

Homework

Due: November 15

Problem 1: Two languages L_1 and L_2 are logspace equivalent if L_1 is logspace reducible to L_2 and L_2 is logspace reducible to L_1 . Show that the equivalence relation on **EXP** induced by logspace equivalence has infinitely many equivalence classes.

(This makes the empiric phenomenon of natural problems falling into a small number of classes surprising).

Problem 2: Show that the following problems are complete for **NL**:

1. 2SAT
2. The circuit value problem for *stratified circuits*. Here, a Boolean circuit is called *stratified* if it contains either AND-gates or OR-gates but not both, and no negation gates.
3. The problem of deciding whether a nondeterministic finite automaton M has $L(M) = \emptyset$.

Problem 3: In contrast to item 3 of Problem 2, show that the problem of deciding whether a nondeterministic finite automaton M with input alphabet Σ has $L(M) = \Sigma^*$ is complete for **PSPACE**.

Hint for membership in **PSPACE**: A nondeterministic automaton can be simulated by a deterministic one. You may not have enough space to store it as a graph, but you can still “run” it. Also remember that **NPSpace=PSPACE**, so your algorithm can be nondeterministic.

Hint for hardness for **PSPACE**: Given a language in **PSPACE**, consider a one-tape Turing machine M deciding it and consider the tableau of M on a fixed input x . This tableau can be considered a string y over a finite alphabet by concatenating the rows. Try to construct a nondeterministic automaton that

1. accepts all strings *except* y if the tableau is accepting and
2. accepts all strings if the tableau is not accepting.

Problem 4:

1. Show that **EXP[EXP] = EEXP**.
(NB: $A[B]$ is an alternative way of writing A^B)
2. Does one of the following inclusions hold? Which one? Why?
 - (a) **NP[NEXP] \subseteq NEXP[NP]**,
 - (b) **NEXP[NP] \subseteq NP[NEXP]**.

Problem 5: It is an open problem if all languages in **NP** have circuits of size $O(n)$. On the other hand, show that some language L in the polynomial hierarchy does not have linear sized circuits. Hint: Write down a logical expression capturing the lexicographically first truth table describing a function with a circuit of size n^2 but no circuit of size $n \log n$.

Problem 6: Improve the result of Problem 5 to show that there is a language in $\Sigma_2^P \cap \Pi_2^P$ (i.e., “just above” **NP**) which does not have circuits of size $O(n)$. Hint: Use the Karp-Lipton theorem.

Problem 7: Let $\mathbf{E} = \mathbf{DTIME}(2^{O(n)})$ let and $\mathbf{NE} = \mathbf{NTIME}(2^{O(n)})$. A language over a one-letter alphabet is called a *tally* language. Show that $\mathbf{E} \neq \mathbf{NE}$ if and only if there is a tally language in $\mathbf{NP} - \mathbf{P}$.

Define $\Sigma_i^e = \mathbf{NE}[\Sigma_{i-1}^P]$. Show that Σ_i^e is equal to the languages decided by an alternating Turing machine running in time $2^{O(n)}$ and having at most i alternations between AND and OR states in any computation, the first block of states being OR states.

Similarly define $\Pi_i^e = \mathbf{coNE}[\Sigma_{i-1}^P]$ and $\Delta_i^e = \mathbf{E}[\Sigma_{i-1}^P]$. The union of the classes defined is called *the exponential hierarchy*. Show that

1. There is a language in Δ_3^e that requires exponential circuits (Hint: this generalizes the fact that a language in EXPSPACE requires such circuits, and the proof should also generalize the proof of that).
2. There is a language in $\Sigma_2^e \cap \Pi_2^e$ which is not in $\mathbf{P/poly}$. (Hint: Note the similarity to Problem 6).

Problem 8: Show that **BPP** is a subset of $\mathbf{P/poly}$. Hint: First reduce the error probability **BPP**-algorithm to less than 2^{-n} , where n is the length of the input.

Problem 9: Show that if **NP** is a subset of **BPP**, then $\mathbf{NP} = \mathbf{RP}$.

Problem 10: A *Las Vegas* Turing machine is a Turing machine equipped with an extra tape apart from input tape and work tapes, namely, the *random tape*, which is filled with an infinite sequence of random unbiased coin tosses whenever the computation of the machine starts. A language L is said to be **ZPP** if it is decided by a Las Vegas Turing machine which halts with probability 1 on all inputs, has an *expected* running time which is polynomial and which, when halting, always correctly accepts elements of L and rejects non-elements. Show that $\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}$. Show that **BPP** is a subset of $\mathbf{ZPP}[\mathbf{NP}]$, where the latter class is defined as **ZPP**, but with the Las Vegas Turing machine having access to an **NP**-oracle. Hint: Use $\mathbf{pBPP} = \mathbf{pRP}[\mathbf{pRP}]$.