Problem Set 4

- Due date: **7 May, 2023** (released on 24 Apr, 2023)
- The points for each problem is indicated on the side. The total for this set is **90** points but you are expected to answer any **70 points** worth of questions for a full score (anything additional would nevertheless be included in your aggregate score).
- The problem set has a fair number of questions so please do not wait until close to the deadline to start on them. Try and do one question every couple of days.
- Turn in your problem sets electronically (PDF; either LaTeXed or scanned etc.) via email.
- Collaboration with other students taking this course is encouraged, but collaboration with others is not allowed. Irrespective of this, all writeups must be done individually and must include names of all collaborators (if any).
- Referring to sources other than the text book and class notes is STRONGLY DISCOURAGED. But if you do use an external source (eg. other text books, lecture notes, or any material available online), ACKNOWLEDGE all your sources (including collaborators) in your writeup. This will not affect your grades. However, not acknowledging will be treated as a serious case of academic dishonesty.
- Be clear in your writing.

Throughout this problem set, we will identify $\{0,1\}$ with the field $\mathbb{F}_2$ of two elements (just add and multiply modulo 2). You may assume the following fact for granted.

Claim. *Let $0 < r \leq n$ be integers. Suppose $\mathcal{H} = \left\{h_{A,b} : A \in \mathbb{F}_2^{n\times r}, b \in \mathbb{F}_2^r\right\}$ is the family of functions from $\mathbb{F}_2^n \to \mathbb{F}_2^r$ defined as follows:*

$$h_{A,b}(x) = Ax + b.$$

Then, for any $x \neq y \in \mathbb{F}_2^n$ and $a, b \in \mathbb{F}_2^r$, we have

$$\Pr_{A,b}[h_{A,b}(x) = a \text{ and } h_{A,b}(y) = b] = \Pr_{A,b}[h_{A,b}(x) = a] \cdot \Pr_{A,b}[h_{A,b}(y) = b] = \frac{1}{2^r} \cdot \frac{1}{2^r}.$$

The family $\mathcal{H}$ described above has 2^{nr+r} different functions (in contrast with $2^{r\cdot 2^n}$ possible functions from $\mathbb{F}_2^n \to \mathbb{F}_2^r$). This is the standard construction of a "pairwise independent hash family" and you can use the above property freely for this problem set without proof. (The proof isn't hard; you can prove it yourself if you wish but you aren't expected to.)

1. **[**UniqueSAT **and** $\oplus$SAT**]** (5 + 5 + 3 + 7)

   (a) Suppose $S \subseteq \{0,1\}^n$ with $2^{r-2} \leq |S| \leq 2^{r-1}$. Show that there is a fixed constant[1] α such that

   $$\Pr_{A,b}\left[\begin{array}{c}\text{There is a unique } x \in S \\ \text{such that } h_{A,b}(x) = 0^r\end{array}\right] \geq \alpha.$$

   where $h_{A,b} : \mathbb{F}_2^n \to \mathbb{F}_2^r$ is as defined above.

[1] The proof I have in mind gets $\alpha = 1/8$ but any positive constant would do.

(b) Let UniqueSAT be the language consisting of string encoding formulas $\varphi(x)$ that have a unique satisfying assignment. Construct a polynomial time *randomised* reduction from SAT to UniqueSAT with the following guarantee. That is, come up with a randomised algorithm $f : \Sigma^* \to \Sigma^*$ with the following property

$$\begin{aligned} \varphi \notin \text{SAT} &\implies \Pr[f(\varphi) \notin \text{SAT}] = 1 \quad \text{(and hence } f(\varphi) \notin \text{UniqueSAT)} \\ \varphi \in \text{SAT} &\implies \Pr[f(\varphi) \in \text{UniqueSAT}] \geq 1/\text{poly}(n). \end{aligned}$$

[Hint: The proof I have in mind has the success probability to be $1/8n$ where $1/8$ comes from the above lemma and $1/n$ comes from "guessing" the r to use.]

(c) For a formulas φ, let $\#\varphi$ refer to the number of satisfying assignments of φ. Given two formulas φ_1 and φ_2, construct algorithms to build formulas ψ for each of the following guarantees.

i. $\#\psi = (\#\varphi_1) + 1$.
ii. $\#\psi = (\#\varphi_1) + (\#\varphi_2)$.
iii. $\#\psi = (\#\varphi_1) \times (\#\varphi_2)$.

