

COMS 4236 – Homework 1

DUE ON GRADESCOPE: Oct 1, 11:59pm

Instructions. Please review the course **Syllabus** for policies on collaboration and use of AI before starting this assignment. Please submit a typed (not hand-written) pdf of your solutions on Gradescope, with one question per page. Each problem is worth 10 points (for a total of 60 points).

1. How short can NP witnesses be?

Let A be an NP-complete language, and let $V(x, y)$ be a polynomial-time verifier for A .

- (a) Suppose every YES instance $x \in A$ of length n has an accepting witness of length $O(\log n)$. Prove $P = NP$.
- (b) Now suppose that every YES instance $x \in A$ has a witness of length $O((\log n)^2)$. Assume that A is NP-complete under polynomial-time many-one reductions. Prove that every language in NP would then have a quasipolynomial-time algorithm, carefully accounting for the polynomial blowup of the reduction. Explain why this is evidence against NP-completeness but not a known contradiction.

2. Resolution, and NP versus coNP

For this question we consider the very popular Resolution proof system for proving the unsatisfiability of CNF formulas.

The *Resolution Rule*: Let $C = (x \vee C')$, and $D = (\neg x \vee D')$, where C', D' are disjunctions of literals. Then the Resolution Rule derives the new clause $(C' \vee D')$ from C, D by *resolving* on the variable x . For example, from the clauses $C = (x_2 \vee x_3 \vee \neg x_5)$ and $D = (x_1 \vee \neg x_2 \vee x_3)$ we can derive the clause $(x_1 \vee x_3 \vee \neg x_5)$ by resolving on the variable x_2 . (Note that for the purposes of Resolution, we view clauses as unordered sets of literals, so a literal occurs at most once in a clause.)

Now let $F = C_1 \wedge C_2 \wedge \dots \wedge C_m$ be a CNF formula over Boolean variables $x_1, \dots, x_n$ where each C_i is a clause (ie., a disjunction of literals). A *Resolution refutation* of F is a sequence of clauses (or alternatively a dag of clauses) such that: (1) each clause in the sequence is either a clause from F , or obtained from two previous clauses in the sequence by an application of the Resolution rule; and (2) the last clause in the sequence is the empty clause.

- (a) (Resolution Soundness) Prove that for every CNF formula F , if there exists a Resolution refutation of F , then F is unsatisfiable. (Hint: First prove that the Resolution rule is sound, meaning that if there is an assignment satisfying both C and D , then the assignment also satisfies the resolvent.)
- (b) (Resolution Completeness) [Challenging – won't be graded] Prove that *every* unsatisfiable CNF formula F has a Resolution refutation. What is the best upper bound on the Refutation size in terms of $|F|$?
- (c) (Polytime Checkable). A propositional proof system P is polytime checkable if there is a polynomial-time algorithm that on input a pair (F, Π) accepts the pair if and only if Π is a valid encoding of a P -refutation of F . Describe a reasonable way to encode a CNF F

and a Resolution refutation, Π , and using your encoding, explain why Resolution is polytime checkable.

- (d) (Resolution refutation length) Suppose that *every* unsatisfiable CNF F has a Resolution refutation of size polynomial in $|F|$. Prove that this implies $\text{NP} = \text{coNP}$. (Hint: use the fact that UNSAT is coNP complete).

3. Poly Hierarchy and Alternating Quantifiers

Let R be polynomial-time, and let all unmentioned quantified strings have polynomial length.

- (a) Let $L = \{x : \forall u \in \{0, 1\}^{\lceil c \log |x| \rceil} \exists v R(x, u, v) = 1\}$. Explain briefly why L is in Π_2^P (See Section 2.3, Lectures in Computational Complexity of Jin-Yi Cai (LC) for definition.) Prove the stronger statement $L \in \text{NP}$ by describing a polytime verifier for L .
- (b) Explain why the argument from part (a) fails when the short block $(\forall u)$ is replaced by a polynomial-length block.

4. Languages between P and EXP

Recall the strong time hierarchy theorem which states: If $t(n) \log(t(n)) = o(T(n))$ for suitable time-constructible functions t, T , then $\text{DTIME}(T(n)) \not\subseteq \text{DTIME}(t(n))$. Define quasipolynomial time by $\text{QP} = \bigcup_{c \geq 0} \text{DTIME}(n^{(\log n)^c})$. You may assume the relevant functions are time constructible.

- (a) Prove $\text{QP} \not\subseteq \text{P}$ using the hierarchy theorem with $t(n) = n^{\log \log n}$ and $T(n) = n^{\log n}$. In particular, verify that every polynomial is eventually bounded by t , that T is quasipolynomial, and that $t \log t = o(T)$.
- (b) Now prove that $\text{EXP} \not\subseteq \text{QP}$. (Here you need to pick appropriate choices for the functions $t(n)$ and $T(n)$ and then prove your separation similar to part (a).)

5. Composition of logspace reductions

Suppose that f and g are functions with polynomial-length outputs, so $f(x) = y$ where $|y| = \text{poly}(|x|)$, and similarly for g . Assume that both f and g are computable by deterministic Logspace TMs. Prove that g composed with f , $g(f(x))$, is also computable in deterministic Logspace. Your proof should explain how to bypass storing the entire intermediate string $f(x)$ (which is much too long to store in Logspace) by simulating the outer function g by recomputing requested bits of $f(x)$ on demand.

For this question you should describe the idea behind your algorithm in English. Please do not give a formal algorithm or TM. Instead explain the main steps of the TM computation, and justify why the space complexity of your algorithm is $O(\log |x|)$.

6. Unary NP

A language is *unary* if every input is of the form 1^m . Prove that if every unary language in NP is in P, then $\text{EXP} = \text{NEXP}$. Your proof should construct an appropriate unary padded language for an arbitrary NEXP language and translate polynomial time in the padded length back to exponential time in the original length.