

CSCI 341 Workshop 7

(Un)Decidability

November 7, 2025

Problem 1. Show that the language $L_{TM} = \{[\mathcal{T}] \mid \mathcal{T} \text{ is a Turing machine}\}$ is decidable using the following theorem.

Theorem 1 (CFL in Dec). *Every context-free language is decidable. That is, $\text{CFL} \subseteq \text{Dec}$.*

Hint: cook up a grammar for the BuckLang programming language.

Problem 2. Prove that the following language

$$L_{11} = \{[\mathcal{T}] * x \mid x \text{ halts on input } 11 \text{ in the Turing machine } \mathcal{T}\}$$

is undecidable by showing that L_ε reduces to L_{11} .

Hint: see next page for a general methodology if you get stuck.

Technique for Problem 2

- (1) Assume for a contradiction that there is a Turing machine \mathcal{E} with a state x_{11} that decides L_{11} .

We are going to build a decision procedure for L_ε using x_{11} . That is, given a Turing machine \mathcal{T} with a state x , we are going to use x_{11} to decide if x halts on input ε .

- (2) Show that the following Turing program halts on input ε if and only if x (in \mathcal{T}) halts on input 11:

```
state start
if _ : write 1.write 1.move right.goto x
if 0 : write 1.write 1.move right.goto x
if 1 : write 1.write 1.move right.goto x
[ $\mathcal{T}$ ]                (include all of the code that programs  $\mathcal{T}$ )
```

- (3) Let \mathcal{W} at state y be a Turing program that takes any string of the form $[\mathcal{T}]*x$ as input and outputs the string immediately above this followed by $*start$.

- (4) Now, what is $\mathcal{E}_{x_{11}}(\mathcal{W}_y([\mathcal{T}]*x))$ if

- A. x halts on input ε ?
- B. x does not halt on input ε ?

The definitions and theorem below, taken together, form a swift technique for telling that a given language is undecidable.

Definition 2 (Non-trivial and Extensional). Let \mathbf{TM} be the set of all pairs (\mathcal{T}, x) where \mathcal{T} is a Turing machine and x is a state of \mathcal{T} . Let $P \subseteq \mathbf{TM}$ (called a *property of Turing programs*).

- (1) P is *nontrivial* if $P \neq \{\}$ and $P \neq \mathbf{TM}$. I.e., there is at least one Turing program that satisfies the property and at least one Turing program that does not.
- (2) P is *extensional* if the following holds: for any $(\mathcal{T}, x) \in P$ and any $(\mathcal{S}, y) \in \mathbf{TM}$, if $\mathcal{T}_x = \mathcal{S}_y$, then $(\mathcal{S}, y) \in P$ also. I.e., if \mathcal{T} at x implements the same string transformer as \mathcal{S} at y , then either (\mathcal{T}, x) and (\mathcal{S}, y) both satisfy the property or neither do.¹

Theorem 3 (Rice's). Let $P \subseteq \mathbf{TM}$ be nontrivial and extensional. Then the language

$$L_P = \{[\mathcal{T}] * x \mid (\mathcal{T}, x) \in P\}$$

is undecidable.

Problem 3. Use Rice's Theorem to prove that all of the following languages are undecidable.

- (1) $L_1 = \{[\mathcal{T}] * x \mid x \text{ accepts the word } 01101\}$
- (2) $L_2 = \{[\mathcal{T}] * x \mid x \text{ does not accept the word } 01101\}$
- (3) $L_3 = \{[\mathcal{T}] * x \mid x \text{ recognizes every even-length word}\}$
- (4) $L_4 = \{[\mathcal{T}] * x \mid \mathcal{T}_x = \mathcal{U}_c \text{ where } \mathcal{U} \text{ at } c \text{ is a universal Turing program}\}$
- (5) $L_5 = \{[\mathcal{T}] * x * w \mid \mathcal{T}_x(w) = \varepsilon\}$

Why can't Rice's theorem be used to show that the language in Problem 1 is undecidable?

Hint: in each case, the defining property of the language is a property of Turing programs. Explain why the property is nontrivial and extensional.