Student ID: 805167986

Collaborators: Sashwath Sundher, David Tran, Dylan Gunn

CS181 Winter 2021 – Problem Set 5

Due Sunday, March 7, 11:59 pm

- Please write your student ID **and the names of anyone you collaborated with** in the spaces provided and attach this sheet to the front of your solutions. **Please do not include your name anywhere since the homework will be blind graded.**
- An extra credit of **5%** will be granted to solutions written using LaTeX. Here is one place where you can create LaTeX documents for free: `https://www.overleaf.com/`. The link also has tutorials to get you started. There are several other editors you can use. We have also posted a short LaTeX tutorial on CCLE under References.
- If you are writing solutions by hand, please write your answers in a neat and readable hand-writing.
- Always explain your answers. When a proof is requested, you should provide a rigorous proof.
- If you don't know the answer, write "I don't know" along with a clear explanation of what you tried. For example: "I couldn't figure this out. I think the following is a start, that is correct, but I couldn't figure out what to do next. [[Write down a start to the answer that you are sure makes sense.]] Also, I had the following vague idea, but I couldn't figure out how to make it work. [[Write down vague ideas.]]" At least 20% will be given for such an answer. Note that if you write things that do not make any sense, no points will be given.
- The homework is expected to take anywhere between 8 to 14 hours. You are advised to start early.
- Submit your homework online on Gradescope. The Gradescope code is 5V7GW5.
- Homework points will be scaled according to the number of homework assignments. All assignments will be weighted equally. As per the syllabus, your homework assignments will together comprise 25% of your final grade.

1. **(20 points).** Prove that the language

$$\mathsf{COMPL_{TM}} = \{\langle M_1, M_2\rangle \mid L(M_1) = \overline{L(M_2)}, \text{ where } M_1 \text{ and } M_2 \text{ are Turing machines}\}$$

is not Turing-recognizable.

Let there be a TM $M_{\mathsf{COMPL_{TM}}}$ which recognizes $\mathsf{COMPL_{TM}}$.

For contradiction we have TMs M_1 and M_2 which are constructed in the following way.

TM M_1 takes input $x \in \Sigma^*$:

(a) $M_1(x)$ **rejects** $\forall x$
$\therefore L(M_1) = \emptyset$

TM M_2 takes input x:

(a) By the recursion **recursion theorem**, $z = \langle M_2\rangle$
Simulate $M_{\mathsf{COMPL_{TM}}}(\langle M_1\rangle, z)$.

(b) If
If $M_{\mathsf{COMPL_{TM}}}$ **accepts** then **reject** x.
If $M_{\mathsf{COMPL_{TM}}}$ **rejects** then **accept** x.

Proof

If $M_{\mathsf{COMPL_{TM}}}$ **accepts** then $\overline{L(M_2)} = \Sigma^*$.

Therefore $L(M_1) \neq \overline{L(M_2)}$, which is a **contradiction**.

If $M_{\mathsf{COMPL_{TM}}}$ **rejects** then $\overline{L(M_2)} = \emptyset$.

Therefore $L(M_1) = \overline{L(M_2)}$, which is a **contradiction**.

If $M_{\mathsf{COMPL_{TM}}}$ **loops** then $\overline{L(M_2)} = \emptyset$.

Therefore $L(M_1) = \overline{L(M_2)}$, which is a **contradiction**.

$\therefore M_{\mathsf{COMPL_{TM}}}$ does not exist and $\mathsf{COMPL_{TM}}$ is not Turing-recognizable.

$\square$

2. **(20 points).** Define $\mathsf{SUBSET_{TM}}$ to be the problem of testing whether the set of strings accepted by a Turing machine, say M_1, is also accepted by another Turing machine, say M_2. More formally,

$$\mathsf{SUBSET_{TM}} = \{\langle M_1, M_2\rangle \mid L(M_1) \subseteq L(M_2)\}.$$

Show that $\mathsf{SUBSET_{TM}}$ is undecidable.

Let there be a TM $M_{\mathsf{SUBSET_{TM}}}$ which decides $\mathsf{SUBSET_{TM}}$.

For contradiction we have TMs M_1 and M_2 which are constructed in the following way.

TM M_1 takes input $x \in \Sigma^*$:

(a) $M_1(x)$ **accepts** $\forall x$.
$\therefore L(M_1) = \Sigma^*$

TM M_2 takes input x:

(a) By the **recursion theorem**, $z = \langle M_2\rangle$.
Simulate $M_{\mathsf{SUBSET_{TM}}}(\langle M_2\rangle, z)$.

(b) If $M_{\mathsf{SUBSET_{TM}}}$ **accepts** then **reject**.
If $M_{\mathsf{SUBSET_{TM}}}$ **rejects** then **accept**.

Proof

If $M_{\mathsf{SUBSET_{TM}}}$ **accepts** then $L(M_2) = \emptyset$.

Therefore $L(M_1) \not\subseteq L(M_2)$, which is a **contradiction**.

