TheoryPrep **(Due: 7/11/20)**

Assignment #6: Reductions

Name: Student name(s)

We've taken some problems/solutions from the course texts and CS 121 material. Some problems are marked as (Challenge) or (Additional Problems). Do the rest first, and if you have time, return to these problems. It's **much more important** to have a rigorous understanding of the core problems and how to prove them than to simply finish all the additional problems. This week's pset will have a few longer proofs.

Problem 1: Uncomputability Reductions

Learning goal: **This is the main problem on the pset.** It starts with some shorter questions to build intuition for uncomputability reductions and then has two longer problems that are designed to help you understand uncomputability reductions.

(a) Your friend wants to prove that some function F is uncomputable and they've heard that they can do something called "reducing to $HALT$." They've never seen a reduction before. Explain at a high level what they should do. Make sure to include the steps they need to do and what the proof implies.

(b) (Additional Problem) An uncomputability reduction could be considered an application of the contrapositive. Explain why this is true (think about the $HALT$, $HALTONZERO$ example).

(c) Let $EVENLENGTH : \{0,1\}^* \to \{0,1\}$ be the function that takes in a turing machine M and evaluates to 1 if and only if every input halts and gives an even-length output. Prove by reduction that $EVENLENGTH$ is uncomputable (we suggest you reduce to $HALT$ or $HALTONZERO$). You can use \begin{lstlisting} to write code.

(d) Prove Rice's Theorem. To review, Rice's Theorem says that any $F : \{0,1\}^* \to \{0,1\}$ that is semantic and nontrivial is uncomputable.

Problem 2: Polynomial Time Reductions (Additional Problem)

Learning goal: This section has some brief questions to help you build intuition for polynomial time reductions and then some reduction problems. This entire section is optional, and you should prioritize building a strong grasp of uncomputability reductions in the previous section first.

(a) (Additional Problem) Suppose that for functions $F, G : \{0,1\}^* \to \{0,1\}$, F reduces to G. Complete each of the following statements to form the strongest true statements (a statement A is stronger than B if $A \Rightarrow B$–another way to think about the strongest statement is that it's the most you can assert).

1. If G is polynomially time computable, then F
2. If F cannot be computed in polynomial time, then G
3. We'd say that F is (easier/weakly easier/equal/weakly harder/harder) compared to G.
4. If F cannot be computed in exponential time, then G
5. If G can be computed in linear time, then F

(b) (Additional Problem) In 121, we think any algorithm that runs in polynomial time is efficient. Thus, we wanted our definition of a polynomial time reduction to satisfy "F reduces to G implies that if G is polynomially time computable then so is F" (which it does by Solved Exercise 13.1 on page 15 of the reading). Why does R need to be polynomially time computable in order for this property to hold? An intuitive explanation will suffice (this question exemplifies how to interrogate a definition–like we learned last week).

(c) (Additional Problem) Now to do a reduction. In the readings/video we talked about the 3SAT problem, but we might wonder what happens if we have 4 variables in each clause. This motivates the 4SAT problem. $4SAT : \{0,1\}^* \to \{0,1\}$ takes in the encoding of a 4CNF (like $(x_0 \vee \bar{x_2} \vee x_3 \vee \bar{x_4}) \wedge (\bar{x_0} \vee x_1 \vee \bar{x_2} \vee x_3)...$) and returns 1 if and only if it is satisfiable. You'll prove that interestingly, even though we added more variables, it doesn't become more difficult; prove that $4SAT$ reduces to $3SAT$ (it may be helpful to think of the 01EQ, 3SAT reduction in the reading and how it used slack variables).

(d) (Additional Problem) The function $VERTEXCOVER : \{0,1\}^* \to \{0,1\}$ on input a graph G and number k returns 1 if and only if there is a vertex cover of size at most k of the graph. A vertex cover is a set S of vertices such that every edge has at least one endpoint in S. $HALFCOVER : \{0,1\}^* \to \{0,1\}$ also takes in a graph G and number k and returns 1 if and only if there is a set of vertices S of size at most k such that at least half the edges have an endpoint in S. It seems that HALFCOVER should be an easier problem , but your task is to prove that VERTEXCOVER reduces to HALFCOVER.

(e) (Challenge Problem) Reduce 3SAT to VERTEXCOVER. It may help to think about the reduction from 3SAT to ISET from the video.

Problem 3: Review

Learning goal: The goal for this section is to reflect on how you can improve your proof skills! We're probably going to use this format for Problem 4 for the rest of the course since reflecting and fixing past proofs is a great exercise–if you'd prefer more review problems from previous weeks please let us know (in a response to Problem 5 or on the Piazza).

(a) Select the problem from a previous week that you think you would learn the most from redoing. Copy the problem and your instructor's feedback below, rewrite the proof, and add a short reflection on the changes you made.

(b) (Additional Question) Choose another problem you received feedback on and copy the problem and feedback you received, rewrite the proof, and write a sentence about why you made your changes (so do Part A again) OR solve a problem you left blank from a previous week (try to pick from a week you felt least secure on).

Problem 5: Logistics

Purpose: This helps us make sure the course is going at the right speed!

(a) How long did you spend on the videos and readings this week?

(b) How long (including time in problem sessions) did you spend on this problem set?

(c) Do you have any feedback about the course in general (did the videos and readings sufficiently prepare you for the problem set)?