CSE 431 – Theory of Computation. Spring, 2014. Instructor: James R. Lee

ASSIGNMENT 3. Due Thursday, May 1st, in class (or via email to cse431-staff@cs before class starts)

1. Describe a pair of distinct Turing machines A and B such that when started on any input, A outputs $\langle B \rangle$ and B outputs $\langle A \rangle$.

2. Suppose that A is a language of Turing machine descriptions that satisfies two properties:
   a) Membership in A depends only on the **language** of a Turing machine, i.e., if $L(M) = L(N)$ then $\langle M \rangle \in A \Leftrightarrow \langle N \rangle \in A$.
   [Recall that $L(M) = \{ w \in \Sigma^* : M \text{ accepts } w \}$.
   As an example, the language $A = \{ \langle M \rangle : M \text{ accepts the empty string} \}$ depends only on $L(M)$ (the language M accepts). On the other hand, the language $B = \{ \langle M \rangle : M \text{ has 12 states and accepts the empty string}\}$ does not depend just on the language of M. It also depends on $M's$ implementation.]
   b) A is **non-trivial** in the sense that does not contain **all** Turing machine descriptions, but it does contain at least one TM description.

   Use the recursion theorem to prove that, if these two properties are true, A must be undecidable. [In other words, **every** interesting question about languages of TMs is undecidable!]

3. For any natural number $q > 1$, consider the set $\mathbb{Z}_q = \{0,1, \dots, q-1\}$, and let $\mathrm{Th}(\mathbb{Z}_q, +, \times)$ be the set of true sentences using quantifiers, logical operators, and the operations $+$ and $\times$, where the two operations correspond to addition and multiplication modulo q. Show that for every $q > 1$, the theory $\mathrm{Th}(\mathbb{Z}_q, +, \times)$ is **decidable.** In other words, there exists a Turing machine that takes a sentence and accepts when the sentence is true and rejects when the sentence is false.

   [To clarify, the goal is to show that if one is given a sentence like

   $$\forall y \exists p \forall x (x = py \rightarrow \exists r\ (r = x + p))$$

   then we can decide whether it's true using a Turing machine. If the quantifiers quantify over $\mathbb{N}$, we saw the problem is undecidable, but now they quantify over $\mathbb{Z}_q$.]

OPTIONAL PROBLEM (You may do this problem for extra credit, OR you can do it instead of the first three problems!)

An **oracle** for a language L is an external device that can report whether any string w is a member of L. An **oracle Turing machine** is a modified Turing machine that has the additional capability of querying such an oracle. We write M^L to describe an oracle Turing machine that has an oracle for the language L.

We say that a language A is **Turing-reducible** to a language B, written $A \leq_T B$, if there is a Turing machine M such that M^B decides A. (In other words, A can be decided with an oracle for B.) As a warm up, you might want to confirm the following two facts:

1) If $A \leq_T B$ and B is decidable, then A is decidable.
2) $A_{TM} \leq_T HALT_{TM}$ (in other words, the acceptance language for Turing machines can be decided if we have an oracle for the halting problem)

Now here's the problem: Show that there are two languages A and B such that $A \nleq_T B$ and $B \nleq_T A$. In other words, A cannot be solved with an oracle for B and B cannot be solved with an oracle for A.