1. Show that if $\text{PH}$ does not collapse, that is $\text{PH} \neq \Sigma_k$ for every $k$, then it does not have a complete problem under polynomial-time reductions.

2. Recall that a Boolean formula is an expression involving Boolean variables, and Boolean operations $\land, \lor, \neg$ (but no quantifiers $\exists, \forall$). Two Boolean formulas are called equivalent if they are on the same set of variables, and are true on the same set of truth assignments. Consider the following language

$$\text{MinFormula} = \{ \phi : \phi \text{ has no shorter equivalent formula} \}.$$ 

- Prove that $\text{MinFormula} \in \text{PSPACE}$.
- Prove that if $\text{P} = \text{NP}$, then $\text{MinFormula} \in \text{P}$.

3. (Exercise 7.2) Where is the error in the following incorrect proof that $\text{NonPATH} \in \text{NL}$?

Given an input $\langle G, s, t \rangle$, it is easy to see that there are no $st$-paths in $G$ if and only if there exists a partition of the vertices into two parts $A$ and $B$ such that $s \in A$, $t \in B$, and there are no edges going from $A$ to $B$. Let the vertices of $G$ be $u_1, \ldots, u_n$, and let $x = (x_1, \ldots, x_n) \in \{0, 1\}^n$ be such that $x_i = 0$ if $u_i \in A$ and $x_i = 1$ if $u_i \in B$. To verify that $\langle G, s, t \rangle \in \text{NonPATH}$, the verifier is going to be provided with a certificate $C$ consisting of $k$ copies of $x$, where $k$ is the number of edges in $G$:

$$C = [x, \ldots, x]^k.$$
Now as the verifier reads the $i$-th $x$, it verifies that the edge $e_i$ does not go from $A$ to $B$. It will also use $x$ to verify that $s \in A$ and $t \in B$. Once it finishes reading all of $C$ it will have verified that none of the edges $e_i$ goes from $A$ to $B$. Thus it will successfully verify that $(G, s, t) \in \text{NonPATH}$.

4. (Exercise 6.8) Prove that if in the definition of the $\text{NL}$-certifier if relax the read-once condition on the certificate-tape to an ordinary read-only tape, then it will be powerful enough so that every problem in $\text{NP}$ will have an $\text{NL}$-verifier.

5. For functions $f, g : \mathbb{N} \to \mathbb{N}$, let $\text{TimeSpace}(f, g)$ denote the class of languages that can be decided by a deterministic algorithm that runs in time $O(f(n))$ and uses space $O(g(n))$. Let $\Sigma_2 \text{Time}(n^p)$ be the class of languages $L$ such that there is a deterministic Turing Machine $M$ that runs in time $O(|x|^p)$, and

$$x \in L \iff \exists y_1 \forall y_2 \ M(x, y_1, y_2) = \text{ACCEPT}.$$ (a) Prove that

$$\text{TimeSpace}(n^{k_1}, n^{k_2}) \subseteq \Sigma_2 \text{Time} \left( n^{k_1 + k_2 \log(n)} \right).$$

Here the $\log(n)$ appears because of the simulation overlay.

(b) Prove that if $\text{NTIME}(n) \subseteq \text{DTIME}(n^t)$, then

$$\text{NTIME}(n^k) \subseteq \text{DTIME}(n^{tk}),$$

for every $k \geq 1$. Use this to show that if $\text{NTIME}(n) \subseteq \text{DTIME}(n^t)$, then

$$\Sigma_2 \text{Time} \left( n^k \right) \subseteq \text{NTIME}(n^{tk}).$$

(c) Use the above, and the non-deterministic time hierarchy theorem to conclude that

$$\text{NTIME}(n) \nsubseteq \text{TimeSpace}(n^{4/3}, n^{0.2/3}).$$

6. Prove a Non-deterministic Time Hierarchy theorem: If $T_2$ is time-constructible and $T_1(n + 1) \log T_1(n + 1) = o(T_2(n))$, then $\text{NTIME}(T_1(n)) \neq \text{NTIME}(T_2(n))$.

- First explain why the same diagonalization used for the deterministic time complexity does not work here without any changes.
- Show how to fix the argument so that it works for the non-deterministic time complexity. 
  (Hint: Use a lazy diagonalization argument: Only try to disagree at least once in an exponentially large interval. Consider a large intervals $(\ell_k, u_k]$, and for an input $1^n \in (\ell_k, u_k]$ if $n = u_k$, then deterministically run $M_k$ on $1^{\ell_k + 1}$ for an appropriate number of steps, and negate its output. If $n < u_k$, then use a non-deterministic simulation.)

---

1It is believed that one needs exponential deterministic time to simulate a non-deterministic algorithm. In particular $\text{NTIME}(n) \nsubseteq \text{DTIME}(n^c)$ for any constant $c$, or even any $c$ with $\log(c) = o(\log(n))$. However, no real separation is known between non-deterministic and deterministic time. For example, in contrast to this exercise, we do not know if $\text{NTIME}(n) \nsubseteq \text{DTIME}(n^{\log(n)})$. This exercise shows that if we somewhat limit the space to $n^{0.2/3}$ (which is still polynomially large), we can prove some separation. The best known lower-bound based on this technique is that $\text{NTIME}(n) \nsubseteq \text{TimeSpace}(n^{2 \cos(\pi/7)}, n^{0.1})$, where $2 \cos(\pi/7) \approx 1.801$. 

2