Assignment I

1. Suppose NSPACE($f(n)$)$\subseteq$ DTIME($f^k(n)$) for some constant $k > 0$ for all constructible f, can we conclude that NP=co-NP or NP$\neq$ co-NP? Justify.

2. Show that

   - If NP$\neq$ co-NP, then P$\neq$ NP.
   - If any co-NP complete problem is in NP then NP=co-NP.
   - If any problem in NP$\cap$ co-NP is NP-complete then NP=co-NP.

3. Suppose we define a language $L \in polyL$ to be $polyL$-complete if for all $L' \in polyL$, $L' \preceq_m^{\log} L$. Show that there is no complete problem in the class $polyL$.

4. Let L be an NEXP complete problem. Let c be a constant such that $L \in NTime(2^{n^c})$. Show that the language $L' = \{(x, 1^{2^{|x|}}) : x \in L\}$ is in NP. (L' consists of strings in L suffixed by an exponentially large number of 1^s). Suppose $L' \in$ P, show that L is in EXP. From this, conclude that if P=NP, then NEXP=EXP. The technique used in this proof is called *padding argument*.

5. Show that for any language L, $L \in$ Dtime(n^k) if and only if the language $L' = \{(x, 1^{|x|^k}) : x \in L\}$ is in Dtime(n). (for each string x in L, we add L padded with $|x|^k$ 1^s to L'). Using this, conclude that $P \neq$ Dspace(n).

6. Show that the problem of determining whether a graph is bipartite is in NL. (You need to use some characterization for bipartite graphs).

7. Show that the problem of determining whether a graph is strongly connected (that is does there exists a (directed) path from each vertex to the other) is NL complete. (Reduce from $s - t$ REACH).

8. If P=NP, show that every language in P except two languages are NP complete. Which are these two languages?

9. Recall that we defined polyL=$\bigcup_i$ DSPACE$(\log^i n)$. The Steve's class SC (named after Stefan Cook) is defined as SC=$\{L : L$ is accepted by some (off-line) Turing machine M that runs in time $O(n^c)$ steps using at most $O(\log^k n)$ space for some constants k and $c\}$. Why is SC not the same as polyL$\cap$ P?

10. Given graph (G_1, G_2), Show that the problem of deciding whether a graph G_1 contains G_2 as an induced subgraph is NP-complete.

11. Recall that in the class we characterized NP as NP=$\{L$: there exists a polynomial time verifying algorithm A such that $x \in L$ if and only if there exists a polynomially (length) bounded y such that $A(x, y) = 1\}$. Show that co-NP=$\{L$: there exists a polynomial time verifying algorithm A such that $x \in L$ if and only if for all polynomially bounded y, $A(x, y) = 1\}$.

12. Suppose we define the complexity class extending NP as follows: $\Sigma_2 = \{L$: there exists a polynomial time verifying algorithm $A(x, y, z)$on three inputs and constants k_1 and k_2 such that $x \in L$ if and only if there exists y such that for all z satisfying $|y| \leq |x|^{k_1}$ and $|z| \leq |x|^{k_2}$, $A(x, y, z) = 1\}$. We define the class $\Pi_2 = \{L : \overline{L} \in \Sigma_2\}$. Derive a characterization as in the above problem for Π_2. Show that NP$\cup$ co-NP $\subseteq \Sigma_2$ and NP$\cup$ co-NP $\subseteq \Pi_2$

13. Suppose we try to give the following characterization for NL: We say that a language L is log-space verifiable if there exists an off-line Turing machine M that runs using $O(\log n)$ work tape space having a read only input tape supplied with two inputs – the first one being the string x whose membership in L is to be determined and the second, a certificate y whose length is polynomially bounded by the length of x such that (x, y) is accepted by M if and only if $x \in L$. Show that the class of log-space verifiable languages is precisely NP. (It turns out that the ability to read y multiple times is causing the problem. If M is allowed to read each bit of y only once, then we get NL.)