CS:4330 Theory of Computation **Spring 2018**

HW 6: Undecidable Languages

Instructor: Haniel Barbosa, TA: Shantanu Agarwal *Due date: April 1, 2018*

1. (15 points) Let $T = \{\langle M \rangle \mid M \text{ is a TM that accepts } w^R \text{ whenever it accepts } w\}$. Show that T is undecidable.

2. (20 points) Consider the problem of determining whether a single-tape Turing machine ever writes a blank symbol over a nonblank symbol during the course of its computation on any input string. Formulate this problem as a language and show that it is undecidable.

3. (20 points) A *useless state* in a Turing machine is one that is never entered on any input string. Consider the problem of determining whether a Turing machine has any useless states. Formulate this problem as a language and show that it is undecidable.

4. (30 points) We can use Rice's theorem to show that a language is undecidable. For example:

   Consider the language $\mathit{INFINITE}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } \mathcal{L}(M) \text{ is an infinite language}\}$. It satisfies the two conditions of Rice's theorem. First, it is nontrivial because some TMs have infinite languages and others do not. Second, it depends only on the language. If two TMs recognize the same language, either both have descriptions in $\mathit{INFINITE}_{TM}$ or neither do. Consequently, Rice's theorem implies that $\mathit{INFINITE}_{TM}$ is undecidable.

   Use Rice's theorem to prove the undecidability of the following languages:

   (a) $\{\langle M \rangle \mid M \text{ is a TM and } 1011 \in \mathcal{L}(M)\}$

   (b) $\mathit{ALL}_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } \mathcal{L}(M) = \Sigma^*\}$

5. (15 points) [Gödel's Incompleteness Theorem] One of the most important results of 20th century logic (and arguably of all 20th century mathematics) was the discovery that there are some true mathematical statements that cannot be proven using the standard axioms of mathematics. This 1931 result by Kurt Gödel is known as Gödel's Incompleteness Theorem. In this problem, you will prove one version of Gödel's Theorem using what we know about Turing machines, 79 years too late to revolutionize modern mathematics.

   (The actual proof given by Gödel is a little more complicated, since Turing Machines hadn't been invented yet, but it relies on the same basic notion of diagonalization, and the two results are intimately related.)

   Some preliminaries: we say that a system of mathematics is *consistent* if there exists a special proof-checking Turing machine M^* that can verify or reject any proof that it is given in the language of that system. In other words, there exists a machine M^* that decides the language

   $$\mathit{PROOFS} = \{\langle P, T \rangle \mid P \text{ is a valid proof of statement } T\}$$

   Assume that our system of mathematics is *consistent*, so that we can check algorithmically whether a proof is correct. Assume for simplicity's sake that any statement T in our system of mathematics is either true or false — exactly one of T or $not(T)$ is true. For this question, you can express a mathematical statement in the usual way using English, rather than encoding it into any special logical language.

   (a) Suppose for a contradiction that every true statement T_i has a proof P_i. Using this assumption, give a Turing machine decider for the language

   $$\mathit{TSTMTS} = \{\langle T \rangle \mid T \text{ is a true statement}\}$$

(b) Use the previous result to construct a Turing machine that solves the halting problem.

(c) Conclude that there exists a true statement that does not have a proof, proving Gödel's Theorem.