Assignment 4

Due Friday April 5th, 2019

Updated March 29th. The text of Q4 contained a mistake, and has been corrected in this version.

1. [10 marks] Short answers about Turing machines

Answer each of the following questions in *one or two* sentences.

(a) [2 marks] Explain the difference between a Turing recognizable and a Turing decidable languages.

(b) [2 marks] Explain the Halting problem and its significance.

(c) [2 marks] Consider the set of languages recognizable by a determnistic Turing Machine. Now consider a multi-tape Turing machine, that is, a Turing machine with multiple taples. Could a multi-tape Turing machine possibly recognize more languages than a single-tape Turing Machine? Why or why not?

(d) [2 marks] Under what circumstance does a non-deterministic Turing machine output `accept`?

(e) [2 marks] In class we mentioned that the set of all possible Turing Machines is countably infinite, and that the set of all languages is uncountably infinite. Use these facts to reason why there *must* be languages that cannot be recognized by a Turing machine.

2. [6 marks] Complexity Classes

For each of the complexity classes covered in the lectures, i.e., **P**, **NP**, **NP**-complete, **NP**-hard, **BPP** and **BQP**, state whether the problems below are known to be in that class or not. For each positive answer (i.e., *yes, problem ____ is in class ____.*), give the name of an algorithm from that class that solves it:

(a) [3 marks] **PRIMES**, the problem of determining if a given integer is a prime number.

(b) [3 marks] **FACTOR**, the problem of determining if an integer contains a factor less than some bound b.

3. [4 marks] Boolean Satisfiability

(a) [2 marks] **Boolean Satisfiability (SAT)**. Consider the following boolean equation:

$$((x_1 \wedge x_2) \vee (x_3 \wedge \neg x_4)) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_2 \vee \neg x_3)$$

Is this equation satisfiable? If so, assign the variables to satisfy the equation. If not, explain why not.

(b) [2 marks] Recall **SAT** was the first problem proven to be **NP**-complete (by Cook and Levin). What would the consequence be of finding an efficient algorithm to solve **SAT**?

4. [10 marks] Prove NP-Hardness Pt 1.

We say a boolean expression ϕ always outputs the same result if $\phi(V)$ either always outputs True, or always outputs False for all possible variable assignments V. For example, $\phi = x_1 \vee \overline{x_1}$ always outputs the same result (i.e., True) for all possible assignments of x_1. Now consider the set of all such Boolean expressions:

ALWAYS $= \{\langle\phi\rangle$: ϕ is a Boolean expression that always outputs the same result for any input$\}$

Prove ALWAYS is NP-Hard. Hint: SAT is NP-complete.
NOTE: *A previous version of this assignment incorrectly asked you to prove AWLAYS is NP-complete.*

5. [10 marks] Prove NP-Completeness

Let S be a set of integers. Recall SUBSETSUM is the problem of deciding whether there is some subset of S which sums to zero. Now consider the related problem PARTITION which asks whether S can be partitioned (i.e., split) into two subsets such that they have the *same* sum. For example, $\{1, 2, 3, 4\}$ contains such a partition: $\{1, 4\}$ and $\{2, 3\}$.

Given that SUBSETSUM is **NP**-complete, prove PARTITION is **NP**-complete. **Hint**: Suppose the elements of a set S sum to some integer s. Define a new set $S' = S \cup \{-s\}$ and use it in your proof.

6. [10 marks] Prove NP-Hardness Pt. 2

Recall the Halting problem:

$$\text{HALT} = \{\langle M, w\rangle : M \text{ halts on input } w\}.$$

Prove the Halting problem is NP-Hard.