

**Due on October 27**  
**In class, at the beginning of the lecture**

*The work you submit must be your own.* You may discuss problems with each other; however, you should prepare written solutions alone. In particular, you should not leave with any written notes from such discussions. The style and clarity of your answers will be an important factor in the grade.

Five questions, each worth %20.

1. This exercise is just an *amuse-gueule* for your enjoyment.

A game is *finite* if there is an *a priori* upper bound on the number of moves after which the game is over and one of the two players is declared the victor (there are no draws). A game has *perfect information* if both players know everything (i.e., they know the entire history of the game). Show that in every finite 2-person game with perfect information one of the two players has a winning strategy<sup>1</sup>.

2. In this problem you will show that three different definitions of the Polynomial Time Hierarchy **PH** are equivalent. Let  $\Sigma_0\mathbf{P} = \Sigma_0^{\text{Alt}} = \Sigma_0^p = \mathbf{P}$  (and same for  $\Pi_0$ ).

(a) Let  $\Sigma_{i+1}\mathbf{P} = \mathbf{NP}^{\Sigma_i\mathbf{P}}$ , and  $\Pi_{i+1}\mathbf{P} = \mathbf{co-NP}^{\Sigma_i\mathbf{P}}$ .

(b) An *Alternating Turing Machine* is a non-deterministic TM with an additional feature: all its states, except  $q_{\text{accept}}$  and  $q_{\text{reject}}$ , are divided into two groups: universal and existential. We give a (recursive) definition of what it means for a node to be accepting: a universal (existential) node, is accepting if all (some) of its children are accepting. We determine acceptance by looking at the root node.

Let  $\Sigma_i^{\text{Alt}}$  be a *polytime* ATM where the the initial state is existential and there are at most  $i$  “runs” of states (i.e., a sequences of existential states, followed by a sequence of universal states, then another sequence of existential states, etc., and there are at most  $i$  such sequences).  $\Pi_i^{\text{Alt}}$  is defined analogously, except it starts with a universal state.

(c) The definition of  $\Sigma_i^p$  and  $\Pi_i^p$  given in class.

Show that for all  $i$ ,  $\Sigma_i\mathbf{P} = \Sigma_i^{\text{Alt}} = \Sigma_i^p$ , and same for  $\Pi$ .

3. This problem is a nice application of the padding argument.

Show that if  $\mathbf{P} \neq \mathbf{NP}$ , then there exists a language that is in **NP** but not in **P**, and which is *not* **NP**-complete. We suspect that both *Graph Isomorphism* and *Factoring* are examples of such languages.

**Hint.** Consider PADDED-SAT,

$$\{ \phi\# |\phi|^{p(|\phi|)} \mid \phi \in \text{SAT} \}$$

---

<sup>1</sup>A good example of such a game is “chomp”: <http://www.math.ucla.edu/~tom/Games/chomp.html>

where we define the “padding” function  $p(n)$  as follows. First, fix some encoding of TMs, so we have a list of all the TMs under the sun:  $M_1, M_2, M_3, \dots$  (This can be done in a number of ways, for example with a Universal TM  $U$ .)

Next, let  $p(n)$  be the smallest  $i < \log \log n$  such that for every  $x \in \{0, 1\}^*$  with  $|x| \leq \log(n)$  the following is true

$M_i$  halts on  $x$  within  $i|x|^i$  many steps and  $M_i$  outputs 1 iff  $x \in \text{PADDED-SAT}$ .

If there is no such  $i$ , we let  $p(n) = \log \log n$ . The idea for this “strange” definition of  $p(n)$  is to have a function that grows fast enough so that PADDED-SAT is not **NP**-complete, but slowly enough to ensure that it is not in **P**. Show that this is the case.

4. This problem is an application of the *Inductive Counting Method*, which is the technique at the heart of the Immerman-Szelepcsényi theorem.

Let  $A$  be a Context-Sensitive Language. So  $A$  is generated by a grammar  $G$ , with variables  $V$  and terminal alphabet  $\Sigma$ , where all the rules are of the form  $\alpha_1 \rightarrow \alpha_2$ , with  $|\alpha_1| \leq |\alpha_2|$ .

Design a  $\text{NSPACE}(n)$  algorithm for deciding  $\bar{A}$ . Note that the existence of such an algorithm follows from the I-S theorem, since  $\text{CSL} = \text{NSPACE}(n) = \text{co-NSPACE}(n)$ . But you should give the algorithm directly.

**Hint.** Let  $T_i^n = \{x : |x| \leq n, S \xrightarrow{i} x\}$ , meaning that  $T_i^n$  contains all the strings in  $(V \cup \Sigma)^{\leq n}$  derivable from  $S$  in *at most*  $i$  many steps. Clearly,  $T_0^n = \{S\}$  for all  $n$ .

5. This last question is a *hierarchy theorem* for non-deterministic TMs.

Show that  $\text{NTIME}(n) \subsetneq \text{NTIME}(n^{1.5})$ .

**Hint.** Note that the usual diagonalization technique does not apply directly: how do we “flip” the answer of a non-deterministic TM? Recall the  $M_i$ 's from exercise 3. Let  $f : \mathbb{N} \rightarrow \mathbb{N}$  be defined as follows:

$$\begin{aligned} f(1) &= 2 \\ f(i+1) &= 2^{f(i)^{1.2}} \end{aligned}$$

Now  $D$  tries to flip the answer of  $M_i$  on *some* input in  $\{1^n \mid f(i) < n \leq f(i+1)\}$ .