CS 151 Complexity Theory **Spring 2023**

Final

Out: June 1 *Due:* **1pm, Thursday June 8**

This is a final exam. Collaboration is not allowed. You may consult the course notes and the text (Papadimitriou), but not any other source or person. The full honor code guidelines can be found in the course syllabus.

Please attempt all problems. **Please turn in your solutions via Gradescope, by 1pm on the due date.**

1. Define $\mathbf{L}_i$ to be the class of languages decidable by a deterministic Turing Machine using at most $O(\log^i n)$ space, and $\mathbf{NL}_i$ to be the class of languages decidable by a non-deterministic Turing Machine using at most $O(\log^i n)$ space. The classes $\mathbf{L}_1$ and $\mathbf{NL}_1$ should be familiar – they are just deterministic logspace and nondeterministic logspace, respectively.

   (a) Show that for all i, $\mathbf{NC}_i \subseteq \mathbf{L}_i$.

   (b) Show that for all i, $\mathbf{NL}_i$ has $O(\log^{2i} n)$ depth, fan-in 2, Boolean circuits. Your circuits do not need to be uniform.

   (c) It is tempting to try to show that for all i, $\mathbf{NL}_i \subseteq \mathbf{NC}_{2i}$ (since this holds for $i = 1$). Show that this would solve a major open problem. Try to give the strongest implication you can, i.e., if the containment implies A, and A implies B, you should pick A.

2. Consider the following generic setup: out of all 2^n strings in $\{0,1\}^n$, some subset $A \subseteq \{0,1\}^n$ of them are "distinguished." You don't know A directly, but you do have an efficient way to recognize a distinguished string when you see one. That is, there is a small Boolean circuit C with n inputs for which $C(x) = 1$ if and only if $x \in A$. A natural thing to want to do is to estimate the number of distinguished strings. Determining $|A|$ exactly is **#P**-complete but you showed on Problem Set 6 that $|A|$ can be determined *approximately* in $\mathbf{ZPP}^{\mathbf{NP}}$. Here you will show that the related problem of "proving that $|A|$ is large" is in **AM**. We can formalize this as the task of deciding the following promise problem LARGESET:

   - Input: circuit C with n inputs, integer k
   - YES instances: those pairs (C, k) for which $|\{x : C(x) = 1\}| \geq 3 \cdot 2^k$
   - NO instances: those pairs (C, k) for which $|\{x : C(x) = 1\}| \leq \frac{1}{3} \cdot 2^k$

   You will show that LARGESET has an **AM** protocol. The precise meaning of this statement is as follows: given mutual input (C, k), Arthur and Merlin engage in a constant round interactive protocol. If (C, k) is a YES instance, then Merlin has a strategy that causes Arthur to accept with probability 1; if (C, k) is a NO instance, then Arthur rejects with probability at least $1/2$ no matter what Merlin does. The behavior of the protocol is not specified when (C, k) is neither a YES instance nor an NO instance.

(a) For a $k \times n$ matrix M with 0/1 entries and a vector $b \in \{0,1\}^k$, define the function $h_{M,b}(x) : \{0,1\}^n \to \{0,1\}^k$ by $h_{M,b}(x) = Mx + b$ (where all arithmetic is performed modulo 2). Prove that for all $x \in \{0,1\}^n$ and $y \in \{0,1\}^k$,

$$\Pr_{M,b}[h_{M,b}(x) = y] = 2^{-k}$$

and that for all $x_1, x_2 \in \{0,1\}^n$, $x_1 \neq x_2$, and $y_1, y_2 \in \{0,1\}^k$,

$$\Pr[h_{M,b}(x_1) = y_1 \wedge h_{M,b}(x_2) = y_2] = 2^{-2k}.$$

This shows that the family of functions $H = \{h_{M,b}\}$ is a *pairwise independent* family of hash functions from n bits to k bits. The following is a consequence (that you can verify using Chebyshev's Inequality, but you need not prove here): for each fixed $y \in \{0,1\}^k$,

$$\Pr_{M,b}[\exists x \in A \;\; h_{M,b}(x) = y] \geq 1 - \frac{2^k}{|A|}.$$

(b) Using part (a), give an **AM** protocol for LARGESET.

3. Prove that if $\mathbf{PSPACE} \subseteq \mathbf{P/poly}$, then $\mathbf{PSPACE} = \mathbf{MA}$. You may use the following fact: in the proof that $\mathbf{IP} = \mathbf{PSPACE}$, the function describing what message the (honest) prover should send in each round (as a function of the mutual input and the messages seen so far) is computable in polynomial space.

4. Here is a new class involving alternating quantifiers: $\mathbf{S_2^p}$ (the "S" stands for "symmetric alternation"). A language L is in $\mathbf{S_2^p}$ if and only if there is a language $R \in \mathbf{P}$ for which

$$\begin{aligned} x \in L &\Rightarrow \exists y \, \forall z \; (x,y,z) \in R \\ x \notin L &\Rightarrow \exists z \, \forall y \; (x,y,z) \notin R \end{aligned}$$

where as usual $|y| = \text{poly}(|x|)$ and $|z| = \text{poly}(|x|)$. To make sense of this definition it is useful to think of R as defining for each x a 0/1 matrix M_x whose rows are indexed by y and whose columns are indexed by z. Entry (y,z) of matrix M_x is 1 if $(x,y,z) \in R$ and 0 otherwise. Now, the definition says that $x \in L$ if there is an all-ones row in M_x and $x \notin L$ if there is an all-zeros column in M_x (and it is clear that these configurations are mutually exclusive).

(a) Argue that $\mathbf{S_2^p} \subseteq (\mathbf{\Sigma_2^p} \cap \mathbf{\Pi_2^p})$.

(b) The language LEX-FIRST-ACCEPTANCE consists of those pairs (C_1, C_2) for which C_1, C_2 are circuits, and the lexicographically first string x for which $C_1(x) = 1$ is also accepted by C_2. (If there is no lexicographically first string, i.e., C_1 is unsatisfiable, then (C_1, C_2) is not in the language). A bitstring x lexicographically precedes a bitstring y if the first position i in which they differ has $x_i = 0$ and $y_i = 1$. Prove that LEX-FIRST-ACCEPTANCE is $\mathbf{P^{NP}}$-complete. Note: this problem is intended to be challenging.

Hint: as a warm-up, it may be useful to give the reduction from a language $L \in \mathbf{P^{NP}}$ that is decided by a oracle Turing Machine that makes only a *single* oracle query.

(c) Use the previous part to show that $\mathbf{P^{NP}} \subseteq \mathbf{S_2^p}$.

(d) Prove that $\mathbf{MA} \subseteq \mathbf{S_2^P}$.

(e) Prove a stronger form of the Sipser-Lautemann Theorem (Lecture 13): $\mathbf{BPP} \subseteq \mathbf{S_2^P}$.

(f) Prove a stronger form of the Karp-Lipton Theorem (Lecture 13): if SAT has polynomial-size circuits then $\mathbf{PH} = \mathbf{S_2^P}$.