Aaron Hampshire
Homework #5
CS 311, Spring 2014

1. **Show that the language ALL_{TM} is undecidable**

$$\boldsymbol{ALL_{TM} = \{\langle M\rangle \mid M \text{ is a } TM \text{ and } L(M) = \Sigma^*\}}$$

ALL_{TM} is undecidable by Rice's Theorem. It satisfies Rice's Theorem because:

(a) ALL_{TM} is non-trivial because there is at least one TM that will accept all inputs, and at least one TM that rejects every input.
(b) It depends only on the language. If two TM's recognize the same language, either they both are in ALL_{TM} or neither are. Explicitly, suppose we have two machines, R and S :

$$\langle R\rangle \in ALL_{TM} \iff L(R) = \Sigma^* = L(S) \iff \langle S\rangle \in ALL_{TM}$$

Therefore, Rice's theorem states that ALL_{TM} is undecidable.

2. **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.**

$$UL_{TM} = \{\langle M, q\rangle \mid M \text{ is a TM and } q \text{ is a useless state in } M\}$$

[Proof by Contradiction]
Let's assume that TM R decides UL_{TM}. We construct TM S to decide $HALT_{TM}$, with S operating as follows:

S = "On input $\langle M, q\rangle$:

1. Construct a new Turing Machine, T :
   $T =$ "On input string x:
   1. Run x on M.
   2. If M halts, T enters the state, q_{halt}.
2. Run R on $\langle M, q_{halt}\rangle$
3. If R accepts, *accept*; if R rejects, *reject*.

Because S decides $HALT_{TM}$ and $HALT_{TM}$ is known to be undecidable, a contradiction is reached. Therefore, UL_{TM} is also undecidable.

3. **If $A \leq_m B$ and B is a regular language, does this imply that A is a regular language? Why or why not?**

No. As a counterexample, let $A = \{0^n1^n \mid n \geq 0\}$ and let $B = \{1 \mid n \geq 0\}$. A is one of our context-free language archetypes and B is clearly regular. However, $A \leq_m B$ because we can define a function, f, that reduces A to B such that

$$w \in A \iff f(w) \in B$$

f would simply output 1 if w was in A, and f would output 0 if w was not in A.

4. **Prove that $\leq_m$ is a transitive relation.**

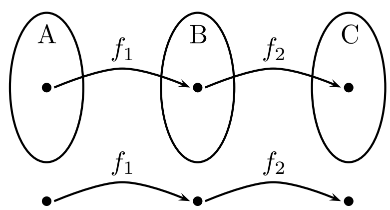
Let A, B, and C be languages such that $A \leq_m B$ and $B \leq_m C$. For this to be true, there must be two functions, f_1 and f_2 to reduce A to B and B to C, respectively.

By definition of mapping reductions:

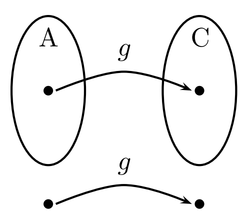
$$w \in A \iff f_1(w) \in B$$

$$z \in B \iff f_2(z) \in B$$

Pictorially, this looks like the following figure.



We can also create a composite function, g, such that $g(w) = f_2(f_1(w))$. This function first computes the output of $f_1(w)$ on a TM and uses that output, z, as the input for f_2. Pictorially, g looks like the following figure.



Because g creates a reduction between A and C, we can conclude that $A \leq_m C$, proving that $\leq_m$ is transitive.

5. **Prove that if A is Turing-recognizable and $A \leq_m \overline{A}$, then A is decidable.**

If $A \leq_m \overline{A}$, there must be some function, f, for an input w, such that $w \in A \iff f(w) \in \overline{A}$. To define the function, let's assume the strings x and y, such that $x \in A$ and $y \notin A$:

$$f(w) = \begin{cases} y & \text{if } w \in A \\ x & \text{if } w \notin A \end{cases}$$

Because of the nature of this function, the same function would also reduce $\overline{A}$ to A, $\overline{A} \leq_m A$.

We know from Theorem 5.28:

If $A \leq_m B$ and B is Turing-recognizable, then A is is Turing-recognizable.

Applying Theorem 5.28 to our problem, we can say that $\overline{A}$ must also be Turing-recognizable because of the mapping reduction, $\overline{A} \leq_m A$.

Finally, Theorem 4.22 states:

A language is decidable exactly when both it and its complement are Turing-recognizable.

Because both A and $\overline{A}$ are recognizable, we can conclude that A must also be decidable.