If $M_{\mathsf{SUBSET_{TM}}}$ **rejects** then $L(M_2) = \Sigma^*$.

Therefore $L(M_1) \subseteq L(M_2)$, which is a **contradiction**.

$\therefore$ decider $M_{\mathsf{SUBSET_{TM}}}$ does not exist and $\mathsf{SUBSET_{TM}}$ is undecidable.

$\square$

3. **(60 points)** We define a new notion called a "certified" language. A language L over the alphabet $\{0, 1\}$ is called "certified" if there exists a Turing machine M satisfying the following conditions:

   - For all $x \in L$, there exists a $y \in \{0, 1\}^*$ such that $M(x, y)$ accepts.
   - For all $x \notin L$ and for all $y \in \{0, 1\}^*$, $M(x, y)$ rejects.
     (We think of y as being the "certificate" that allows x to be in the language.)

   (a) **(15 points).** Let Halt_ϵ denote the language of all the Turing machines which halt on input ϵ. More formally,

   $$\text{Halt}_\epsilon = \{\langle N \rangle \mid N \text{ halts on } \epsilon\},$$

   where $\langle N \rangle$ denotes the code of the machine N. Show that Halt_ϵ is a certified language.
   Hint: Think about how the input y could be made to relate to whether or not a machine "halts".

Let there be a TM M_{Halt_ϵ} which "certifies" Halt_ϵ as defined below.

TM M_{Halt_ϵ} takes input x and y, $M_{\text{Halt}_\epsilon}(x, y)$

i. **Let** x be interpreted as the description of TM N, $\langle N \rangle$.
   Let y be interpreted as an integer n, the # of steps to simulate N for.
ii. **Simulate** $N(\epsilon)$ for n steps.
iii. **If** N **halted** then **accept**.
   If N has **not halted** then **reject**.

Claim

$\text{Halt}_\epsilon = \{\langle N \rangle \mid N \text{ halts on } \epsilon\}$ is a "certified" language.

Proof

If $x \in \text{Halt}_\epsilon$, $x = \langle N \rangle$ s.t. N will **halt** on ϵ.

If this is true then there is a finite number of steps n which we convert to a binary string y and simulate $M_{\text{Halt}_\epsilon}(x, y)$.

There **must be** a finite number of steps y which N will halt on ϵ.

Therefore the "certifier" will **accept**.

If $x \notin \text{Halt}_\epsilon$, $x = \langle N \rangle$ s.t. N will **not** halt on ϵ.

There is **not** a finite number of steps y which N will halt on ϵ.

Therefore the "certifier" will **reject** after y steps.

$\therefore$ Halt_ϵ is a "certified" language.

$\square$

(b) **(15 points).** Explain why your method does not work if you try to prove that the language $\text{Halt}_{\mathsf{all}} = \{\langle N \rangle \mid N \text{ halts on all inputs}\}$ is a certified language.

Lets try to change M_{Halt_ϵ} to satisfy $\text{Halt}_{\mathsf{all}}$.
For Halt_ϵ we simulated $N(\epsilon)$.
For $\text{Halt}_{\mathsf{all}}$ we would have to simulate N for all inputs.
If we did this then M_{Halt_ϵ} would never halt, and therefore it would not go from step 2 to 3.
Therefore we cannot use that method to prove $\text{Halt}_{\mathsf{all}}$.

(c) **(30 points).** Use the recursion theorem to prove that the language $\text{Halt}_{\textsf{all}} = \{\langle N \rangle \mid N \text{ halts on all inputs}\}$ is not a certified language.

Let there be a TM $M_{\text{Halt}_{\textsf{all}}}$ which "certifies" $\text{Halt}_{\textsf{all}}$ as defined below.
For contradiction we have TM M which is constructed in the following way.
TM M takes input s

i. **By the recursion theorem**, $z = \langle M \rangle$.
   Simulate $M_{\text{Halt}_{\textsf{all}}}(z, s)$.
ii. If $M_{\text{Halt}_{\textsf{all}}}$ **accepts** then **do not halt**.
   If $M_{\text{Halt}_{\textsf{all}}}$ **rejects** then **halt and reject**

Proof

If $\langle M \rangle \in \text{Halt}_{\textsf{all}}$, $M(s)$ will halt $\forall s$ and we know that $\exists y$ s.t. $M_{\text{Halt}_{\textsf{all}}}(\langle M \rangle, y)$ **accepts**.

If M **halts** $\forall s$, then $\langle M \rangle \in \text{Halt}_{\textsf{all}}$ but $M_{\text{Halt}_{\textsf{all}}}(\langle M \rangle, s)$ **rejects** $\forall y$,
which is a **contradiction**.

If M does **not halt** for any s, then $\langle M \rangle \notin \text{Halt}_{\textsf{all}}$ but $M_{\text{Halt}_{\textsf{all}}}(\langle M \rangle, s)$ **accepts**,
which is a **contradiction**.

$\therefore M_{\text{Halt}_{\textsf{all}}}$ does not exist and $\text{Halt}_{\textsf{all}}$ is not a "certified" language.

$\square$