(d) Let $\oplus$SAT be the language consisting of strings encoding formulas $\varphi(x)$ that have an odd number of satisfying assignments. Construct a polynomial time randomised reduction from SAT to $\oplus$SAT with the following guarantees:

$$\begin{aligned} \varphi \in \text{SAT} &\implies \Pr[f(\varphi) \in \oplus\text{SAT}] \geq 1 - 1/2^{\text{poly}(n)}, \\ \varphi \notin \text{SAT} &\implies \Pr[f(\varphi) \in \oplus\text{SAT}] < 1/2^{\text{poly}(n)}. \end{aligned}$$

[Hint: You can begin with f being the reduction to UniqueSAT, since 1 is an odd number, but the success probability is too small. Try "amplifying" the success by using multiple attempts of $f_1(\varphi), \ldots, f_t(\varphi)$ into a single formula ψ so that ψ has an odd number of satisfying assignments if and only if one of the $f_i(\varphi)$'s had an odd number of satisfying assignments.]

2. **[Using hash functions to show** $\text{BPP} \subseteq \Pi_2$**]** (10)

Let $L \in \text{BPP}$. As we saw in class, let $\text{ACC}_M(x) = \{r \in \{0,1\}^m : M(x,r) = \text{accept}\}$. If M is a "BPP-machine" for L, then we can ensure the following guarantee:

$$\begin{aligned} x \in L &\implies |\text{ACC}_M(x)| \geq (1 - 1/2^{\text{poly}(n)}) \cdot 2^m, \\ x \notin L &\implies |\text{ACC}_M(x)| \leq 1/2^{\text{poly}(n)} \cdot 2^m, \end{aligned}$$

Using a suitable family of hash functions $\mathcal{H}$ (by choosing the right output length), build a Π_2 sentence for L that roughly translates to "every $h \in \mathcal{H}$ will create a collision on $\text{ACC}_M(x)$". Choose the $\text{poly}(n)$ in the BPP guarantee, and the right output length for $\mathcal{H}$ to formally show that this sentence is true only for $x \in L$ (thereby proving that $\text{BPP} \subseteq \Pi_2$).

3. **[Round reduction for** AM**]** (3 + 9 + 5 + 3)

We mentioned in our lectures that $\text{AM}[k] = \text{AM}$ for constant k; we will prove this in this problem using some *quantifier jugglery*. For the purpose of this problem, we will use a definition different from that in lecture, the k in $\text{AM}[k]$ and $\text{MA}[k]$ will *not* refer to the number of rounds, but to the number of messages (w/ alternation). In other words, $\text{AM}[3] = \text{AMA}$ and *not* AMAMAM.

Let $\mathsf{prAM}[k]$ be the promise problem version of $\mathsf{AM}[k]$ (i.e, it has the same completeness and soundness properties for the YES and NO instances as $\mathsf{AM}[k]$, but the YES and NO instances do not partition the universe (there could be "don't care" instances)).

For a class C of promise problems, we define $\mathsf{pr}\Sigma \cdot \mathsf{C}$ to be the class of promise problems Π such that there exists a promise problem $\Pi' \in \mathsf{C}$ and a polynomial p for which

$$\begin{aligned} x \in \Pi_Y &\Rightarrow \exists y \in \{0,1\}^{p(n)} \ : \ (x,y) \in \Pi'_Y \\ x \in \Pi_N &\Rightarrow \forall y \in \{0,1\}^{p(n)} \ : \ (x,y) \in \Pi'_N \end{aligned}$$

(You can check for yourself that $\Pi_Y \cap \Pi_N = \emptyset$ since $\Pi'_Y \cap \Pi'_N = \emptyset$.)

Similarly, we define $\mathsf{prBP} \cdot \mathsf{C}$ to be the class of promise problems Π such that there exists a promise problem $\Pi' \in \mathsf{C}$ and a polynomial p for which

$$\begin{aligned} x \in \Pi_Y &\Rightarrow \Pr_{y \in \{0,1\}^{p(n)}}[(x,y) \in \Pi'_Y] \geq 2/3 \\ x \in \Pi_N &\Rightarrow \Pr_{y \in \{0,1\}^{p(n)}}[(x,y) \in \Pi'_N] \geq 2/3 \end{aligned}$$

(You can check for yourself that $\Pi_Y \cap \Pi_N = \emptyset$ since $\Pi'_Y \cap \Pi'_N = \emptyset$.)

(a) Show that for every integer $k \geq 1$, we have $\mathsf{prMA}[k] = \mathsf{pr}\Sigma \cdot \mathsf{prAM}[k-1]$ and $\mathsf{prAM}[k] = \mathsf{prBP} \cdot \mathsf{prMA}[k-1]$, (where $\mathsf{prMA}[0] = \mathsf{prAM}[0] = \mathsf{prP}$ by definition).

(b) Similar to the proof you saw in class for $\mathsf{MA} \subseteq \mathsf{AM}$, show that we have $\mathsf{pr}\Sigma \cdot \mathsf{prBP} \cdot \mathsf{C} \subseteq \mathsf{prBP} \cdot \mathsf{pr}\Sigma \cdot \mathsf{C}$ for any class C of promise problems.

