Problem Set 3

- Due date: **15 April, 2024** (released on 30 Mar, 2024).
- The points for each problem is indicated on the side.
- The total for this set is **100** points.
- 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.) on Piazza via private post.
- 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.

---

1. **[One-sided-ness of NP]** (5)

   Show that if $\mathsf{NP} \subseteq \mathsf{BPP}$, then $\mathsf{NP} = \mathsf{RP}$.

2. **[Alternate definition of ZPP]** (5)

   The class ZPP was defined via TMs that could potentially never halt on some random strings, which was a bit awkward to deal with. Let us consider the following different definition for a class ZPP$'$ that also captures 'zero-error randomised algorithms'.

   > A ZPP$'$- machine M for a language L is a randomized machine that always halts in poly(n)-time on *all (random) computational paths* after outputting either 0 or 1 or ? such that $M(x,r) = \{\mathbb{1}(x \in L), \text{'?'}\}$ and $\Pr_r[M(x,r) = \text{'?'}] \leq \frac{1}{2}$. That is, on every input x, the machine M either outputs the correct answer for "$x \in L$" or says it is unsure (by outputting a '?') with an additional guarantee that the machine is unsure on at most half the random paths.

   Prove formally that $\mathsf{ZPP}' = \mathsf{ZPP}$.

3. **[Generalising Karp-Lipton-Sipser]** (2 + 3 + 2 + 3 + 5)

   (i) Consider the following language L corresponding to the encodings of *True* expressions of the form

   $$"(\exists x \in \{0,1\}^m \ \varphi(x)) \Leftrightarrow (\forall y \in \{0,1\}^m \ \psi(y))"$$

   where φ and ψ are some polynomial time computable predicates.
   Show that $L \in \mathsf{PH}$. At what level of the hierarchy is it in?

(ii) Let $L \in \mathsf{NP}$ be any language that you are promised is in $\mathsf{P/poly}$. This means that there is a sequence of "advice strings" $\{z_i\}_{i=1}^{\infty}$ and a polynomial time deterministic TM M such that for all x we have $x \in L \Leftrightarrow M(x, z_{|x|}) = 1$.
Define the following language:

$$\text{ValidAdvice}_L = \left\{(z, n) \ : \ \forall x \in \{0,1\}^n \ \ x \in L \Leftrightarrow M(x, z) = 1\right\}$$

Show that $\text{ValidAdvice}_L \in \mathsf{PH}$. What level of the hierarchy is it in?

(iii) Try and give a different proof of the Karp-Lipton-Sipser theorem (or a possibly weaker statement) using the above observations.

(iv) Suppose $L \in \mathsf{coNP}$ that is promised to be in $\mathsf{NP/poly}$. Show that $\text{ValidAdvice}_L \in \mathsf{PH}$. What level of the hierarchy is it in?

(v) Show that if $\mathsf{coNP} \subseteq \mathsf{NP/poly}$, then PH collapses. (To what level?)

4. **[Kannan's theorem]** (7 + 8 + 5)

(i) Fix any constant $c > 0$. Show that there is a language $L \in \mathsf{PH}$ that is not in $\text{SIZE}(n^c)$.

[Hint: Can you try and encode "the lexicographically smallest circuit of size $10n^c$ that is not computable by c ircuits of size n^c" as a quantified expression?]

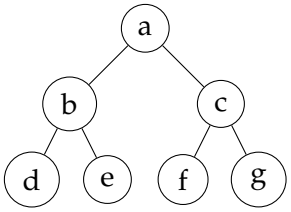
(ii) Show that, for any constant $c > 0$, there is a language in $L \in \Sigma_2^P$ that is not in $\text{SIZE}(n^c)$.

[Hint: Either $\mathsf{NP} \subseteq \mathsf{P/poly}$ or not...]

(iii) Note that this means in particular that, for any constant $c > 0$, we know NP is not computable by circuits of size n^c. Why does this not show that $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ (which, if you recall, is stronger than saying $\mathsf{P} \neq \mathsf{NP}$)?

5. **[Boolean Formula Evaluation]** (7 + 8)

(i) Prove that computing the DFS order (the order of vertices visited, including repetitions, in a DFS traversal that starts at the root and ends at the root) of an *undirected binary tree* $T = (V, E)$ can be done in L (logspace).
For example, the DFS order of the tree below is $a, b, d, b, e, b, a, c, f, c, g, c, a$.



[Hint: (a) You may assume that the tree is described as follows: For every vertex $v \in V$, there is a function $next_v : V \to V \cup \{\top\}$ which gives a clockwise ordering of the edges around the vertex V. I.e., For every vertex v, there is a cyclic ordering among the neighbours of v and $next_v(u)$ is the next neighbour in this cyclic ordering if u is a neighbour of v and $\top$ otherwise. Finally, check that one can make this assumption without loss of generality.]

