COMP 531 - Fall 2021 - Assignment 1

Due: Oct 8, 11:59pm

- In solving these questions you may only consult the lecture notes, and the Sipser book, but you need to provide citations in that case.
- Each student must find and write their own solution. Copying solutions from any source, completely or partially, allowing others to copy your work, will not be tolerated, and will be reported to the disciplinary office. You are allowed to discuss the problems with each other without revealing your solution to each other.
- You must submit your solutions as **one readable pdf file** to mycourses.
- Your grade will be based on the mathematical correctness of your solution as well as the quality of your presentation.
- In addition to the following problem, (for statistical purposes) list all the problems that you have solved from the notes. (Just provide the list. Do not write the solutions).

---

1. Show that if PH does not collapse, that is $\mathsf{PH} \neq \Sigma_k$ 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 $\wedge, \vee, \neg$ (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 MinFormula $\in \mathsf{PSPACE}$.
   - Prove that if $\mathsf{P} = \mathsf{NP}$, then MinFormula $\in \mathsf{P}$.

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

   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 $\langle G, s, t\rangle \in$ 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:

$$C = [\underbrace{x, x, \ldots, x}_{k \text{ times}}].$$

Now as the verifier reads the i-th x, it verifiers 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 $\langle G, s, t\rangle \in$ NonPATH.

4. (Exercise 6.8) Prove that if in the definition of the 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 NP will have an NL-verifier.

5. For functions $f, g : \mathbb{N} \to \mathbb{N}$, let $\mathsf{TimeSpace}(f, g)$ 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 $\Sigma_2\mathsf{Time}(n^r)$ be the class of languages L such that there is a deterministic Turing Machine M that runs in time $O(|x|^r)$, and

$$x \in L \Leftrightarrow \exists y_1 \forall y_2 \; M(x, y_1, y_2) = \textsc{Accept}.$$

   (a) Prove that
$$\mathsf{TimeSpace}(n^{k_1}, n^{k_2}) \subseteq \Sigma_2\mathsf{Time}\left(n^{\frac{k_1}{2}+k_2}\log(n)\right).$$
   Here the $\log(n)$ appears because of the simulation overlay.

   (b) Prove that if $\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(n^t)$, then
$$\mathsf{NTIME}(n^k) \subseteq \mathsf{DTIME}(n^{tk}),$$
   for every $k \geq 1$. Use this to show that if $\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(n^t)$, then
$$\Sigma_2\mathsf{Time}\left(n^k\right) \subseteq \mathsf{NTIME}(n^{tk}).$$

   (c) Use the above, and the non-deterministic time hierarchy theorem to conclude that[1]
$$\mathsf{NTIME}(n) \not\subseteq \mathsf{TimeSpace}(n^{4/3}, n^{0.2/3}).$$

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 $\mathsf{NTIME}(T_1(n)) \neq \mathsf{NTIME}(T_2(n))$.

   - 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^{ℓ_k+1} for an appropriate number of steps, and negate its output. If $n < u_k$, then use a non-deterministic simulation.)

[1] It is believed that one needs exponential deterministic time to simulate a non-deterministic algorithm. In particular $\mathsf{NTIME}(n) \not\subseteq \mathsf{DTIME}(n^c)$ 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 $\mathsf{NTIME}(n) \not\subseteq \mathsf{DTIME}(n\log(n))$. 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 $\mathsf{NTIME}(n) \not\subseteq \mathsf{TimeSpace}(n^{2\cos(\pi/7)}, n^{o(1)})$, where $2\cos(\pi/7) \approx 1.801$.