CS 535: Complexity Theory, Fall 2026

Homework 8

Due: 11:59PM, Friday, October 30, 2026.

Reminder. Homework must be typeset with LaTeX preferred. Problem 1 and Problem 3 are required. Problem 2 is optional.

Problem 0 (Reading Project)**.** No need to turn anything in this week, but you can start thinking about what paper you want to read for the end-of-semester reading project. `https://cs-people.bu.edu/mbun/courses/535_F26/handouts/research_project.pdf` (Topic selection due Friday 11/13.)

Problem 1 (*Required* Circuit Lower Bounds for $\mathbf{PH}$)**.** In this problem, you will prove that $\mathbf{PH}$ can compute languages with high circuit complexity. Specifically, you will show that for every integer $k \geq 1$, there is a language in $\mathbf{\Sigma}_2^p$ that cannot be computed by circuits of size at most n^k. (Curiously, this is a non-constructive proof...we know the language exists, but we don't know what it is. For more context, note that the best circuit size lower bound we have for SAT is roughly $5n$.)

(a) Show that for every sufficiently large n, there exists a function $f : \{0,1\}^n \to \{0,1\}$ that is computed by a circuit of size at most n^{k+1}, but not computed by any circuit of size at most n^k.

Hint: This does not directly follow from the statement of the nonuniform hierarchy theorem (Theorem 6.22) in Arora-Barak (do you see why?), but can be proved using the same argument.

(b) Let C, C' be circuits, both on n-bit inputs. Say that C' comes lexicographically before C, written $C' <_{\text{lex}} C$, if the string encoding C' precedes the string encoding C in the lexicographic ordering. Define the language L to consist of all strings x such that $C(x) = 1$, where C is the lexicographically first circuit (on $|x|$-bit inputs) of size at most $|x|^{k+1}$ that is not computed by any circuit of size at most $|x|^k$.

By construction, we have that $L \notin \mathbf{SIZE}(n^k)$. Show that the language $L \in \mathbf{\Sigma}_4^p$, and thereby conclude that $\mathbf{\Sigma}_4^p \not\subseteq \mathbf{SIZE}(n^k)$.

Hint: C is the lexicographically first circuit of size at most n^{k+1} that is not computed by any circuit of size at most n^k if: $|C| \leq n^{k+1}$, the function computed by C cannot be computed by a circuit of size at most n^k, and *for all* $C' <_{\text{lex}} C$ where $|C'| \leq n^{k+1}$, there *exists* a smaller circuit C'' of size $\leq n^k$ such that $C'' \equiv C'$.

(c) Combine part (b) with the Karp-Lipton Theorem ($\mathbf{NP} \subseteq \mathbf{P}_{/\mathbf{poly}} \implies \mathbf{PH} = \mathbf{\Sigma}_2^p$) to show that $\mathbf{\Sigma}_2^p \not\subseteq \mathbf{SIZE}(n^k)$.

(d) Does part (c) imply $\mathbf{\Sigma}_2^p \not\subseteq \mathbf{P}_{/\mathbf{poly}}$? Explain your answer.

Problem 2 (*Optional* $\mathbf{NC}^1$ Captures Poly-Size Boolean Formulas)**.** For this problem, all circuits have fan-in 2 $\{\wedge, \vee\}$ gates, and you can assume all negations are pushed to the bottom. A Boolean formula is a special case of a circuit where every intermediate $\wedge$ or $\vee$ gate has fan-out 1. To make the accounting in this problem easier, we'll write "C has s gates" to mean that C has s $\{\wedge, \vee\}$ gates, explicitly not including the $2n$ inputs and their negations.

In this problem, you will prove that a language is in the complexity class $\mathbf{NC}^1$ if and only if it is also computable by a poly-size family of formulas.

(a) Suppose $f : \{0,1\}^n \to \{0,1\}$ is computed by a circuit with s gates and depth d. Show that f is also computed by a formula of size $O(2^d \cdot s)$.

(b) Now we'll get to work on the converse, showing how to convert a formula into a low-depth circuit. First, prove the following fact about binary trees: For every full binary tree with s vertices, there exists a vertex v such that the subtree rooted at v contains between $s/3$ and $2s/3$ vertices.

(c) Use part (b) to show that every formula F with s gates is equivalent to a formula $F' = (A \wedge B) \vee (C \wedge D)$, where each subformula A, B, C, D has at most $2s/3$ gates.

(d) Use part (c) to show that every formula with s gates has an equivalent formula of depth $O(\log s)$ and $\mathrm{poly}(s)$ gates.

Problem 3 (*Required* Verifying Matrix Products)**.** Suppose you are given matrices $A, B, C \in (\mathrm{GF}(p))^{n \times n}$ where p is a prime number with $p \geq 4n$. (Appendix A.4 of Arora-Barak tells you everything you need to know about finite fields.) The "matrix product verification" problem is to determine whether or not $AB = C$.

(a) First consider the following randomized algorithm solving this problem using n random bits. Sample $r \in \{0,1\}^n$ uniformly at random and check that

$$A(Br) = Cr.$$

Show that this algorithm uses $O(n^2)$ basic arithmetic operations on elements of $\mathrm{GF}(p)$ (and hence can be implemented in polynomial-time) and has the following correctness guarantee: If $AB = C$, the algorithm accepts with probability 1, and if $AB \neq C$, it rejects with probability at least $1/2$.[1]

(b) Describe and analyze an algorithm with the same runtime and correctness guarantees, but that uses only $O(\log n)$ random bits.

Hint: Set $r(t) = (1, t, t^2, \ldots, t^{n-1})$ for an appropriate random choice of t and use the Schwartz-Zippel Lemma.

[1] At the time of this writing, the fastest deterministic algorithm for this problem uses $\approx n^{9/4}$ operations. See `https://github.com/openai/math/blob/main/preprints/Matrix-Multiplication-Nine-Fourths-October-2-2026/paper.pdf`, `https://arxiv.org/abs/2610.28627`, `https://arxiv.org/abs/2610.30054`