CS46 Homework 12

This homework is due at 11:59pm on **Wednesday, April 27**. Note the unusual deadline! This is a **14-point** homework.

For this homework, you will work with a partner. It's ok to discuss approaches at a high level, but most of your discussions should just be with your partner. The only exception to this rule is work you've done with another student *while in lab*. In this case, note who you've worked with and what parts were solved during lab. Your partnership's write-up should be your own: do not share it, and do not read other people's write-ups. Please refer to the course webpage or ask me any questions you have about this policy.

1. **Closure properties.**

   (a) Prove that P is closed under concatenation.

   (b) Prove that P is closed under complement.

   (c) Prove that NP is closed under union.

   (d) Prove that NP is closed under concatenation.

2. **Is (almost) everything NP-complete?**

   Show that if P = NP, then every language $A \in$ P is NP-complete except $A = \emptyset$ and $A = \Sigma^*$.

3. **Vertex cover and independent set, again.**

   Recall that a **vertex cover** in a graph G is a subset of vertices where every edge of G has at least one endpoint in the subset.

   $$\text{VertexCover} = \{\langle G, k\rangle \mid G \text{ has a } k\text{-node vertex cover}\}$$

   Theorem 7.44 says that VertexCover is NP-complete .

   An **independent set** in a graph G is a subset of vertices with no edges between them.

   $$\text{IndependentSet} = \{\langle G, k\rangle \mid G \text{ contains an independent set of } k \text{ vertices}\}$$

   We will show that IndependentSet is NP-complete .

   (a) Prove that IndependentSet $\in$ NP.

   (b) Prove that IndependentSet is NP-hard .
   (Hint: reduce from VertexCover . This is *not* the same direction you did in lab, but you might be able to use the same idea as the core of your reduction.)

4. **An NP-complete problem related to regular expressions.**

   A regular expression is ***-free** (pronounced "star-free") if it does not include any Kleene stars, so for example the regular expression "$(1 \cup 0)00$" is *-free but "$0^*(1 \cup 11)$" is not *-free.

   Consider the language:

   $$L = \{\langle R_1, R_2\rangle \mid R_1 \text{ and } R_2 \text{ are *-free regular expressions and } L(R_1) \neq L(R_2)\}$$

   You will prove that L is NP-complete , using a reduction from Satisfiability .

(a) Show that $L \in \mathrm{NP}$ by giving a deterministic polynomial-time verifier and describing the certificate ("extra information") strings it uses to check membership in L. (Make sure your verifier fits the definition of "verifier" correctly — in particular, you need to make sure it can't be tricked into accepting by a bad certificate!)

(b) Given a formula ϕ in conjunctive normal form, write a regular expression that matches the language:

$$\{w \mid w \text{ encodes a truth assignment for } \phi\}$$

(c) Given a set of n literals $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$, consider the clause:

$$c = \alpha_1 \vee \alpha_2 \vee \cdots \vee \alpha_n$$

Write a regular expression that matches the language:

$$\{w \mid w \text{ encodes a truth assignment which does } \textit{not} \text{ satisfy } c\}$$

(d) Given a formula ϕ in conjunctive normal form, write a regular expression that matches the language:

$$\{w \mid w \text{ encodes a truth assignment which does } \textit{not} \text{ satisfy } \phi\}$$

(Use part (c).)

(e) Use parts (b) and (d) to give a polynomial-time reduction from SATISFIABILITY to L. Conclude that L is NP-COMPLETE . (**Hint:** If you want to, you may assume throughout this problem that formulas in SATISFIABILITY are always in conjunctive normal form.)

5. (**extra credit**) Show that if $\mathrm{P} \cap \text{NP-HARD} \neq \emptyset$, then $\mathrm{P} = \mathrm{NP}$.

6. **(extra credit)** Does $\mathrm{coNP} = \mathrm{NP}$? Support your answer with a proof.