Assignment 3: Computational Complexity

Due date: Thursday 21 March *before* the class starts
(i.e., it *cannot* be returned during class)
10 extra credit if returned by Sunday 16 March 11:59pm

1. (5) Assuming the time hierarchy theorem (proved in class and Theorem 4.8 of the book) show that $P \neq EXP$. Hint: We proved that in class.

2. (10) Assuming the space hierarchy theorem and assuming Savitch's theorem prove a hierarchy theorem (similar to Theorem 4.8) for non-deterministic space complexity. Note: the relation between the two functions in your theorem does not need to be as tight as those of Theorem 4.8. However, your theorem should be strong enough to, e.g., can show that $NSPACE(n^2) \neq NSPACE(n^5)$.

3. (5) Assuming the space hierarchy theorem and assuming Savitch's theorem prove that $NL \neq NSPACE$.

4. (10) Prove that TQBF problem is in $PSPACE$. Hint: since the number of quantifiers is not fixed, we cannot use a fixed code that has a bunch of nested loops. Instead, try using recursive procedures (a la the "bad" poly-space algorithm for SAT that we saw in class). Pay attention to the details both in the description of your algortihm as well as the analysis of its space complexity. Hint 2: the proof is written in the book, but you cannot copy/paste the proof, and if your proof uses different ideas from the book, you could get extra credit.

5. (10) Read the proof of Theorem 4.13 from the book. Note that the proof heavily relies on the internal steps of the Savitch's theorem that was covered in class. You do not need to write the proof again here (even though you should understand all the details and if you could not let me know).

   Prove the last line of the theorem (which is left to the reader). Namely, show that the formula that we get at the end can be converted into a "nice" form that indeed looks like a TQBF instance.

6. (10) Let X, Y be two languages in logarithmic space (i.e. $X \in L$ and $Y \in L$). Show that unless Y is trivial (i.e. empty or all strings) it holds that $X \leq_p Y$. Then solve question 4.3 of the book.

   This question shows why the existence of Karp reductions between log-space languages is not interesting. Def 4.16 defines a more restricted notion of reductions that is useful to denote the "hardest" problems in NL. This notion of reductions is used in Theorem 4.18 to show that the problem PATH is NL-complete (i.e. if we can solve PATH in logarithmic space, we can solve all of NL in log-space—note that we already know that $NL \subseteq SPACE(\log^2 n)$.)

7. (10) Show that if $P = NP$ then we can solve all problems in the polynomial time hierarchy in polynomial time (i.e. $PH = P$). Hint: This is the second part of Theorem 5.4 (so you can read it, understand it and write it in your own words), but if you can write the proof based on the discussion of our own class, that might come with extra credit.

It is interesting to note here that PH is the largest class of problems that we can solve in polynomial time if $P = NP$.

8. (10) Why the proof of the previous question does not work when we try to show that:

$$P = NP \Rightarrow TQBF \in P$$

Hint: Suppose we start from a polynomial time algorithm that solves SAT and try to solve any problem in Σ_i for any constant i in polynomial time by "eliminating" the quantifiers one by one (i.e. induction). How does the running time of the algorithm increase upon removing every quantifier? What happens to the running time of the algorithm when we want to eliminate more than a constant number of quantifiers?

9. (5) Read Definition 4.16 and make sure you understand it (you do not need to write it). Give a log-space reduction between CLIQUE and VERTEX-COVER problems.

10. Extra credit (10). Question 4.6 from the book (for this you need to solve question 9 first!)