Computational Models, Spring 2015 Exrecise #5

Reductions

1. Show that the following language is undecidable:

$$L_{01} = \{< M > \mid M \text{is a TM that accepts the string } 01\}$$

We will show a mapping reduction from A_{TM} to L_{01}: We map the pair $(< M >, w)$ to $< M_w >$ where M_w is a TM that behaves as follows: If M accepts w then it accepts everything; otherwise it accepts nothing.

Note that we must also map inputs not of the form $< M, w >$ somewhere.

2. Prove or show a counter-example: If $A \leq_M B$ and B is a regular language, does that imply that A is a regular language?

No. For example, define the languages $A = \{0^n 1^n \mid n \geq 0\}$ and $B = \{1\}$, both over the alphabet $\Sigma = \{0, 1\}$. Define the function $f : \Sigma^* \to \Sigma^*$ as
$f(w) = 1$ if $w \in A$ and $f(w) = 0$ if $w \notin A$.
Observe that A is a context-free language, so it is also Turing-decidable. Thus, f is a computable function. Also, $w \in A$ if and only if $f(w) = 1$, which is true if and only if $f(w) \in B$. Hence, $A \leq_M B$. Language A is nonregular, but B is regular since it is finite.

3. A useless state in a Turing machine is one that is never entered on any input string. Consider the problem of determining whether a state in a Turing machine is useless:

$$\text{USELESS}_{TM} = \{< M >, q \mid q \text{ is a useless state in the TM } M\}$$

Show that USELESS_{TM} is undecidable.

We show that USELESS_{TM} is undecidable by reducing E_{TM}. Suppose that USELESS_{TM} is decidable and that TM R decides it. Note that for any Turing machine M with accept state q_{acc}, q_{acc} is useless if and only if $L(M) = \emptyset$. Thus, since TM R solves USELESS_{TM}, we can use R to check if q_{acc} is a useless state to decide E_{TM}.

Specifically, below is a TM S that decides $E_{\{}TM\}$ by using the decider R for USELESS_{TM} as a subroutine:
$S =$ On input $< M >$, where M is a TM: Run TM R on input $< M >, q_{acc}$, where q_{acc} is the accepting state of M. If R accepts, accept. If R rejects, reject.

However, since we know that E_{TM} is undecidable, there cannot exist a TM that decides USELESS_{TM} .

4. Show that S_{TM} is undecidable, where

$$S_{TM} = \{< M > \mid M \text{ is a TM that accepts } w^R \text{ whenever it accepts } w\}$$

The basic idea is to reduce A_{TM} to S_{TM}. Let us assume that S_{TM} is decidable. For convenience, we will denote the TM's that accept S_{TM} and A_{TM} by S and A, respectively. The TM A receives $< M >, w$ as input. It rejects the input if the syntax is incorrect. Otherwise, it constructs a TM M_2 from M and w, and feeds $< M_2 >$ to S. We describe how to construct TM M_2 from M and w below. The TM A accepts $< M >, w$ if and only if S accepts $< M_2 >$.

The TM M_2 works as follows. It scans the input x and accepts if $x = 01$. Otherwise, it simulates M on w. If M accepts w, then M_2 accepts irrespective of what the input for M_2 was. Note that the description of M and w is hard-coded into M_2; it is not fed as input to M_2. By construction, M_2 recognizes the language $\{01\}$ if M does not accept

w; otherwise, M_2 recognizes Σ^*. Recall that $L(M_2)$ is the language recognized by M_2. When the language $L(M_2)$ is $\{01\}$, if $w \in L(M2)$, then w begins with 0 and ends with 1, so $w^R \notin L(M_2)$; thus, $< M_2 > \notin S_{TM}$ in this case. On the other hand, when the language $L(M2)$ is Σ^*, if $w \in L(M_2)$, then $w^R \in L(M2)$ because $L(M_2)$ contains all strings over $\Sigma = \{0, 1\}$; thus, $< M_2 > \in S_{TM}$ in this case. Therefore, because S decides S_{TM} by assumption, $< M_2 >$ will be accepted by S if and only if M accepts w. This means that TM A will accept $< M >, w$ if and only if S accepts $< M_2 >$. By assumption, S_{TM} is decidable. This implies A_{TM} is decidable, which is a contradiction.

5. Fermats Last Theorem, until recently one of the most famous unproved statements in mathematics, asserts that there are no integer solutions (x, y, z, n) to the equation $x^n + y^n = z^n$ satisfying $x, y > 0$ and $n > 2$. Show that if the halting problem would be decideable it would allow you to determine whether the statement is true or false.

   We can construct a Turing Machine T that enumerates 4-tuples (x, y, z, n) of positive integers (with $n > 2$). After each 4-tuple is enumerated, the machine T checks if $x^n + y^n = z^n$ (computing the relevant quantities) and checks that $n > 2$. If both checks succeed then the machine halts, otherwise it moves on to the next 4-tuple. Note that this machine ignores its input (or can be assumed to run on the empty string). Furthermore, since there infinitely many 4-tuples of integers, machine T halts if and only if there exists a 4-tuple (with $n > 2$) such that $x^n + y^n = z^n$. In other words, T halts on the empty string if and only if Fermats Last Theorem is false. If we had a solution TH to the halting problem, then we could feed the pair $< T >, \epsilon$ to TH, and get the output. If the output was 1 then T halts and Fermats Last Theorem is false. If the output was 0 then T doesnt halt, and Fermats Last Theorem is true.

6. Show that the problem of deciding whether hamsters are smarter than humans is decidable. More precisely, show that there is a TM that will print "YES" if hamsters are smarter than humans and "NO" otherwise.

   A problem is decideable if there exists a TM that decides it. However, we do not need to construct one machine. We simply have to prove the existence of a machine that decides it.

   Take a TM M_1 that always prints "YES" and halts and a TM M_2 that always prints "NO" and halts. We have two cases: If hamsters are smarter than humans then M_1 does the job, If hamsters are not smarter than humans then M_2 does the job. In either case there is a TM that does the job and the problem is decidable (though we are not sure which one of the two TM actually decides it).

7. For the language below, determine whether or not it is in $\mathcal{R}$. If it is in $\mathcal{R}$, describe a Turing machine that decides it. If it is not, show this using a reduction.

$$L = \{< M > \mid M \text{ is a TM that decides the halting problem}\}$$

   There are no TM's that decide the halting problem. Thus, $L = \emptyset$ and it is decidable just make a TM that transitions to its reject state on its first move regardless of its input.