[Hint: You may want to change that $2/3$ to something exponentially close to 1 first (with appropriate justification of course!).]

(c) Prove that for every constant $k \geq 2$, we have $\mathsf{prAM}[k] = \mathsf{prAM}$. Conclude that $\mathsf{AM}[k] = \mathsf{AM}$.

(d) Where in the above parts was it important to work with promise problems instead of languages? And where in the above parts was it important that k is a constant?

4. **[GI is unlikely to be NP-complete]** (6 + 4)

(a) Show that $\mathsf{AM} \subseteq \mathsf{NP/poly}$ (recall, the RHS refers to non-deterministic machines that receive an *advice* string for every fixed input length).

[Hint: Recall the proof of Adleman's theorem of $\mathsf{BPP} \subseteq \mathsf{P/poly}$ and use that here.]

(b) Recall one of your earlier problem set questions where you proved that $\mathsf{coNP} \subseteq \mathsf{NP/poly}$, then $\mathsf{PH} = \Sigma_3 = \Pi_3$. Using this fact[2], show that PH collapses to $\Sigma_3 \cap \Pi_3$ if GI is NP-complete.

5. **[Upper bound for IP]** (3 + 7 + 2 + 3)

In an interactive protocol, the prover and verifier can be abstracted to just be the following 'next-message' functions:

$$\begin{aligned} V : \Sigma^* \times \Sigma^* \times \Sigma^* \to \Sigma^* \qquad & P : \Sigma^* \times \Sigma^* \times \Sigma^* \to \Sigma^* \\ V : (x, r, s_i) \mapsto y_{i+1} \qquad & P : (x, s_i, y_{i+1}) \mapsto z_{i+1} \end{aligned}$$

where x refers to the input, r refers to the instantiation of randomness, and s_i is the message stream so far (i.e., $y_1 \# z_1 \# \cdots y_i \# z_i$).

[2] You are allowed to use it even if you didn't solve the question in the previous problem set. :-)

(a) In the above definition, the prover is assumed to be deterministic and not randomised. Why is this without loss of generality?

(b) Given a specific V, P, and for a fixed x, s_i, let

$$\mathrm{Acc}(V, P, x) = \Pr_r [V(r) \leftrightarrow P \text{ accepts for input } x] .$$

Furthermore, we will extend the above definition to denote

$$\mathrm{Acc}(V, P, x, s_i) = \Pr_r [V(r) \leftrightarrow P, \text{ starting with } s_i, \text{ accepts for input } x] .$$

Let us fix a verifier function V. For each y_{i+1}, define the function P^* as follows:

$$P^*(x, s_i, y_{i+1}) = \arg\max_{z_{i+1}} \mathrm{Acc}(V, P^*, x, s_i \# y_{i+1} \# z_{i+1}).$$

That is, z_{i+1} is the response that maximises the Verifier's probability of acceptance, defined recursively. This is certainly what the Prover ought to respond with in each round in order to maximise their chance of acceptance.

Show that for any given verifier V, the above prover P^* is in PSPACE (functional version rather than decision version).

(c) Conclude that IP $\subseteq$ PSPACE.

(d) Show that if PSPACE $\subseteq$ P/ poly, then PSPACE $=$ MA.

6. **[Perfect-completeness in public-coin interactive proofs]** (15)

In this question, will restrict our attention to public-coin protocols (we outlined in class that this is essentially without loss of generality). However, it could be that we have a public coin protocol with completeness probability $c < 1$. The goal of this question is to modify the protocol to get another public coin protocol that has perfect-completeness.

Given below is a sketch of the protocol. Complete the sketch and argue correctness and perfect-completeness.

> Let us consider the protocol on inputs of length n. We will assume that the protocol uses t rounds, and message is of length exactly m. Note that the verifier's messages are just random strings as this is a public-coin protocol.
>
> In the new protocol, the Prover begins by sending k (to be worked out) strings $s_1, \ldots, s_k \in \{0,1\}^{tm}$. We will think of each s_i as broken into $s_i^{(1)}, \ldots, s_i^{(t)} \in \{0,1\}^m$. The Prover and verifier play k parallel simulations of the old protocol in the following fashion. In the Verifier's turn in round i, the verifier sends a random string $r_i \in \{0,1\}^m$. This is semantically interpreted as the Verifier sending $r_i \oplus s_j^{(i)} \in \{0,1\}^m$ in the j-th parallel simulation of the old protocol. The prover returns answers $y_1, \ldots, y_k \in \{0,1\}^m$ for each of the parallel simulations. The verifier eventually accepts if any of the k simulations resulted in the verifier accepting.