(ii) A Boolean formula φ on n inputs is a directed tree with n *sources* (vertices with no incoming edges) and one *sink* (vertex with no outgoing edges). All nonsource vertices are called *gates* and are labeled with one of $\vee, \wedge$ or $\neg$. The vertices labeled with $\vee$ or $\wedge$ have fan-in 2 and the vertices labeled with $\neg$ have fan-in 1. Let $x \in \{0,1\}^n$ be some input. The output of φ on x, denoted by $\varphi(x)$, is defined in the natural way. The Boolean formula evaluation problem deals with, given a formula φ on n inputs and $x \in \{0,1\}^n$, computing the value of $\varphi(x)$. Show that formula evaluation can be done in logspace. More precisely, define

$$\text{FormulaEval} = \{\langle \varphi, x \rangle \ : \ \varphi \text{ is a Boolean formula and } \varphi(x) = 1\}$$

Prove that FormulaEval $\in$ L.

6. **[Generalised Geography]** (7 + 8)

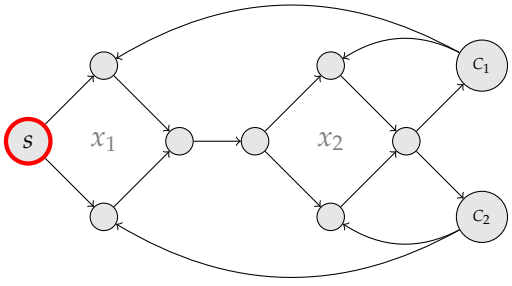
The following is a generalisation of popular games such as *antakshari*, *word-building*, *geography* etc. We are given a directed graph G, and a specified start-vertex s. The game is played between two players P1 and P2 with a token placed at vertex s and P1 moving first. At each turn, the player whose turn it is must move the token from the current vertex u to an adjacent vertex v such that $u \to v$ is an edge, and the token has not been placed at v earlier. (Typically in games such as antakshari, word-building etc., each vertex represents a song / word, with an edge from u to v if the *last-letter* of u equals the *first-letter* of v). If a player is unable to make a valid move (when the token is at a vertex v and all out-neighbours of v have already been 'played' earlier) then that player loses.

We now define the language version of this game.

$$\text{GeneralisedGeography} = \left\{ (G,s) \ : \ \begin{array}{c} \text{P1 has a winning strategy for the game} \\ \text{played on dir. graph } G \text{ with start vertex } s \end{array} \right\}$$

(i) Show that GeneralisedGeography $\in$ PSPACE.

(ii) Show that GeneralisedGeography is in fact PSPACE-complete by reducing TQBF to PSPACE. The following 'example' might be helpful:



$$\exists x_1 \forall x_2 \ : \ (x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2})$$

7. **[Randomised log-space]** (4 + 6)

In this problem, we are going to try and define the class RL just like we defined the class RP. But we will have to handle a few subtleties.

Just like in the witness definition of NL, we can think of a log-space machine M that is given a special 'random tape' that will be filled with a uniformly random string. Like in

the witness definition, this tape will be read-once. We will think of this as a definition of the class RL but we need to address one subtlety.

An important subtlety is what is the halting requirement for an RX (for X replaced by your favourite class)? There are two possible definitions we could think of:

- The machine M must halt on *every* input and *every* setting of the random tape.
- The machine M must halt on *every* input with probability 1 with respect to the random tape contents.

(a) Show that for any time-bounded class X (such as P, or EXP etc.) both the above variants of RX coincide. That is, the distinction is not important when talking about classes such as RP or RE etc.

(b) For space-bounded classes, this becomes an important distinction. Define RL_1 to be the variant where we require the TM to halt on each input with probability 1 over the the random tape contents. Show that RL_1 is infact equal to NL.

Thus, the *right* definition for RL is one where we should insist that every computational path is halting (and not just with probability 1).

8. **[Circuit complexity of multiplication]** (3 + 3 + 6 + 3)

We say in class that given two n-bit numbers, we can compute their sum in AC^0. In this problem we will understand the complexity of multiplying two n-bit numbers.

(a) Show that multiplying two given n-bit numbers *cannot* be done in AC^0 by reducing Parity_m to it.
(Since we are dealing with *really low-level* classes such as AC^0, your reduction has to be *super*-low-level. Use this problem to come upw with a suitable definition for such a "class" of reductions.)

[Hint: $q_2 0 0 b_1 0 0 q_0 \times 1 0 0 1 0 0 1 = ?$]

(b) Build an NC^0 circuit (fan-in 2, constant depth; each output bit can therefore depend on just constantly many input bits!) that takes three n-bit numbers x, y, z and output an n-bit number u and an $(n+1)$-bit number v such that $x + y + z = u + v$.

[Hint:
$$\begin{array}{r} 110101 \\ 011010 \\ 101100 \\ \hline 000011 \\ 111000 \end{array}$$
]

(c) Show that given n numbers that are n-bits long each, we can compute their sum using an NC^1 circuit (fan-in two, $O(\log n)$-depth circuits).

(d) Conclude the product of two given n-bit numbers can be computed in NC^1.