Weekly exercises (with proposed solutions) on Undecidability

Exercise 1 Let $\overline{A}$ denote the *complement* of A, that is, $\overline{A} = \{w \mid w \notin A\}$.
Show that if A is a decidable language, then so is $\overline{A}$.

Solution proposal Since A is decidable, a decider M_A for A exists. Create $M_{\overline{A}}$:
"On input w:

(1) Simulate M_A on w.

(2) If M_A accepts, *reject*.

(3) If M_A rejects, *accept*."

Exercise 2 For each language, show that it is undecidable by giving a reduction from $HALT$.

a) $L_1 = \{\langle M\rangle \mid \text{TM } M \text{ writes a \$ on the tape on every input}\}$

b) $L_2 = \{\langle M\rangle \mid \text{TM } M \text{ writes a \$ on the tape on input } 010\}$

c) $L_3 = \{\langle M\rangle \mid \text{There is no } y \text{ such that TM } M \text{ writes a \$ on the tape on input } y\}$

Solution proposal For L_1 and L_2 the same reduction as given in the lecture does the trick, since M' writes a $ on every input iff M halts on w.

For L_3, we use the same reduction as in the lecture, but we flip the accept and reject in stage (3) of H. This way we reject if M_{L_3} accepts (since M did not halt on w), and vice versa.

Exercise 3 Let $L = \{\langle M\rangle \mid \text{TM } M \text{ writes a \$ on the tape, during the first 100 steps of computation, on every input}\}$.
Is L decidable? Prove your answer.

Solution proposal Here it looks like L might be undecidable, but it is actually decidable. We could simulate M for 100 steps, and reject if a $ has not been written. We would need to simulate M on all different inputs, so it seems we would have to check infinitely many cases. The trick here, is that M only has 100 steps to write a $, including the steps it takes to read the input. Thus, two inputs that are similar in the first 100 symbols can not make the TM behave differently for the first 100 steps. This means that we only have to simulate M for 100 steps on all inputs that are different in the first 100 symbols. This is a finite number, since the input alphabet is finite.

Exercise 4 Let $EQUAL = \{\langle M_1, M_2\rangle \mid L(M_1) = L(M_2)\}$.

Is $EQUAL$ decidable? If yes, show a decider for $EQUAL$, if no, show a reduction from $HALT$.

Solution proposal The following reduction shows that $EQUAL$ is undecidable. Assume M_{EQUAL} is a decider for $EQUAL$. To decide an instance $\langle M, w\rangle$ of $HALT$, simulate M_{EQUAL} on $\langle M_1, M_2\rangle$, where M_1 is a TM that accepts all (well-formed) inputs, and where M_2 first simulates M on w, then accepts. This way, the machines will be equal iff M halts on w.

Exercise 5 Let $PI = \{k \mid$ There occurs k consecutive 0's in the decimal expansion of π $\}$.

Is PI a decidable language?

Solution proposal Yes, PI is decidable.

Case 1: there is a limit to the number of consecutive 0's in the decimal expansion of π, say n. That is, the longest sequence of 0's is n long, and no sequence of length $n + 1$ occurs. In this case, a decider for PI checks its input, k, to compare with n and accepts if $k \leq n$, and rejects otherwise. Note that if the decimal expansion contains 17 0's in a row, then it also contains 16 in a row.

Case 2: no such upper limit exists, in which case the decider can accept right away.

In any case we have created a decider for PI, so the language is decidable.

Challenge 1 Recall from the lecture Rice's theorem.

Theorem 1 (Rice's theorem) *Let R be a language consisting of Turing machine descriptions, such that R contains some, but not all Turing machine descriptions. Furthermore, let the membership in R for any Turing machine M_1, depend solely on the language of M_1, that is: $L(M_1) = L(M_2) \implies (\langle M_1\rangle \in R \leftrightarrow \langle M_2\rangle \in R)$. Then R is an undecidable language.*

Give a proof of Rice's theorem.

Hint: Show a reduction from $HALT$ to R. Find a yes-instance and a no-instance of R, and create a TM that recognizes the one of those language depending on whether M halts on w on not. The result from Exercise 1, may come in handy.

Solution proposal We assume for contradiction that M_R decides R. We will show how to decide $HALT$ by using M_R as a subroutine in H. First, let $T_\emptyset$ be a TM that always rejects, that is, $L(T_\emptyset) = \emptyset$. We assume that $T_\emptyset \notin R$. If $T_\emptyset \in R$, we could continue the proof with $\overline{R}$ instead of R. (Here we use the result from Exercise 1). Since R contains some TM T, we have our yes-instance (T) and no-instance ($T_\emptyset$) of R. Now, we create a TM called M', such that $L(M') = L(T_\emptyset) = \emptyset$ if M does not halt on w, and $L(M') = L(T)$ if M does halt on w:

M' = "On input x:

(1) Simulate M on w.

(2) If M halts: Simulate T on x.

(3) If T accepts, *accept*.

(4) If T rejects, *reject*.

Note that M' behaves like T if M halts on w and M' behaves like $T_\emptyset$ if M loops on w.

Finally, to decide $HALT$ we simulate M_R on $\langle M' \rangle$ and accept if M_R accepts and reject if M_R rejects.