CS340: Theory of Computation
Assignment 4 Solutions

Yatharth Goswami
Roll No. 191178

November 8, 2021

I Problem 1 Solution

In this problem, we were asked to find if a language is decidable or not.

- $L_1 = \{\langle M,N\rangle \mid M,N$ are two TMs and M takes fewer steps than N on input $\epsilon\}$

  We will show that L_1 is **undecidable** using technique of reduction. Consider the langugage

  $$L' = \{\langle M\rangle \mid M \text{ accepts } \epsilon\}$$

  We will first show that L' is undecidable.

  Proof. We will show this using the Rice theorem. Consider the property P such that $P(L(M)) = 1$ iff $L(M)$ contains the string ϵ. We know that there exists TMs which do not contain ϵ in their language (eg: a TM that rejects every string) and also TMs that do contain ϵ (eg: a TM that accepts every string) in their language. Hence this is a non-trivial property of languages of Turing Machines. Hence, using Rice Theorem, we have that L' is undecidable. □

  Now, we will use the fact that L' is undecidable and reduce it to our language L_1 to show that L_1 is undecidable.

  Claim 1.0.1. $L' \leq_m L_1$

  Proof. We will construct a computable function f that takes an input $\langle M\rangle$ and produces an output $\langle N_1,N_2\rangle$ such that N_1 takes fewer steps than N_2 on ϵ iff $\epsilon \in L(M)$. Now we will describe the reduction function.
  Input: $\langle M\rangle$

  - Construct a TM N_1 that does the following on seeing an input x does the following
    * If $x \neq \epsilon$ then reject.
    * If $x = \epsilon$ then simulate M on ϵ, accept if M accepts and loop infinitely in other cases. (This can be achieved by moving to a self looping state in finite state machine and then just moving the tape head to right in future).
  - Construct a TM N_2 that loops infinitely on any input, including ϵ. (This can also be achieved using similar technique as told above).

  Output: $\langle N_1,N_2\rangle$

  Proof of Correctness:
  It is easy to see that f halts since all of the steps in the algorithm are halting in nature. Now, Consider the two cases,

  - $\epsilon \in L(M) \rightarrow N_1$ accepts ϵ in finite number of steps, which in turn implies that N_1 takes fewer steps than N_2 on ϵ.

- $\epsilon \notin L(M) \rightarrow N_1$ goes into infinite loop on ϵ and this in turn implies that N_1 does not take fewer steps than N_2 on epsilon.

Therefore from the above argument, we can say that N_1 takes fewer steps than N_2 on ϵ iff $\epsilon \in L(M)$ and hence $L' \leq_m L_1$. □

Using the above claim and the fact that L' is undecidable, we get that L_1 is undecidable. Hence proved that the language L_1 is undecidable.

- $L_2 = \{\langle M \rangle \mid M$ takes at most 2^{340} on some input$\}$

We will show that this language is **decidable**. To show this, we will construct a halting TM H that does the following
Input: $\langle M \rangle$

- Run M on every input string of length at most 2^{340} for at most 2^{340} steps.
- If M accepts any of these inputs then accept, else reject.

Notice that the above turing machine is halting for any input, since it executes finite number of steps (2^{340}) on each of the finitely many (2^{340}) strings. Also notice that the algorithm is correct since the bits after the length 2^{340} are not of value to us as if any input gets accepted within 2^{340} steps, then we would have read at most 2^{340} bits from the input and hence taking strings of length at most 2^{340} suffices.

- $L_3 = \{\langle M \rangle \mid$ there are infinitely many TMs equivalent to $M\}$

This language is **decidable**, since every TM M has infinitely many TMs equivalent to it. An easy way to see this claim is to add dead states to the finite control of the given TM M and this way we can generate infinitely many TMs all equivalent to M. Hence, every TM is accepted (if the input string is a valid encoding of some TM).

- $L_4 = \{\langle M,N \rangle \mid L(M) \cap L(N)$ is infinite$\}$

We will prove that this language is **undecidable**. Consider the langugage

$$L' = \{\langle M \rangle \mid L(M) \text{ is infinite}\}$$

We will first show that L' is undecidable.

Proof. We will show this using the Rice theorem. Consider the property P such that $P(L(M)) = 1$ iff $|L(M)|$ is infinite. We know that there exists TMs which have finite $|L(M)|$ (take a language which rejects all the inputs) and also TMs which have infinite $|L(M)|$ (take a language which accepts all inputs) and hence this is a non-trivial property of languages of Turing Machines. Hence, using Rice Theorem, we have that L' is undecidable. □

Now, we will use the fact that L' is undecidable and reduce it to our language L_4 to show that L_4 is undecidable.

Claim 1.0.2. $L' \leq_m L_4$

Proof. We will construct a computable function f that takes an input $\langle M \rangle$ and produces an output $\langle N_1, N_2 \rangle$ such that $L(N_1) \cap L(N_2)$ is infinite iff $L(M)$ is infinite. Now we will describe the reduction function.
Input: $\langle M \rangle$

- Set $N_1 = M$.
- Construct a TM which accepts all the inputs (say M_{all}).
- Set $N_2 = M_{all}$.

