} = \mathsf{BPP}$. Describe a deterministic simulation of an arbitrary $\mathsf{BPP}$ algorithm, justify its correctness on both YES and NO inputs, and bound its running time. You may use the fact that, after fixing an input $x$, a polynomial-time computation $A(x, r)$ has a circuit of size polynomial in $|x|$ as a function of its random bits.

**Solution.** Fix a BPP decider $A$ and an input $x$ of length $n$. Choose a polynomially bounded $s = s(n)$ large enough both for its random tape and for a circuit $C_x(r) = A(x, r)$; pad the random tape to $s$ bits. The hypothesis applies to each $C_x$, including its hardwired input $x$. On YES inputs, acceptance on a uniform seed is at least $2/3 - 1/10 = 17/30 > 1/2$. On NO inputs, it is at most $1/3 + 1/10 = 13/30 < 1/2$.

Enumerate all $2^{\ell(s)} \le s^c$ seeds, simulate $A(x, G_s(z))$ for each, and take the majority. Enumerate seeds with their multiplicities, even if different seeds produce the same output. The two probability bounds imply that this majority is always correct. The number of seeds, the computation of each output, and each run of $A$ all take polynomial time in $n$. This proves $\mathsf{BPP} \subseteq \mathsf{P}$; the reverse containment is immediate.

## 6. Multiplicative Chernoff

You are given a Boolean formula $\varphi$ on $n \ge 2$ variables. Let

$$p = \Pr_{a \in \{0,1\}^n}[\varphi(a) = 1],$$

where $a$ is uniformly random. You are promised that $p \ge 1/n$, but you do not know $p$.

Give a sampling algorithm that outputs an estimate $\widehat{p}$ satisfying

$$\frac{p}{2} \le \widehat{p} \le \frac{3p}{2}$$

with probability at least $1 - 1/n^2$, using only $O(n \log n)$ evaluations of $\varphi$. Prove your guarantee using the multiplicative Chernoff bound.

**Solution.** Let

$$p = \Pr_{a \in^n}[\varphi(a) = 1], \qquad p \geq \frac{1}{n}, \quad n \geq 2.$$

We must produce $\widehat{p} \in [p/2, 3p/2]$ with probability at least $1 - 1/n^2$, using $O(n \log n)$ evaluations of $\varphi$.

**The two given bounds.** For independent 0–1 variables with sum $X$ and mean $\mu = [X]$, and for $0 < \delta \leq 1$, the first page supplies

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\delta^2\mu/3}, \qquad \Pr[X < (1-\delta)\mu] \leq e^{-\delta^2\mu/2}.$$

We apply these *separately*, keeping their different constants.

**Step 1: Define the sampling algorithm and its mean.** Choose $t$ independent uniform assignments $a^{(1)}, \ldots, a^{(t)}$. Let $X_j = 1$ when $a^{(j)}$ satisfies $\varphi$, and let $X_j = 0$ otherwise. Set

$$X = \sum_{j=1}^{t} X_j, \qquad \widehat{p} = \frac{X}{t}.$$

Each indicator has expectation $p$. Independence of the sampled assignments gives independence of the indicators, and linearity of expectation gives

$$\mu = [X] = \sum_{j=1}^{t} [X_j] = tp.$$

**Step 2: Bound the probability of an estimate that is too large.** The event $\widehat{p} > 3p/2$ is exactly the event $X > 3tp/2 = 3\mu/2$. Since $3\mu/2 = (1+\delta)\mu$ for $\delta = 1/2$, the upper-tail bound gives

$$\begin{aligned} \Pr[\widehat{p} > 3p/2] &= \Pr[X > 3\mu/2] \\ &\leq \Pr[X \geq (1 + \tfrac{1}{2})\mu] \\ &\leq \exp\left(-\frac{(1/2)^2\mu}{3}\right) = e^{-\mu/12} = e^{-tp/12}. \end{aligned}$$

The first inequality only enlarges the event, from $>$ to $\geq$.

**Step 3: Bound the probability of an estimate that is too small.** The event $\widehat{p} < p/2$ is exactly the event $X < tp/2 = \mu/2$. Now $\mu/2 = (1-\delta)\mu$, again with $\delta = 1/2$. The lower-tail bound gives

$$\begin{aligned} \Pr[\widehat{p} < p/2] &= \Pr[X < (1 - \tfrac{1}{2})\mu] \\ &\leq \exp\left(-\frac{(1/2)^2\mu}{2}\right) = e^{-\mu/8} = e^{-tp/8}. \end{aligned}$$

**Step 4: Combine the two failure events.** The estimate fails precisely when it is too small *or* too large:

$$\{\widehat{p} \notin [p/2, 3p/2]\} = \{\widehat{p} < p/2\} \cup \{\widehat{p} > 3p/2\}.$$

Thus the union bound gives

$$\begin{aligned} \Pr[\widehat{p} \notin [p/2, 3p/2]] &\leq e^{-tp/8} + e^{-tp/12} \\ &\leq 2e^{-tp/12} \\ &\leq 2e^{-t/(12n)}. \end{aligned}$$

The second inequality holds because $e^{-tp/8} \le e^{-tp/12}$. The third uses $p \ge 1/n$; increasing $p$ makes the negative exponent more negative, and hence makes the exponential smaller. No independence between these two failure events is required.

**Step 5: Choose the number of samples.** To make the last expression at most $1/n^2$, it suffices that

$$\begin{aligned} 2e^{-t/(12n)} \le \frac{1}{n^2} \quad &\iff \quad e^{-t/(12n)} \le \frac{1}{2n^2} \\ &\iff \quad -\frac{t}{12n} \le -\ln(2n^2) \\ &\iff \quad t \ge 12n \ln(2n^2). \end{aligned}$$

Therefore choose

$$\boxed{t = \left\lceil 12n \ln(2n^2) \right\rceil}.$$

With this choice,

$$\Pr\left[\frac{p}{2} \le \widehat{p} \le \frac{3p}{2}\right] \ge 1 - \frac{1}{n^2}.$$

The algorithm does not need to know $p$. Its sample count depends only on the promised lower bound $p \ge 1/n$. It evaluates $\varphi$ once per sample, for a total of $t = O(n \log n)$ evaluations, and its overall running time is polynomial in the formula's encoding length.