Computer Science 468/568
Homework #2, due in class Thursday, February 1, 2018.

Reading: Chapter 3 (Diagonalization).

1. (10 points) Please list any persons (including course staff) you talked with about this assignment and cite any resources (other than the textbook) you consulted in connection with this assignment (with enough information that another person could find the materials you cite).

2. (30 points) Define the language H to be all pairs $\langle \alpha, x \rangle$ such that M_α halts in a finite number of steps on input x. Prove that H is NP-hard but not NP-complete.

3. (30 points)

   (a) If $x, y \in \{0,1\}^*$ we define $s(x,y)$ to be the set of strings that can be obtained by arbitrarily interleaving the symbols of x and y, preserving the relative order of symbols in each string. Thus, 1001011 is an element of $s(1101, 001)$ but 1000111 is not. If $L_1, L_2 \subseteq \{0,1\}^*$ then define $s(L_1, L_2)$ to be the set of all strings z such that $z \in s(x,y)$ for some $x \in L_1$ and $y \in L_2$. Prove that if L_1 and L_2 are in NP, then so is $s(L_1, L_2)$.

   (b) If $L_1, L_2 \subseteq \{0,1\}^*$ are NP-complete, must $L_1 \cap L_2$ also be NP-complete?

4. (30 points) We inductively define the set of uc-expressions over an alphabet Σ and the set $D(E)$ of strings denoted by a uc-expression E as follows.

   (a) Each $\sigma \in \Sigma$ is a uc-expression, with $D(\sigma) = \{\sigma\}$.

   (b) If E_1 and E_2 are uc-expressions, then so are $u(E_1, E_2)$ and $c(E_1, E_2)$, where

   $$D(u(E_1, E_2)) = D(E_1) \cup D(E_2)$$

   and

   $$D(c(E_1, E_2)) = D(E_1) \cdot D(E_2)$$

   where for two sets of strings S and T, $S \cdot T$ is the set of all strings xy such that $x \in S$ and $y \in T$.

   Define Q to be the set of all pairs $\langle E_1, E_2 \rangle$ such that $D(E_1) = D(E_2)$. Classify the computational complexity of Q as accurately as you can.

5. (30 points) This problem is required of students enrolled in CPSC 568, but not students enrolled in CPSC 468. Students enrolled in CPSC 468 may do the problem and have it graded, but will receive no extra credit for it.

   The text gives the following definition: if L_1 and L_2 are in NP and f is a polynomial time reduction of L_1 to L_2, then f is *parsimonious* if and only if for all $x \in \{0,1\}^*$, x and $f(x)$ have the same number of certificates. Think about how to make this definition more detailed and formal, describe the issues you encounter, and then give a more detailed and formal definition. For your definition, is the composition of two parsimonious reductions necessarily parsimonious?

   Construct a polynomial time reduction f from SAT to 3SAT such that the number of satisfying assignments of ϕ is equal to the number of satisfying assignments of $f(\phi)$.