Output: $\langle N_1, N_2 \rangle$

Proof of Correctness:
It is easy to see that f halts since all of the steps in the algorithm are halting in nature. Now, Consider the two cases,

- $L(M)$ is infinite. This leads to $L(M) \cap \Sigma^*$ being infinite. Therefore $L(N_1) \cap L(N_2)$ is infinite.
- $L(M)$ is finite. This leads to $L(M) \cap \Sigma^*$ being finite. Therefore $L(N_1) \cap L(N_2)$ is finite.

Therefore from the above argument, we can say that $L(N_1) \cap L(N_2)$ is infinite iff $L(M)$ is infinite and hence $L' \leq_m L_1$. □

Using the above claim and the fact that L' is undecidable, we get that L_4 is undecidable. Hence proved that L_4 is undecidable.

II Problem 2 solution

In this problem, we are given a language L, such that L is Turing Recognisable while $\bar{L}$ is not. Now, we were asked to comment on the language

$$L' = \{0w \mid w \in L\} \cup \{1w \mid w \notin L\}$$

We will show that both L' and $\bar{L}'$ are non-TR languages. We will prove this using contradiction.

Let us look at L' first. Assume on contrary that L' is Turing-recognisable. This means that there exists a TM M such that $L' = L(M)$. Now, using this we will construct another TM N, in this way -> On receiving any input string w, N appends 1 to the front of w and then passes it through M. It accepts if M accepts. We will now show that $L(N) = \bar{L}$. First note that any string w which lies in $\bar{L}$ will be accepted by N because all strings of the form $1w$ where $w \notin L$ are a part of $L(M)$. From this we get that $\bar{L} \subseteq L(N)$. Now, notice the fact that any string w being accepted by N, means $1w$ is accepted by M and it happens only if $1w \in L(M)$ or $w \in \bar{L}$. From this, we obtain that $L(N) \subseteq \bar{L}$. Hence, we get that $L(N) = \bar{L}$. This gives us a contradiction as $\bar{L}$ was non-TR. Hence our assumption was wrong and L' is non-TR.

Similar to the above method we will now show that $\bar{L}'$ is also non-TR. First notice that $\bar{L}'$ will look like

$$\bar{L}' = \{\epsilon\} \cup \{0w \mid w \notin L\} \cup \{1w \mid w \in L\}$$

Now, assume on contrary that $\bar{L}'$ is Turing-recognisable. This means that there exists a TM M such that $\bar{L}' = L(M)$. Now, using this we will construct another TM N, in this way -> On receiving any input string w, N appends 0 to the front of w and then passes it through M. It accepts if M accepts. We will now show that $L(N) = \bar{L}$. First note that any string w which lies in $\bar{L}$ will be accepted by N because all strings of the form $0w$ where $w \notin L$ are a part of $L(M)$. From this we get that $\bar{L} \subseteq |L(N)|$. Now, notice the fact that any string w being accepted by N, means $0w$ is accepted by M and it happens only if $0w \in L(M)$ or $w \in \bar{L}$. From this, we obtain that $L(N) \subseteq \bar{L}$. Hence, we get that $L(N) = \bar{L}$. This gives us a contradiction as $\bar{L}$ was non-TR. Hence our assumption was wrong and L' is non-TR.

Hence, we can say that L' and $\bar{L}'$ are both non-TR and hence undecidable as well.

III Problem 3 Solution

In this problem, we were asked to prove the undecidability of the following languages.

- $INFINITE_{TM} = \{\langle M \rangle \mid M$ is a TM and $L(M)$ is an infinite language$\}$

  We will show that $INFINITE_{TM}$ is undecidable using Rice Theorem. Moving ahead, I will assume here that finding if $\langle M \rangle$ denotes a turing machine is decidable problem (If not, the problem $INFINITE_{TM}$ is as it is undecidable). For showing this, consider the property P such that $P(L(M)) = 1$ iff $|L(M)|$ is infinite. We know that there exists TMs which have finite $|L(M)|$ (take a TM which rejects all the inputs) and also TMs which have infinite $|L(M)|$ (take a TM which accepts all inputs) and hence this is a non-trivial property of languages of Turing Machines. Hence, using Rice Theorem, we have that we have that $INFINITE_{TM}$ is an undecidable language.

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

  We will show that ALL_{TM} is undecidable using Rice Theorem. Moving ahead, I will assume here that finding if $\langle M \rangle$ denotes a turing machine is decidable problem (If not, the problem ALL_{TM} is as it is undecidable). For showing this, consider the property P such that $P(L(M)) = 1$ iff $L(M) = \Sigma^*$. We know that there exists TMs which do not have $L(M) = \Sigma^*$ (take a TM which rejects all the inputs) and also TMs which have $L(M) = \Sigma^*$ (take a TM which accepts all inputs) and hence this is a non-trivial property of languages of Turing Machines. Hence, using Rice Theorem, we have that we have that ALL_{TM} is an undecidable language.